APPLICATION OF METAHEURISTIC
IN INVENTORY ROUTING PROBLEMS

NUR ARINA BAZILAH BINTI AZIZ

THESIS SUBMITTED IN FULFILLMENT
OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF SCIENCE
UNIVERSITY OF MALAYA
KUALA LUMPUR

2013

UNIVERSITI MALAYA
ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: **Nur Arina Bazilah binti Aziz** (I.C/Passport No:**811022-09-5186**)

Registration/Matric No: **SHB050001**

Name of Degre: **Ph.D**

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):
Application of Metaheuristic in Inventory Routing Problems.

Field of Study: **Operational Research**

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright to this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date

Subscribed and solemnly declared before,

Witness's Signature Date

Name:
Designation:

ABSTRACT

The coordination of the inventory management and transportation is often known as the Inventory Routing Problems (IRP). The problem addressed in this study is outbound and inbound distribution network consisting of an assembly plant and many geographically dispersed suppliers/customers where the supplier/customers supplies/receives distinct product to/from the assembly plant. It is based on a finite horizon, multi-periods, multi-suppliers and multi-products where a fleet of capacitated homogeneous vehicles, housed at a depot, transport parts from/to suppliers/customers to meet the demand specified by the assembly plant in each periods. We propose a hybrid genetic algorithm based on allocation first, route second method to determine an optimal inventory and transportation policy that minimizes the total costs. We introduce two new representations. The first is based on a $N \times T$ binary matrix, where N and T are the number of suppliers/customers and the number of periods respectively. It determines which supplier that needs to be visited in each period. The second representation encodes a collection/delivery matrix that determines concurrently which suppliers to be visited and the amount to be collected from those suppliers in each period. To ensure that all the related constraints are not violated, a new crossover and mutation operators are introduced. Both algorithms embed a double sweep algorithm proposed by Lee et. al [1] to cluster and route the suppliers. It is observed that the simple representation produces better results for medium sized problems. We device a modification to the binary representation in order to maximize vehicles utilization and also to allow some flexibility where part of the demand in a particular period can be met in more than one period. We found that real representation performs better in larger problems and the modified algorithms are found to behave consistently better on larger problems and in problems with higher inventory holding costs.

ABSTRAK

Koordinasi pengurusan inventory dan pengangkutan dikenali sebagai Masalah Laluan Inventori. Masalah yang dikaji di dalam kajian ini ialah rangkaian agihan keluar dan rangkaian agihan masuk yang terdiri daripada satu pusat pemasangan, dan pembekal/pelanggan yang berserakan secara geografi dimana pembekal/pelanggan akan membekalkan/menerima produk kepada/daripada pusat pemasangan. Masalah ini berasaskan horizon yang terhingga, pelbagai tempoh, pelbagai pembekal, dan pelbagai produk dimana kenderaan yang berkapasiti yang ditempatkan di depoh akan mengangkut bahagian daripada/kepada pembekal/pelanggan untuk memenuhi permintaan yang ditetapkan oleh pusat pemasangan pada setiap tempoh. Kami mencadangkan Algorithm Genetik Kacukan berdasarkan kaedah pembahagian inventori dahulu, diikuti dengan laluan untuk menentukan polisi optimal bagi inventori dan pengangkutan yang dapat meminimakan jumlah kos. Dua perwakilan baru diperkenalkan didalam kajian ini. Perwakilan pertama berdasarkan matriks binari $N \times T$ dimana N nombor pembekal/pelanggan dan T ialah nombor tempoh. Matriks ini menentukan pembekal/pelanggan mana yang perlu dilawat di dalam setiap tempoh. Perwakilan kedua ialah matriks penghantaran/penerimaan yang menentukan jumlah inventori yang perlu dihantar/diterima didalam setiap tempoh. Untuk memastikan semua kekangan yang berkaitan tidak dilanggar, satu mekanisma baru untuk operasi *crossover* dan mutasi telah diperkenalkan. Kedua-dua algorithm turut dimasukkan algorithm sapuan berganda yang telah diperkenalkan oleh Lee et. al. [1] untuk mengkelompokkan dan menyusun laluan pembekal/pelanggan. Daripada pencerapan, dapat disimpulkan bahawa perwakilan yang mudah memberikan keputusan yang lebih baik untuk masalah bersaiz medium. Kami telah mereka satu modifikasi kepada perwakilan binary untuk memaksimakan penggunaan kenderaan dan juga untuk

membenarkan sedikit fleksibiliti dimana sebahagian daripada permintaan didalam tempoh tertentu dapat dipenuhi lebih daripada didalam satu tempoh. Kami mendapati perwakilan dengan nombor nyata menunjukan prestasi yang lebih baik untuk masalah bersaiz besar dan algorithm yang dibaikpulih itu memberikan keputusan yang lebih konsisten untuk masalah bersaiz besar dan masalah dengan kos pegangan inventori yang tinggi.

ACKNOWLEDGEMENTS

First and foremost I would like to express my gratitude to Allah for giving me the courage to start this journey and strength to complete it. There were difficult times and only with His helps that I can pull myself up again to finish this work I have started.

Next, I would like thank my supervisor, Associate Professor Dr. Noor Hasnah Moin for her continuous patience, support and guidance throughout the project. Without her this thesis would not have been possible.

I would also like to thank my colleagues in Department of Mathematical Sciences, Universiti Teknologi Malaysia for being so thoughtful and concern with my progress while I was writing this thesis. A special thanks to Professor Dr. Zainal Abdul Aziz and Dr. Zaitul Marlizawati for their suggestions and comments that help me to write this thesis.

To my lovely family, relatives, best friends and friends, there are no words to describe how thankful I am to have your love, supports and advices so that I can finish this Ph.D. You are my pillar of strength.

Lastly to all other people who has contributed to this work whether directly and indirectly, many thanks to you. Each of you plays a very important part in making this thesis completes.

TABLE OF CONTENTS

CHAPTER 5: MODIFIED HYBRID GENETIC ALGORITHMS (MHGA)

CHAPTER 6: CONCLUSION AND FUTURE RESEARCH

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.0 Overview

Nowadays, the role of logistics management is vital especially in manufacturing industry. Many companies are realizing that value to a customer can be created through logistics management. Customer's value can be created through product availability, timelines and consistency of delivery, ease of placing orders, and other elements of customer services. It addresses the issue of coordinating inventory replenishment policies and distribution plans in a cost effective manner.

Vendor Managed Inventory (VMI) is an emerging trend in logistics that is an example of value creating logistics. Before VMI is introduced, the conventional inventory management refers to two-way communication between customer and vendor in traditional way. Customers will monitor the inventory level and place the orders while vendor are responsible in manufacturing the products, assembling orders, loading and routing the vehicles and making deliveries. In another words, the customers independently decide the time and amount to reorder the inventory [2]. Therefore, back then, this area of study was done as a single-day problem, more on obtaining the optimal route to deliver goods. In cases like this, the amount of inventory is fixed and researchers focus on finding methods to achieve the most cost-effective route.

However, with the advancement in technology, the VMI policy becomes an alternative that is possible and more effective where a supplier manages the inventory of his customers. This implementation is a complete opposite to the conventional inventory management. Customers no longer need to contact vendors to request for delivery; instead vendors will decide the frequency and the amount of products to be delivered. Campbell and Savelsbergh have done a research with PRAXAIR [2], one of the largest industrial gases company worldwide by adopting such delivery system [3].

The vendors will take the responsibility to replace the inventory of customer. They also will be in-charge of the transportation costs and inventory holding costs. It is vital to ensure that there are no shortages of materials or excessive inventory as both are costly to any organization. Shortages of materials cause immense penalties to an organization in terms of customer goodwill or a halt in production process. Whereas having excessive inventory impose a significant sum of holding cost to the organization. Hence, inventory management for the vendors is to take into account the monitoring of inventory level then integrates transportation and inventory holding incurred so as to obtain the most minimum cost.

1.1 Background of the Problem

Supply Chain Management (SCM) is a logistics management system consisting of suppliers, manufacturers and customers. The need for the integration and coordination of various components in a SCM has been recognized as an important factor for most companies to remain competitive. Most of the activities in the SCM are inter-related and changes in one part of the SCM are likely to affect the performance of other processes.

Inventory management and transportation are two of the key logistical drivers of the SCM. Other components include production, location, marketing and purchasing [4]. The coordination of these two components, often known as the Inventory Routing Problems (IRP) is critical in improving the SCM. The IRP seeks to determine an optimal inventory and distribution strategies that minimizes the total cost [5]. The resulting inventory and transportation policies usually assign customers to routes and then determine the replenishment intervals and collection sizes for each retailer. The implementation of IRP is critical especially in a VMI replenishment system where the supplier or manufacturer observes and controls the inventory levels of its customers or retailers. One of the most important benefits of VMI is that it permits a more uniform utilization of transportation resources. This leads to a higher level of efficiency and a much lower distribution cost that often constitutes the largest part of the overall cost.

IRP can be broadly categorized according to the following criteria: planning horizon, single or multi-periods and whether the demand is deterministic or stochastic. Several other variants of IRP can also be found depending on the underlying assumptions in the models. In this study, we consider two types of distribution networks that consist of a depot, an assembly plant and N suppliers/customers. The distribution networks will be referred to as one-to-many (outbound) network and many-to-one (inbound) network. Further explanation on the distribution networks are discussed in Section 2.1.2. The problem addressed in this study is based on a finite horizon, multi-period, multi-supplier, single warehouse, where a fleet of capacitated vehicles, housed at a depot, transports/collects products to/from the suppliers to meet the demand specified by the assembly plant/suppliers for each period. The inventory holding costs are incurred at the assembly plant which also acts as the warehouse in the one to many networks. The holding cost at the suppliers is not taken into consideration in both models. The vehicles return to the depot at the end of the trip. In this model, no backordering/backlogging is allowed. However, if the demand for more than one period is collected, then the inventory is carried forward subject to product-specific holding cost incurred at the assembly plant.

Therefore, it is important to have efficient inventory management due to the fact that an optimal balance between inventory and transportation can considerably reduce the costs incurred thus saving the organization a great deal of unnecessary expenses.

1.2 Problem Statement

Inventory control and vehicle routing has traditionally been dealt separately and consequently, inventory and transportation costs are typically minimized separately. However, studies showed that their integration can have a dramatic impact on overall system performance especially in reducing the cost and time through the optimal routing and inventory policy. Therefore, this research will concentrate on solving the IRP using heuristic approach namely Genetic Algorithm (GA) method since heuristic approach can approximate the optimal solution by exploring various parts of the feasible region and gradually evolving toward the best feasible solutions.

1.3 Scopes and Objectives

In this study, we use Genetic Algorithm (GA) and design some heuristics to solve the Inventory Routing Problem (IRP). Some modification is done to improve the final results. The modeling is done using C++ with Genetic Algorithm Library (GALIB).

The objectives of this study are:

(i) to design algorithms based on Genetic Algorithm with binary and real-valued integer representations for the inbound and outbound IRP by considering various logistics conditions.

(ii) to propose a modified hybrid genetic algorithms.

(iii) to propose a new reformulation of the IRP model in order to reduce the complexity of the problem..

1.4 Organization of the Thesis

This thesis is organized as follows. Chapter 2 presents the literature review on the Inventory Routing Problems (IRP). In this chapter the concept of IRP and Vendor Managed Inventory (VMI) is explained in great details. Different types of logistics management are also explained. In the final part of this chapter, the mathematical formulation that is used in this study is presented together with the elaboration on the constraints and assumptions.

Chapter 3 constitutes the literature review on metaheuristics. There will be an explanation on Genetic Algorithms with its components. There is also description on Evolutionary Algorithms and Local Search. The applications of metaheuristics for Inventory Routing Problems are presented in this chapter.

The methodology for the first and second model in this study is explained in Chapter 4. The first model is called Hybrid Genetic Algorithms (HGA) while the second model is the Knowledge-based Genetic Algorithms (KBGA). HGA uses binary matrix representation for the chromosome. Meanwhile KBGA uses real-valued integer matrix as the chromosome representation. The crossover operators are specially designed for each of the method. For KBGA, a new mutation operator is designed. The datasets that have been used throughout this study are also explained in this chapter. In the last part of this chapter, the computational results for each of the models are tabulated.

In Chapter 5, the methodology for Modified Hybrid Genetic Algorithms (MHGA) is presented. MHGA is a modification procedure to HGA in Chapter 4 in order to maximize the utilization of the vehicle capacity. The computational results are presented and the solution approach is compared with the previous two methods to evaluate its performance. Later, some post-optimization is done using 2-opt for the results from HGA method in Chapter 4. The new results are then compared to the work done by Moin et. al [6] that uses Variable Neighborhood Search (VNS) on the same datasets.

Some reformulation is done in the IRP model used earlier in HGA and KBGA method in order to find the lower bound for the dataset by using CPLEX. In this reformulation, the route length is removed to reduce the dimension of the formulation. The lower bound is then compared to the results from HGA, KBGA and MHGA methods.

Finally, Chapter 6 concludes all the chapters in this thesis and discusses future research directions.

CHAPTER 2

INVENTORY ROUTING PROBLEMS

2.0 Introduction

Supply Chain Management (SCM) is one of the management systems to coordinate the materials and information flows between vendors, manufacturers, assembly plants and distribution centres. Many organizations find that it is crucial to have an effective SCM to compete in the business network. That is why it is important that these organizations know how to strike a balance between various logistics functions such as the inventory management, transportation, production, location, marketing and purchasing [4].

One of the aspects of SCM is to focus on distribution logistics which involve the transportation management and inventory control. Even though initially these two components have been treated separately, after some time, the relationship between these two has been considered as important. Studies showed that significant cost reduction can be seen from the integration of inventory management and vehicle routing [2, 7]. Hence more researches are being done to solve the combination of these two components and this new problem is called the Inventory Routing Problems (IRP).

2.1 Inventory Routing Problems (IRP)

Vehicle routing problem is an NP-hard problem. Thus, a combination of vehicle routing and inventory makes the IRP as a very complex problem [8]. A good coordination in making decision for inventory and transportation management will lead to an optimized IRP. While it is cheaper to have a truck full with inventory sent to a supplier, the inventory cost might increase due to the time and space needed to store the inventory before it is being consumed. That is why it is important to balance the inventory and transportation costs. IRP has been implemented in many industrial sectors such as oil and gas delivery [9]. Due to its importance, many researchers are attracted to study this area.

The common features that are usually found in IRP are a network, transportation and inventory management. A network usually consists of a warehouse (depot), multiple customers (suppliers) and an assembly plant. Meanwhile the transportation and inventory management problems exist when the vehicle capacity constraint is inserted or when there is limitation on the number of trucks being used. The main objective of IRP is to minimize the total cost by finding the optimal inventory to be delivered/picked-up and feasible routing strategy for the delivery/pick-ups.

Earlier studies in IRP focus mostly on a single period model with deterministic demand. This problem is also a classic model for vehicle routing problem (VRP). Federgruen and Zipkin [10] are among the first to study the inventory routing problem. They approach the problem as a single day problem with a limited amount of inventory and the customers' demands are assumed to be a random variable. They represent the problem as a nonlinear integer program using a generalized Benders' decomposition approach. This approach has the attributes that for any assignment of customers to routes, the problem decomposes into a nonlinear inventory allocation problem which determines the inventory and shortage costs and a Travelling Salesman Problems for each vehicle considered which produces the transportation costs. However, not all customers will be visited every day as there are the inventory and shortage costs as well as the limited amount of inventory to be considered. Later, the problem is extended for perishable products [11] where by using the integrated inventory planning and routing approach, significant cost savings have been achieved.

In 1989, Chien et al. [12] simulated a multiple period planning model based on a single period approach. This is achieved by passing some information from one period to the next through inter-period inventory flow. In their problem, there is a central depot with many customers around it. The supply capacities of the depot and the demand of the customers are fixed. An integer program is modeled using a Lagrangean dual ascent method to handle the allocation of the limited inventory available at the plant to the customers, the customer to vehicle assignments, and the routing. This is the same approach that has been implemented by Fisher et. al [13] in 1982 to solve an inventory routing problem at Air Products, an industrial gas producer. The objective of the Fisher's study is to maximize the profit from product distributions over several days. The demand is given by upper and lower bounds on the amount to be delivered to each customer for every period in the planning horizon.

Dror and Ball [14, 15] in their papers have considered the effect of the short-term over the long term planning period. They proposed a mixed integer program where consequences of present decisions on later periods are accounted for using penalty and incentive factors. In this problem, the single period models are used as sub-problems. Dror and Levy [16] use the same approach to yield a weekly schedule and apply node and arc exchanges to reduce costs in the planning period.

Since the integration of inventory and routing adds the complexity to the problem, many approaches have been designed to tackle this problem. A fixed partition policy was proposed by Anily and Federgruen [17-19] in 1990. In their earlier work, the fixed partition policies were applied on the inventory routing problems with constant deterministic demand rates and an unlimited number of vehicles. They proposed 'modified circular regional partitioning', a heuristic that can choose a fixed partition. In 1993, the problems were extended to solve the problem where the inventory can be stored at the depot [19]. A different approach based on the power-of-two (POT) principle was designed to cater this problem.

Further investigation on the fixed partition was done by Bramel and Simchi-Levi [20]. They applied the fixed partition policy in the inventory routing problem with deterministic demand and unlimited number of vehicles. To choose a fixed partition, they proposed a location based heuristic based on the capacitated concentrator location problem.

In order to obtain high quality solutions to difficult optimization problems, metaheuristics concept are introduced for Inventory Routing Problems. This metaheuristic approach is done by applying a local search procedure and a strategy to avoid local optima by performing a thorough evaluation of the search space [21]. There are many new development in this area included the hybridization of a heuristic and of a mathematical programming algorithm, namely matheuristic algorithm [22].

Recent IRP paper using some of these matheuristics techniques included iterated local search by Ribeiro and Lourenço [23]. They investigate IRP model for two types of customers namely the vendor-managed inventory (VMI) customers and the customer managed inventory (CMI) customers. The former customers have a random demand and the distributor manages the stock at the customers' location. Meanwhile, the CMI type of customers has fixed demand and there are no inventory costs for the distributor. They analyzed both the integrated solutions and the non-integrated solutions. The result shows that the inventory and transportation management in an integration model yields a better performance.

Lee et al. [1] in 2003 work on IRP which consists of multiple customers and an assembly plant in an automotive part supply chain. They address the problem as a finite horizon, multi-period, multi-customer, single assembly plant part-supply network. The objective of their study is to minimize the total transportation and inventory cost over the planning horizon. The problem is divided into two sub-problems that is vehicle routing and inventory control. To solve these problems, a mixed integer programming model is proposed using a heuristic based on simulated annealing. The purpose of using the heuristic is to generate and evaluate alternative sets of vehicle routes while a linear program determines the optimum inventory levels for a given set of routes. In their work, Lee et al. also discover that the optimal solution is dominated by the transportation cost, regardless of the magnitude of the unit inventory carrying cost. Here, it is assumed that no backordering is allowed since any shortage of parts leads to excessively high costs at the assembly plant.

In 2004, Abdelmaguid [24] proposed a construction heuristic to solve the integrated inventory distribution problem (IIDP) by considering backlogging. The backlogging will be penalized in the objective. Later in 2006, Abdelmaguid and Dessouky [25] showed that Genetic Algorithm performed better than construction heuristic to solve IIDP. In 2009, Abdelmaguid et al. [26] reviewed the heuristics for the IRP with backlogging.

Savelsbergh and Song [27] in 2008 studied the IRP with continuous moves where they tackled the problem in which a single producer cannot usually meet the demand of its customers because they are too far away. They proposed a formulation with several suppliers and trips lasting longer than one period and used a local search algorithm applied on an initial solution generated by a randomized greedy heuristic.

Michel and Vanderbeck [28] used a heuristic column generation algorithm to solve a tactical IRP. In their case, the customer demands are deterministic and the customers are served by different vehicles in their own cluster. The solutions deviated by approximately 6% from the optimum and improve upon industrial practice by 10% with respect to travel distances and the number of vehicles used.

Popović et al. [29] analyzed a multi-item IRP where different types of fuel are delivered to a set of customers by vehicles with compartments. They solved the problem using variable neighborhood search (VNS) heuristic and the results outperform the Mixed Integer Linear Programming (MILP) and the deterministic "Compartment Transfer" (CT) heuristic.

Coelho and Laporte [30] in 2013 consider multi-product multi-vehicle IRP (MMIRP) where it deals with share inventory capacity and shared vehicle capacity for all products. They solve the problem using branch-and-cut and the implementation is able to solve instances with up to five products, five vehicles, three periods and 30 customers.

2.1.1 Vendor-Managed Inventory (VMI)

Vendor Managed Inventory is a business practice where vendors monitor their customers' inventories. It was introduced around 1980's by Wal-Mart and Procter & Gamble [31]. Another alternative name for VMI is continuous replenishment, supplier-managed inventory or vendor-managed resupply. In this system, the vendor has access to the distributor's inventory data. Hence, the responsibility to decide when and how much inventory to be replenished belongs to the vendors. This is an innovative approach to inventory management where the responsibility has been shifted from the customers to the vendors. It has been accepted widely as it improves the efficiency of multi-firm supply chain. Soon after Wal-Mart and Procter & Gamble adopted this approach, Glaxosmithkline, Electrolux Italia, Nestle, Tesco, Boeing and Alco also followed suit [32].

Many benefits for both sides of vendors and customers can be gained from VMI implementation. First of all, VMI can help to reduce costs and improve services. Normally the vendors are forced to face with the uncertainty of demands from the customers which will lead to excessive finished goods inventory just to satisfy the customer's needs. However, by using VMI the vendors can plan the amount of inventory that the customer's needed and this allows smaller buffers of capacity and inventory. The vendors can also coordinate the replenishment process with more efficient routes. Hence the transportation cost can be reduced. The planning and ordering cost can also be reduced as the vendors now in charge the inventory replenishment. This is somehow will help the vendors to focus more in providing great services to the customers. Customers normally rate the services by products availability. Therefore in VMI, the vendors can coordinate the delivery by accommodating the delivery to the customers with critical stock replenishment first. By doing so, the customers are assured that their need are being prioritized by the vendors.

To ensure a good implementation of VMI, an electronic data platform such as Electronic Data Interchange (EDI) or the internet can be a system that can be placed at both the vendors and customers. The customers can give the information inside this system where the vendors can check from time to time and plan for inventory replenishment based on the information from this system. The low-cost monitoring technology makes the task to monitor the customer's inventories easier. The vendors can also get accurate information regarding the inventory status.

Even though there are many benefits of VMI, there are also some difficulties in implementing VMI in supply chain management. For example, there are cases in Spartan Stores and Kmart where the VMI vendors failed to perform a good forecasting in replenishing their inventories [32]. Other than that, lack of mutual trust and inaccurate sales and inventory data can also lead to the problems in VMI.

2.1.2 Types of Logistics Management

The objective of logistics management is to find the efficiency of operations through the integration of a few processes such as product acquisition, movement and storage activities. There are two major business processes in transportation planning that are Inbound Logistics (many-to-one network) and Outbound Logistics (one-to-many network). Both types of logistics cover the flow and storage of materials from the original point to the consumption point. The decisions connected with this two network are related with the transportation, warehousing, materials handling, inventory management, inventory control and packaging. However, each system still has some activities that are unique which makes the systems different with each other.

It is hard to tackle these two issues separately because the definition of outbound and inbound logistics is a matter of perspective. For example, if a company is a receiver of a product, the product is inbound into the company. Mean while, if the company initiate a delivery (as a raw materials supplier/manufacturer), then this is called the outbound network. The integrations of these two types of network can produce an efficient and effective management of the logistics supply chain. Therefore, the companies must find the efficient ways to store, move and transport products while at the same time keeping the inventory levels down. By having a good logistics management, other than minimizing the inbound and outbound transportation costs, the process, flexibility and customer service can also be improved.

2.1.2.1 Inbound Logistics

In the inbound logistics problem, the materials from suppliers are managed into production processes or storage facilities. It is also known as the internal focus or many-to-one network. The vehicles will start the route from the depot and visit the suppliers first. Then, the vehicles transport product from the suppliers to meet the demand specified by the assembly plant for each period. At the assembly plant, the products will go through the storing or manufacturing processes. The vehicles then will return to the depot at the end of the trip.

Figure 2.1 below shows the illustrations of an inbound logistics network.

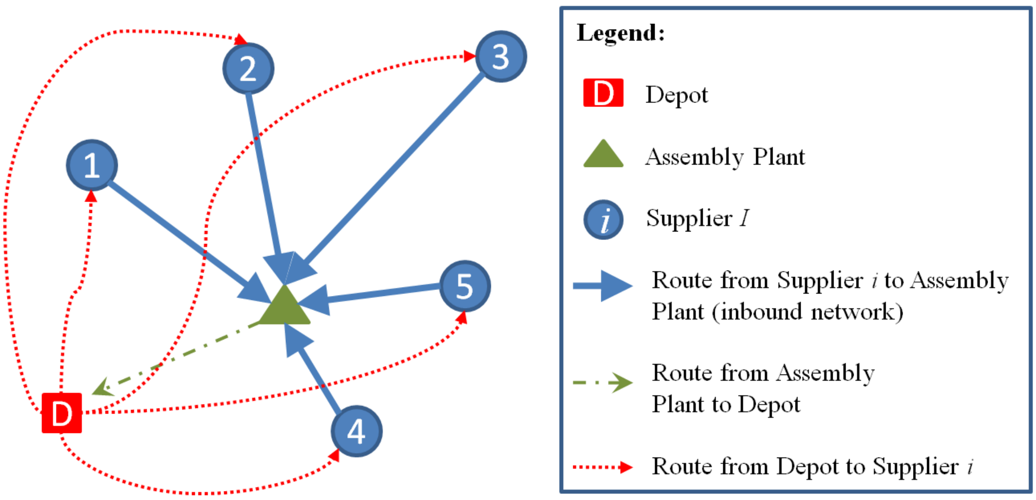

Figure 2.1: Illustration on Inbound Logistics Network

In the inbound network, the product (such as raw, unfinished product, spare parts, assembles) is moved into a firm and not away from it. Therefore, the network design is slightly different than outbound network where it does not require sophisticated transportation or warehouse system. The main focus in inbound logistics network is the material (inventory) management and procurement management. As a major part of inbound system, material management includes the planning and control of the flow of materials. This includes the procurement, warehousing, production planning, inbound transportation, receiving, materials quality control, and scrap disposal. By monitoring all these activities, a plan can be devised to ensure potential cost savings. For example, a good transportation management will ensure potential cost savings due to delivery volume or better negotiated rates with carriers.

2.1.2.2 Outbound Logistics

Another vital process in supply chain management is the outbound logistics which is also known as one-to-many network or external focus. Outbound logistic is a procedure that is related with the movement and storage of finished goods from the production line to the end user. In the outbound network, the vehicles start the route from the depot, visit the assembly plants and then transport all the products to each customer before returning to the depot. This is illustrated in Figure 2.2 below.

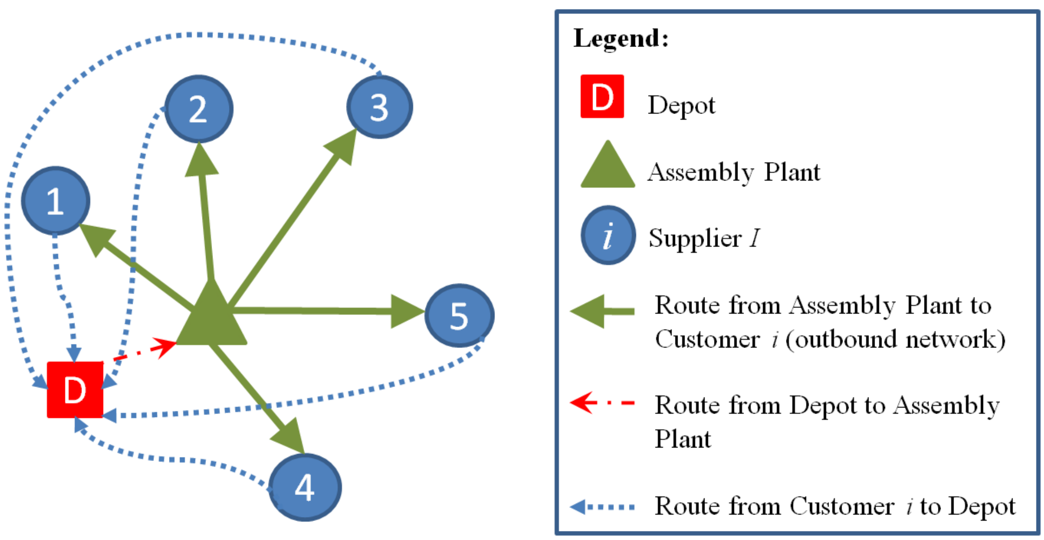

Figure 2.2: Illustration on Outbound Logistics Network.

Unlike inbound systems, outbound logistics network emphasis on customer service and distribution channels. Since this system usually handles the finished products, there are more requirements on the proper warehouse, transportation, materials handling and inventory control.

The importance of good outbound logistics network can be found from the coordinated pickup and delivery and also reduction in the shipping costs. For example, the coordinated systems between a courier service company and an assembly plant where the courier service will pick up and deliver components only when and where it is needed. The components will reach the assembly plant just in time before the assembly and installation process, which means they never go to a warehouse. This is also known as Just-In-Time inventory system and somehow can reduce the inventory holding cost in the warehouse and the amount of time in the distribution system is also reduced.

2.1.2.3 Split Delivery/Pick-up Problems

Splitting the inventory is not normally allowed in the vehicle routing problem. However, more companies are opting for this approach to increase route efficiency. Dror and Trudeau [33] have investigated that by splitting the inventory, the travel distances and the number of vehicles required can be reduced substantially. In this case, the restriction that each customer is visited once can be removed and the demand of each customer can be greater than the vehicle capacity. This means the customer can be served by multiple vehicles.

Table 2.1 : An example of split delivery problem in period 1

(a) Delivery Matrix

		Period				
		1	**2**	**3**	**4**	**5**
Customer	**1**	7	0	4	6	0
	2	8	0	0	5	0
	3	14	0	0	0	0
	4	9	0	6	0	2
	5	4	3	6	0	0

(b) Split delivery/pick-up

Truck	**Customer**	**Total Inventory per Customer**	**Amount to be Delivered/ Collected**	**Remaining Truck Capacity**
1	1	7	7	3
	2	8	3	0
2	2	8	5	5
	3	14	5	0
3	3	14	9	1
	4	9	1	0
4	4	9	8	2
	5	4	2	0
5	5	4	2	8

Table 2.1 shows the example of split delivery/pick-up problem with truck's capacity of 10 units. Table 2.1(a) is the delivery/pick-up matrix for 5 customers in 5 periods. By using the delivery/pick-up amount in Period 1, Table 2.1(b) shows the coordination of the inventory into trucks by using split inventory policy. In Truck 1, inventory Customer 1 is the first to be inserted and that makes Truck 1's content is 7. The next customer is Customer 2 with the total inventory of 8. However, since the truck's capacity is 10, the remaining capacity is 3. Therefore only 3 out of 8 units will be inserted into Truck 1. The remaining inventory (5 out of 8) will be assigned into Truck 2.

One of the approaches that have been used to solve the split delivery/pick-up problem is by using the partitioning policy. In this approach, a customer/supplier may be served by several routes which cause the inventory to be split. These routes are assumed to be controlled independently without coordination. Higher inventory cost may be resulted if these split deliveries/collections are not well coordinated [17, 19, 34].

2.1.2.4 Un-split Delivery/Pick-up Problems

In un-split delivery/pick-up problems, only one vehicle is allowed to visit a customer. The inventory cannot be splitted into different vehicles. However, in our case, an assumption has been made to allow split inventory if the number to be delivered/pick-up exceed the vehicle capacity. In this assumption, the inventory can be splitted at first for direct delivery to the customer. This means that the particular vehicle will only have one customer to be visited in its route. Meanwhile the balance of the inventory (that does not exceed the vehicle capacity) will be coordinated with another vehicle. By using the same delivery matrix in Table 2.2, the example of the un-split inventory is given below.

Table 2.2: An example of un-split delivery/pick-up in period 1

Truck	Customer	Total Inventory per Customer	Amount to be Collected	Remaining Truck Capacity
1	1	7	7	3
2	2	8	8	2
3	3	14	10	0
4	3	14	4	6
5	4	9	9	1
6	5	4	4	6

In Table 2.2, the inventory from Customer 1 is inserted into Truck 1. The remaining truck capacity is 3. However, Customer 2's inventory level is 8 and since splitting the inventory is not allowed in this case, then the inventory for Customer 2 will be put into Truck 2. As a result the capacity in Truck 1 will remain as 7. As for Customer 3, the inventory level exceeds the vehicle capacity of 10. Therefore, the inventory for Customer 3 has to be split such that Truck 3 carries 10 inventories and Truck 4 will carry the remaining inventory of 4.

2.2 Problem Formulations and Assumptions.

In our study, we consider a distribution network that is similar to Lee et. al [1]. The network consists of a depot, an assembly plant and geographically dispersed N suppliers/customers. The problem addressed in this work is based on a finite horizon, multi-period, multi-suppliers, single warehouse where a fleet of capacitated vehicles housed at a depot, transports products from the suppliers to meet the demand specified by the assembly plant for each period. The vehicles return to the depot at the end of the trip. In this model, no shortages are allowed. However, if the demand carried by the vehicle consists of amount for more than one period, then the inventory is carried forward subject to product-specific holding cost incurred at the assembly plant.

The mathematical formulation for the Inventory Routing Problem based on [1] is given below. Note that the following problem formulations are for inbound logistics distribution network. We first introduce the following notations.

Parameters

T	Period in the planning horizon
C	Capacity of the truck
F	Fixed cost per trip
V	Travel cost per unit distance
d_{it}	Demand from supplier i in period t
c_{ij}	Travel distance between supplier i and j
h_i	Unit inventory carrying cost for supplier i
J_t	Upper bound on the number of trips needed in period t

Variables

a_{ikt}	Amount collected by truck k from supplier i in period t
a_{it}	Total amount to be collected from supplier i in period t
s_{it}	Inventory level of supplier i at the end of period t

$$x_{ijkt} = \begin{cases} 1 & \text{if truck } k \text{ visits supplier } j \text{ immediately after supplier } i \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$y_{ikt} = \begin{cases} 1 & \text{if supplier } i \text{ is visited by truck } k \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

Minimize

$$\underbrace{V\sum_{i=0}^{m+1}\sum_{j=0}^{m+1} c_{ij}\left(\sum_{t=1}^{T}\sum_{k=1}^{J_t} x_{ijkt}\right)}_{(A)} + \underbrace{\sum_{i=1}^{m} h_i\left(\sum_{t=0}^{T} s_{it}\right)}_{(B)} + \underbrace{F\sum_{t=0}^{T}\sum_{k=1}^{J_t} y_{0kt}}_{(C)} \qquad (2.1)$$

subject to

$$0 \le a_{ikt} \le C \cdot y_{ikt}, \quad \forall i \in \{1,\ldots,m\}, \forall k, \forall t \tag{2.2}$$

$$\sum_k a_{ikt} = a_{it}, \quad \forall i \in \{1,\ldots,m\}, \forall t \tag{2.3}$$

$$\sum_i a_{ikt} \le C, \quad \forall k, \forall t \tag{2.4}$$

$$\sum_j x_{ijkt} = \sum_j x_{j,i,k,t} = y_{i,k,t}, \quad \forall i \in \{1,\ldots,m\}, \forall k, \forall t \tag{2.5}$$

$$\sum_{i,j \in u} x_{ijkt} \le |u| - 2, \quad u \subseteq \{1,2,\ldots,m\}, \forall k, \forall t \tag{2.6}$$

$$s_{it} = s_{it-1} + a_{it} - d_{it}, \quad \forall i \in \{1,\ldots,m\}, \forall t \tag{2.7}$$

$$x_{i0kt} = 0, \quad \forall i \in \{0,1,\ldots,m\}, \forall k, \forall t \tag{2.8}$$

$$x_{m+1,j,k,t} = 0, \quad \forall j \in \{1,\ldots,m\}, \forall k, \forall t \tag{2.9}$$

$$x_{0,m+1,k,t} = 0, \quad \forall k, \forall t \tag{2.10}$$

$$x_{m+1,0,k,t} \ge x_{ijkt}, \quad \forall i, j \in \{0,1,\ldots,m,m+1\}, \forall t \tag{2.11}$$

$$\sum_{ij} c_{ij} \bullet x_{ijkt} \le L, \quad \forall k, \forall t \tag{2.12}$$

$$s_{it} \ge 0, \quad \forall i \in \{1,\ldots,m\}, \forall t \tag{2.13}$$

$$y_{ikt}, x_{ijkt} \in \{0,1\}, \quad \forall i, j \in \{0,1,\ldots,m+1\}, \forall k, \forall t \tag{2.14}$$

Let $\{0,1,\ldots,m\}$ denotes the set of suppliers where 'supplier 0' is the depot. Meanwhile the warehouse is represented by $m+1$. For simplicity of terminology, a truck is assumed to perform one trip (route) in each period. However, this does not mean that the truck must not be used when it returns to the depot but will simply be given a different name so that 'truck' and 'trip' can be used interchangeably. J_t is an upper bound on the number of trips needed in period t in an optimal solution and it is given by $J_t = \left\lceil \frac{\sum_{\tau=t}^{T} \sum_{i=1}^{m} d_{i\tau}}{C} \right\rceil$ where $d_{i\tau}$ is the demand from supplier i in period τ.

The objective function (2.1) consists of the transportation costs (variable travel cost (A) and vehicle fixed cost (C)) and the inventory cost (B). The fixed transportation cost consists of the fixed costs incurred per trip.

Constraint (2.3) accounts for the split pick-up amount. This constraint can be omitted to cater the un-split delivery/pick-up problem. Meanwhile, constraint (2.4) ensures that the truck capacity is not violated and constraint (2.5) assures that supplier i is visited once with truck k. Constraint (2.6) serves as the sub-tour elimination constraint for each truck in each period and the inventory balance equation is given by constraint (2.7).

Since this is the formulation for the inbound logistics problem, constraint (2.8)-(2.11) ensure that no direct link from the suppliers to the depot, from the assembly plant to the suppliers, and from the depot to the assembly plant, respectively. The newly reformulated constraints for the outbound logistics network are shown as follows:

$$x_{i,m+1,k,t} = 0, \quad \forall i \in \{1,\ldots,m+1\}, \forall k, \forall t \tag{2.15}$$

$$x_{0,j,k,t} = 0, \quad \forall j \in \{1,\ldots,m\}, \forall k, \forall t \tag{2.16}$$

$$x_{m+1,0,k,t} = 0, \quad \forall k, \forall t \tag{2.17}$$

$$x_{0,m+1,k,t} \geq x_{ijkt}, \quad \forall i, j \in \{0,1,\ldots,m,m+1\}, \forall t \tag{2.18}$$

In general, the route for outbound logistics problem starts from the depot that visits the assembly plant first. From the assembly plant, the trucks will go to the customers and finally go back to the depot. Constraint (2.15) above indicates that there is no direct visits from the customers i to the assembly plant by truck k in period t. Constraint (2.16) ensures that there are no direct visits from the depot to the customers. All the trucks must go through the assembly plant first before going to the customers.

The assembly plant however are not permitted to go directly to the depot and this is given by constraint (2.17) and constraint (2.18) ensures that there is at least one visit from the depot to the assembly by truck k in period t.

The route length constraint is given by (2.12) and constraint (2.13) assures that the demand at the assembly plant is completely fulfilled without backorder. The main objective of this study is to calculate the total costs that comprise of inventory costs and total transportation costs.

2.3 Conclusion

In this chapter, we have discussed the investigations that have been done in IRP. The complexity that arises after the combination of inventory and routing problem has drawn many interests to study this area. Among the approaches that have been used in this problem are Simulated Annealing and Fixed Partition Policy.

There is also description about the types of logistics management such as inbound and outbound logistics, split and un-split delivery/pick-up problems. Finally the problem formulation and assumptions for IRP is given in this chapter based on the types of logistics managements.

CHAPTER 3

METAHEURISTICS

3.0 Introduction

Combinatorial optimization (CO) problems such as the Travelling Salesman Problem (TSP), the Quadratic Assignment Problem (QAP) and Timetabling and Scheduling problems are becoming important in both industrial and scientific world [35]. Due to the practical importance of these problems, studies to find the optimal solution are rapidly growing. One of the methods that have received a lot of attention is metaheuristics.

Since combinatorial optimization is an NP-hard problem, metaheuristics can be a method that can hopefully produce an efficient solution. Other than solving combinatorial optimization problems, metaheuristics can also be used to solve the Boolean equation [35]. Boolean equations are often used to design the digital circuits.

3.1 Metaheuristics

Metaheuristics is a heuristic method that is applied in problems with no satisfactory solution such as the combinatorial optimization. It was first introduced in the last 20 years. The function of metaheuristic is to explore a search space effectively and efficiently with the combination of a few basic heuristic methods in order to find the optimal solutions. Early on, this term were called modern heuristics. However in 1986, Glover [36] introduced the new term, 'metaheuristic' from the combination of two Greek words. *Meta* in Greek means "beyond" or "higher level". Meanwhile the original word for *heuristic* is *heuriskein* which means "to find".

Blum and Roli [35] classified metaheuristics into five characteristics.

- Nature-inspired versus non-nature inspired

  Genetic algorithms and Ant Colony Optimization are the example of the nature-inspired algorithms. Meanwhile Tabu Search and Iterated Local Search are listed as non-nature inspired algorithms. However these classifications are not really relevant as sometimes, some recent hybrid algorithms are even fit for both nature-inspired and non-nature inspired category. Therefore it will be hard to differentiate the algorithm into one of these two classes.

- Population-based versus single point search

Trajectory methods are the algorithms that work on single solution such as Tabu Search, Iterated Local Search and Variable Neighborhood Search. They encompass local search-based metaheuristics. On the other hand, the population-based metaheuristics perform search processes which describe the evolution of a set of points in the search space. This class provides clearer description of the algorithms. In addition, the current trend is to integrate the single point search algorithms in population-based ones.

- Dynamic objective function versus static objective function

Some metaheuristics modify the objective function during the search such as the Guided Local Search (GLS). There are also algorithms that maintain the objective function given in the problem representation. By modifying the search landscape, this approach will explore the search space by escaping the local minima to find the better one.

- One versus various neighborhood structures

Various neighborhood structures such as the Variable Neighborhood Search (VNS) change the fitness topology which gives the possibility to diversify the search. However, most metaheuristics algorithms work only on one single neighborhood structure.

- Memory usage versus memory-less methods

  This is an important attribute to determine the memory required during the search history. Nowadays, this criterion is recognized as one of the fundamental attributes of a powerful metaheuristics. The memory less algorithms perform a Markov process and use the current state of the search process to determine the next action.

As described earlier, the most important metaheuristics' classification is the single point versus population-based search as it gives the clearer view of the algorithms. Therefore more explanation will be done on the *trajectory methods* and the population-based methods. The *trajectory methods* perform by searching the search space by a trajectory characteristic. In this method, a successor solution may or may not belong to the neighborhood of the current solution. Under the trajectory methods, there are a few strategies such as Basic Local Search, Simulated Annealing (SA), Tabu Search (TS) and Explorative Local Search Methods.

The next method explained in [35] is the population-based methods. This method works with a set of solutions instead of with a single solution. By incorporating a learning component, this metaheuristic will produce a natural, instrinsic way for the exploration of the search space. Evolutionary Computation (EC) and Ant Colony Optimization (ACO) are listed as the most studied population-based methods.

Simulated Annealing (SA), Genetic Algorithms (GA), Ant Colony Optimization (ACO), scatter search and tabu search are among the popular metaheuristics that have been used in combinatorial optimization problems. All these methods are also known as the Evolutionary Algorithms.

3.2 Evolutionary Algorithms (EA)

Evolutionary Algorithms are the sub-field of metaheuristics. It is the generic, population-based, search method that mimics the biology-inspired mechanisms such as mutation, crossover, natural selection and survival of the fittest [37]. Being the generic, population-based method makes EAs different than the traditional method such as the Tabu Search which uses the single point search.

EAs apply the principle of survival of the fittest which will produce better approximations to a solution. The selection process is normally competitive in order to rule out poor solutions and this is done by finding the higher fitness solutions. The selected individuals will undergo the recombination process to produce new individuals that are better suited to their environment than the previous individuals. Solutions are also mutated by changing a single element of the solution. Both recombination and mutation procedure are done to generate new solutions that are biased towards the space where good solutions have already been seen.

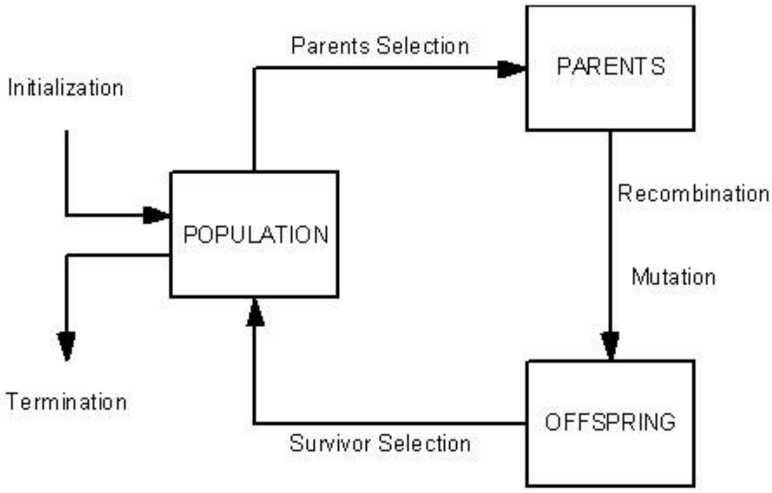

Figure 3.1: Evolutionary Algorithm mechanism

Figure 3.1 shows the important components [38] that must be specified in order to define a particular EA. These methods and operators of EAs contain the natural processes in the biological evolution and will be explained in the following sections.

3.2.1 Selection Process

The selection mechanism is to distinguish the individuals in the population based on their quality to allow better individuals to become parents of the next generation. The basic step for selection is the evaluation process by using the fitness function. Basically this step is the heuristic estimation of the solution quality to facilitate improvement. Based on this evaluation, the best individuals in the population are chosen for mating (recombination). Proportional fitness assignment, rank-based fitness assignment and multi-objective ranking are the examples of the fitness evaluation function that can be used to evaluate the individuals.

After the evaluation, the actual selection is performed by using one of these algorithms:

- Roulette-wheel selection

Roulette-wheel is the simplest and most common selection process where the individuals are selected based on their fitness scale. The main concept in this type of selection is the fittest individuals will have a greater chance of survival compared to the weaker ones. The individuals are mapped in segments according to its fitness rate. Table 3.1 shows the example of five individuals with their fitness rate and selection probability. Individual 1 is the fittest in this group and therefore it has the largest segment/interval. Meanwhile, individual 4 is the second least fit with fitness rate of 0.5 and this gives individual 4 the smallest interval. Individual 5 has 0.0 fitness rate and therefore will has no chance for reproduction.

Table 3.1 : Fitness scale and selection probability

Number of Individual	1	2	3	4	5
Fitness rate	2.0	1.5	1.0	0.5	0.0
Selection probability	0.4	0.3	0.2	0.1	0.0

The next step is to generate a sample of 3 random numbers independently from a uniformly distributed random numbers (between 0.0 to 1.0) for selection process.

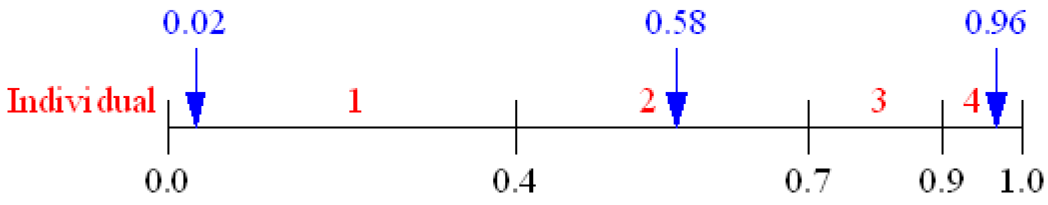

Figure 3.2: The roulette-wheel selection

Figure 3.2 shows the mapping of the individuals in Table 3.1 onto contiguous segments of a line. The intervals between the individuals are based on the size of selection probability. Three random numbers are selected that is 0.02, 0.58 and 0.96 which means the selected individual for reproduction are Individual 1, Individual 2 and Individual 4, respectively. Even though this concept is totally random and no bias, it does not guarantee minimum spread. However, other common techniques such as stochastic universal sampling (SUS) or tournament selection are often used because they have less stochastic noise, easy to implement and have constant selection pressure.

❖ Stochastic universal sampling (SUS)

Another technique used to select potential individual for selection is stochastic universal sampling (SUS). Unlike the roulette-wheel method, SUS is not bias and has minimal spread. In the method, only a single random value are used to sample the solutions which are chosen in an evenly interval space. By using the same initial step in the roulette-wheel selection, the individuals are mapped as segments on a line. The difference between these two types of selection is that SUS has equally spaced pointer placed over the line. The distance between pointers is calculated by dividing 1 with the number of individuals to be selected. For example, if the number of individuals to be selected is 3, then the space between pointers is 0.333. For SUS, only one sample of random number is needed from the range of [0.0, 0.333]. Figure 3.3 shows the illustration of SUS selection.

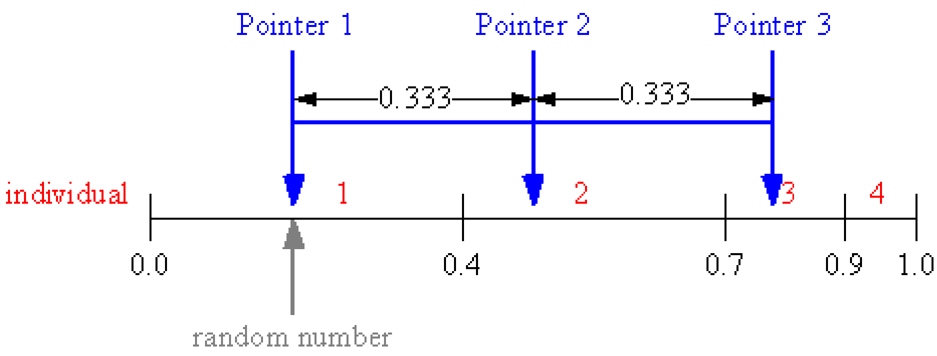

Figure 3.3: Stochastic Universal Sampling selection

❖ Local selection

In local selection, the individuals are selected within the constraint environment called the local neighborhood. The first step is to determine the first half of the population in order to find the neighborhood. This can be done in random order or by using the selection method such as stochastic universal sampling or truncation selection. Once the initial individual is determine, a local neighborhood can be defined based on the structure of the neighborhood.

❖ Truncation selection

Truncation selection is an artificial selection method that normally used for large population size. In this type of selection, the candidates are ordered by fitness and some proportion of the fittest invidual is selected. The proportion for selection is chosen based on parameter called truncation threshold. This parameter indicates the proportion of the population that will be selected to be parents who will produce uniform at random offspring. Normally, the truncation threshold was selected within the range of 50%-100%. However, this method is less sophisticated compared to the other selection method. Thus, it is rarely used in practice.

- Tournament selection

Tournament selection is a method where individuals are selected randomly from the population. This selection process will be repeated until the tournament size is reached. After that, these individuals will go through ‘tournaments’ in order to find the best and fittest individual will be selected for crossover operation. Tournament selection has selection pressure which is the degree to which the better individuals are favored. In this case, the selection pressure is done by controlling the tournament size. The larger the tournament size is, the smaller chances of selecting the weak individuals.

3.2.2 Recombination

This is the first variation operators that will create new individuals from old ones. The principle behind recombination is that by mating two individuals with different but desirable features, an offspring with both of those features can be produced. The following algorithms can be applied depending on the representation of the variables of the individuals.

- Real-valued recombination
  - Intermediate recombination
  - Line recombination
  - Extended line recombination

- Binary valued recombination
  - Single-point / double-point / multi-point crossover
  - Uniform crossover
  - Shuffle crossover
  - Crossover with reduced surrogate

3.2.3 Mutation

Mutation is the second variation operator which is normally done after the recombination. The offspring variables will be mutated by small perturbations stochastically. Generally, mutation is supposed to cause a random, unbiased change. While the recombination process involves multiple individuals in its process, mutation is usually applied on one individual.

3.2.4 Reinsertion

The last procedure in the structure of an Evolutionary Algorithms is reinsertion. Once the parents have been recombined and mutated, new offspring will be produced to be inserted into the population. This reinsertion scheme will determine which parents to be replaced and which individuals can be inserted into the population. There are two types of reinsertion that is global reinsertion and local reinsertion.

For each type of reinsertion there are a few different schemes available. For example, in global reinsertion there are pure reinsertion, uniform reinsertion, elitist reinsertion, and fitness-based reinsertion. In the pure reinsertion, the new offspring will replace all parents in the population. For both uniform and elitist reinsertion, the number of the offspring produced is less than the number of parents. However, in uniform reinsertion, the offspring will be selected to replace the parents in the population where the parents will be selected uniformly at random. On the other hand, in elitist reinsertion, the worst parents will be selected to be replaced. Finally in the last scheme in global reinsertion is the fitness-based reinsertion. In this scheme, the offspring produced are more than the number of parents. Later only the best offspring will be selected to replace the parents in the population.

As for the local reinsertion, the available schemes are similar to the global reinsertion except that the local reinsertion the selection is done within the bounded neighborhood. Hence, this reinsertion method preserves the locality of the information in the neighborhood.

There are several Evolutionary Algorithms family that has been developed independently such as Genetic Programming (GP), Evolutionary Programming (EP), Evolutionary Strategies (ES), Learning Classifier Systems (LCS) and Genetic Algorithms (GA). These methods have been applied on various problems and often perform well as EA does not make any assumption about the underlying fitness landscape. EA tends not to only stop on a local minimum but instead it can find globally optimal solutions. Hence, this method is well suited for a wide range of combinatorial and continuous problems. Different EA techniques are applied on different domains depending on the nature of the particular applied problem. For example, GPs are suitable to solve the problems that require the determination of a function that can be simply expressed in a function form. ES and EP can be used to optimize continuous functions. Meanwhile, GA can perform well on combinatorial problems. EAs have been successfully applied on many optimization problems in the fields such as engineering, art, biology, economics, marketing, genetics, operations research, robotic, social sciences, physics, politics and chemistry [39].

All these approaches have been modified over time according to the variety of the problems faced by the researchers. However, after Goldberg published his book *"Genetic Algorithms in Search, Optimization and Machine Learning"* in 1989, the interest in Genetic Algorithms grew exponentially. Hence, GA became the most popular evolutionary algorithms compared to other approaches.

3.3 Local Search Heuristics

Local search is a metaheuristic that is used to solve computationally hard or NP-hard optimization problems. Local search heuristics find the optimal solution by moving from one solution to another in the search space. It has been applied widely to numerous hard computational problems, including problems in scheduling, Very Large Scale Integration design, network design, distributed planning and production control and many other fields. Local search shows a very good performance in these studies by efficiently computing near-optimum solutions to problems of realistic sizes.

A local search algorithm works by starting from a candidate solution. Then it will iteratively move to the neighbor solution in the search space and the move is only performed if the resulting solution is better than the current solution. The algorithm will stop when it finds a local minimum.

Local search is categorized as the trajectory methods because the search process of these methods are done by a trajectory in the search space [35]. There are two types of local search that is the basic local search and explorative local search. Basic local search is also known as an iterative improvement. Meanwhile in the explorative local search method, there are Greedy Randomized Adaptive Search Procedure (GRASP), Variable Neighborhood Search (VNS), Guided Local Search (GLS) and Iterated Local Search (ILS). These are the recently proposed trajectory methods.

3.4 Genetic Algorithms (GA)

As mentioned in Section 3.2, Genetic Algorithm (GA) is a part of Evolutionary Algorithm (EA) that imitates the process of natural evolution such as mutation, crossover, selection and survival of the fittest. GA is a powerful searching tool and has been used greatly to find the solution for optimization and search problems. The name genetic algorithm originates from the analogy between the representations of a complex structure by means of a vector of components where this idea is familiar to biologist as the genetic structure of a chromosome. In selective breeding of plants and animals for example, offspring are sought which have certain desirable characteristics which are determined at the genetic level by the way parents' chromosomes combine. In a similar way, in seeking better solutions to complex problems, we often intuitively combine pieces of existing solutions.

Genetic Algorithm was introduced by J.H. Holland in 1970's and since then, has proved its capability in solving scientific and engineering problems. This method is an adaptive learning heuristic and they are generally referred to in plural because several versions exists that are adjustments to different problems. Some characteristics of GA that distinguishes them from other heuristics are:

- GA work with coding of the solutions instead of the solutions themselves. Therefore, a good, efficient representation of the solution in the form of a chromosome is required.

- They search from a set of solutions, different from other metaheuristics like Simulated Annealing and Tabu Search that start with a single solutions and move to another solution by some transition. Therefore they do a multidirectional search in the solution space, reducing the probability of finishing in a local optimum.

- They only require objective function values, not continuous searching space or existence of derivatives. Real life examples generally have discontinuous search spaces.

- GA is nondeterministics, i.e. they are stochastic in decision, which make them more robust.

However, initially there are two shortcomings of GA that cause some people not to prefer this method. The first flaw is the long computational process and the second flaw is the nature of randomness that leads to a problem of performance assurance. Nonetheless, GA still become a popular search method especially after the evolution of low-cost but fast-speed small computers that help to speed-up the computational process. The main components in Genetic Algorithms are:

- Representation (definition of the individuals)
- Objective and fitness value
- Parents selection mechanism
- Variation operators – crossover and mutation
- Survivor selection mechanism (replacement/reinsertion)

3.4.1 Representation (Definition of Individuals)

Genetic representation is a way of representing solutions or individuals from the real world problem into the evolutionary computation methods. It is because the search process for the solutions can only be done in the evolutionary computation world. Therefore, it is important to map the original problem context (also known as phenotype space) into the genetic algorithm world (genotype space). The process of converting the phenotypes into genotypes is called representation. There are two ways of representation where the first one is to map the objects from the phenotypes to the genotype space which is known as encoding. On the reverse, decoding is a process of inverse mapping the genotypes to phenotypes. In the real world problems (phenotype space), the points of possible solutions are often denoted as phenotype, individuals or candidate solution. Meanwhile, the elements in the genotypes space are called genotypes, chromosomes, and also individuals. Genetic representation can encode appearance, behavior, and physical qualities of individuals. It is quite complicated to design a good genetic representation that is expressive and evolvable because the differences in genetic representation are one of the major criteria drawing a line between known classes of evolutionary computation.

The most classical approach for GA chromosome representation is linear binary representation or bit-string encoding where the chromosomes are represented with arrays of bits such as an array of bits. This is shown in Figure 3.4a). This type of representation is quite popular among GA researchers because of its simplicity and traceability. Recently, the interest in the manipulation of real-valued representation has increased. The representation such as an array of integer as shown in Figure 3.4b) was introduced specially to deal with real parameter problems.

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 5 & 4 & 0 \\ 9 & 0 & 3 \\ 3 & 2 & 1 \end{bmatrix}$$

a) Binary chromosome representation b) Real-valued representation

Figure 3.4: The example of chromosome representation in Genetic Algorithms.

3.4.2 Objective and fitness value

The evaluation function or fitness function is the basis for selection. The main role of this function is to facilitate improvements and modifications. As the source of the evaluation, objective function can provide the mechanism for evaluating the 'fitness' of each chromosome. This is where the chromosome is taken as input and produces the objective value which is considered as the chromosome's performance. It is important to note the different between fitness and objective value. The objective value is the value from the objective function where it is the raw performance evaluation of a genome. On the other hand, the fitness value is a possibly-transformed rating used by the genetic algorithm to determine the fitness of individuals for mating. This usually can be obtained by a linear scaling of the raw objective scores.

3.4.3 Parents Selection Mechanism

The main objective of this mechanism is to distinguish the individuals in the population based on their quality. It is important because proficient parents will produce good offspring and thus will lead to better generation. Basically the method of selection for GA is the same with the selection mechanism for Evolutionary Algorithms as explained in subsection 3.2.1.

3.4.4 Variation Operators

The role of variation operators is to produce new individuals from old ones. There are two common variation operators in GA that are mutation and crossover. The one-point crossover was inspired by the natural biological process. However, this operator cannot be applied in certain situation. To solve this problem, other types of crossover operators such as multipoint crossover and uniform crossover are introduced. As a result, the performance of the newly generated offspring is greatly improved because the resultant offspring contains a mixture of genes from each parent. The example for the multipoint crossover and uniform crossover are shown in Figure 3.5 and Figure 3.6, respectively.

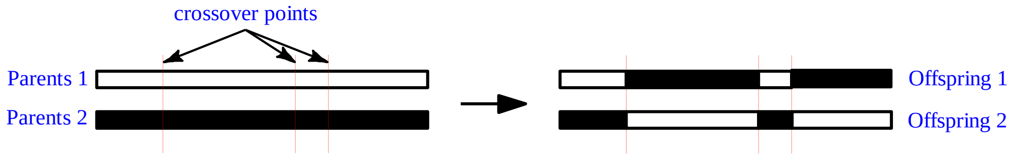

Figure 3.5: Example of multipoint crossover

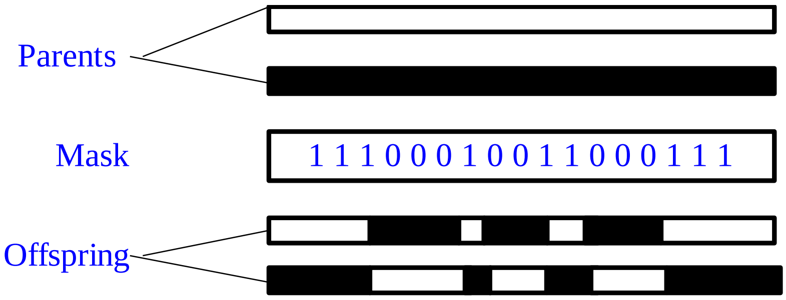

Figure 3.6: Example of uniform crossover

Mutation is the second variation operators in GA and it is used to maintain genetic diversity from one generation of a population of chromosomes to the next. A common method of implementing the mutation operators involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified. The purpose of mutation is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only taking the fittest of the population in generating that the next but rather a random selection with a weighting toward those that are fitter. The example of the binary representation is shown in Figure 3.7

Parent 1: (1 0 **1** 0 1 1 **0**) ⟶ Child 1: (1 0 **0** 0 1 1 **1**)

Figure 3.7 : A binary representation of the mutation

3.4.5 Survivor Selection Mechanism (Replacement/Reinsertion)

After the new offspring are generated, there are strategies to replace the old generation in the population. This is called the replacement or reinsertion mechanism. There are a few concepts of replacement in GA. In the steady-state GA, the chromosomes in population with size N will be replaced completely by the new offspring. Since the best chromosome of the population may fail to reproduce offspring in the next generation, this concept is usually accompanied with the elitist strategy. This strategy ensures that one or a number of best chromosomes can be copied into the succeeding generation. Basically the reinsertion for genetic algorithms are very much similar with the evolutionary algorithm as explained in Subsection 3.2.4.

3.5 Metaheuristics in Inventory Routing Problems

In recent years, metaheuristics have been applied on various numbers of problems such as scheduling problems, vehicle routing problems, inventory routing problems and facility location problems in order to enhance the efficiency of the search process. Unlike the classical heuristics, metaheuristics will not stop at the local optima. Instead, it will continue explore the search space for more possible solutions.

Tabu search is one of the popular heuristics used to find the solutions in IRP. Cousineau-Ouimet [40] has applied the tabu search heuristic in the IRP with multi-customers and multi-vehicle instances. However, some modifications are made on the attributes of the solution. Tabu search heuristic has the notion of memory which help to keep track of all the solutions that have been visited. This attribute is important as it can avoid cycling between the solutions. Although this approach is flexible and efficient, Cousineau-Ouimet found a limitation by the length of the period.

Lee et. al [1] work on IRP which consists of multiple suppliers and an assembly plant in an automotive part supply chain. They address the problem as a finite horizon, multi-period, multi-supplier, single assembly plant part-supply network. The objective of their study is to minimize the total transportation and inventory cost over the planning horizon. The problem is divided into two sub-problems that is vehicle routing and inventory control. To solve these problems, a mixed integer programming model is proposed using a heuristic based on simulated annealing. This simulated annealing is used to control the search process. The heuristic generates and evaluates alternative sets of vehicle routes while a linear program determine the optimum inventory levels for a given set of routes. The authors also observed that the optimal solution is dominated by the transportation cost, regardless of the magnitude of the unit inventory carrying cost. In this model, it is assumed that no backordering is allowed since any shortage of parts leads to excessively high costs at the assembly plant.

Sindhuchao et. al [41] develop a mathematical programming approach for coordinating inventory and transportation decisions for an inbound commodity collection system. Their problem consists of multiple suppliers, multiple items and one warehouse. Each supplier can produce one or more non-identical items. In order to find the minimum cost, they formulate the problem into the set partitioning problem. The lower bound for the total cost is then determined using the column generation approach and the optimal assignment of vehicle is found using the branch-and-price algorithm. To find the near-optimal solutions of the problems, two greedy constructive heuristics and a very large scale neighborhood (VLSN) search algorithm have been proposed.

Abdelmaguid and Dessouky [25] proposed a new genetic algorithm for the integrated inventory distribution problem (IIDP). They formulated the IIDP as the non-linear programming model. The initial genetic representation is designed based on the delivery schedule in the form of 2-dimensional matrix. In the construction phase, they use the randomized version of the Approximate Transportation Costs Heuristics (ATCH) to estimate the transportation cost value for each customer in every period in the planning horizon. After the initialization process, the improvement is done through the crossover and mutation operator. In the crossover mechanism, they introduce two types of matrix breakdown which is either vertically or horizontally. However, the vertical breakdown although will maintain the vehicle capacity constraint, it will violate the customer storage capacity. Thus it will lead to extra unnecessary inventory or backorder. Therefore, they used the horizontal breakdown with a repair mechanism for the vehicle capacity violations. In the horizontal breakdown, the delivery schedule for a selected customer for two selected parents will be exchanged to produce another offspring. Hence, although the inventory decision for the customer will be retained, the vehicle capacity constraint will be violated. Figure 3.8 shows the illustration of the horizontal breakdown.

The mutation operator has been designed to investigate partial delivery and conduct the bit exchanges randomly. This partial delivery will provide better solutions as more transportation and shortage costs can be saved.

Parent 1

		Period			
		1	2	3	4
Customer	1	51	58	0	0
	2	***5***	***15***	***29***	***0***
	3	35	42	79	0
	4	61	0	63	0
Remaining Veh Cap		48	85	29	200

Parent 2

		Period			
		1	2	3	4
Customer	1	33	76	0	0
	2	**49**	**0**	**0**	**0**
	3	53	63	0	40
	4	34	57	0	33
Remaining Veh Cap		31	4	200	127

Child 1

		Period			
		1	2	3	4
Customer	1	51	58	0	0
	2	**49**	**0**	**0**	**0**
	3	35	42	79	0
	4	61	0	63	0
Remaining Veh Cap		4	100	58	200

Child 2

		Period			
		1	2	3	4
Customer	1	33	76	0	0
	2	***5***	***15***	***29***	***0***
	3	53	63	0	40
	4	34	57	0	33
Remaining Veh Cap		75	-11	171	127

Figure 3.8 : Illustration of horizontal breakdown

Zhao et. al [34] focus on the integration of inventory control and vehicle routing schedules for a distribution system. Their problem consists of multiple customers, single item and single warehouse. The demand rate and holding cost in this problem is set to be deterministic, customer-specific and constant, respectively. It is assumed here that no inventory capacity constraint is imposed on neither the warehouse nor the customers. This study adopt the fixed partition policy for their problems and set the replenishment interval using the power-of-two (POT) policy. The POT policy is set such that the replenishment intervals are the power of two multiples of the base planning periods. Then the lower bound for the problem is found for any feasible strategy in the distribution system. Zhao et. al use tabu search algorithm to find the optimal region partition in their problems. The study shows that the policies and algorithms give effective results. However, some modification must be done on the tabu search algorithm to make it compatible with other problems.

3.6 Conclusion

Metaheuristic is a heuristic method that is applied in problems with no satisfactory solution such as the combinatorial optimization. This chapter describes the characteristic of metaheuristic. The metaheuristics are categorized under Evolutionary Algorithms which is the generic, population-based, search method that mimics the biology-inspired mechanisms.

Studies using metaheuristics in IRP are also discussed in this chapter. Among the metaheuristics used to solve IRP are Tabu Search, Genetic Algorithms, Simulated Annealing and Very Large Scale Neighborhood search algorithm.

CHAPTER 4

HYBRID GENETIC ALGORITHMS (HGA) AND KNOWLEDGE-BASED GENETIC ALGORITHMS (KBGA)

4.0 Introduction

This chapter will discuss two methods that have been applied to solve the Inventory Routing Problems. Hybrid means combining two components to produce different types of results. Hence, in Hybrid Genetic Algorithms, two heuristics will be combined to find the solutions for IRP. The first heuristic is Genetic Algorithms (GA) while the second heuristic is the Double Sweep Algorithm (DSA).

GA is a well-known, powerful searching tool which strikes a remarkable balance between exploration and exploitation of the search space. It has been used successfully on optimization problems such as wire routing, scheduling, transportation problems and travelling salesman problems (TSP). One of the studies that uses Genetic Algorithm for IRP is done by Abdelmaguid and Dessouky [25]. They proposed a multi-period IRP and allow backorders in their integrated inventory distribution problem (IIDP). This backorder is penalized later in their objective function.

4.1 Double Sweep Algorithm (DSA)

One of the heuristics in this study is Double Sweep Algorithm (DSA). This heuristic was originated from the Sweep Algorithm proposed by Gillett and Miller [42] in 1974 to solve the vehicle dispatch problem. Their results show that this algorithm performs slightly better compared to other approaches to this problem. In 2003, Lee [1] introduced the modified version of the algorithm which is named as Double Sweep Algorithm (DSA). The purpose of using DSA is to arrange the customers in order to construct the initial routes. This new algorithm creates the route by first sweeping in the vertical dimension to form cluster (of suppliers). Secondly, it will sweep in the horizontal direction to determine the routing within each cluster. This heuristic is applied to each period since the routes may change based on demand.

For simplicity, the following algorithm is given for one period only which means the index t for period is not used in this algorithm. Let d_i be demand from supplier i, and J is the upper bound on the number of trucks which is given by

$J = \sum_{i=1}^{m} \frac{d_i}{C}$ where C is the truck capacity. The Double Sweep Algorithm for inbound logistic network is given below:

1. Rotate the suppliers $(s_1, s_2, \ldots, s_m)$ and the Assembly Plant (s_{m+1}) around the depot (s_0) so that the Assembly Plant has the same y-coordinate value as the depot.

2. Sort the suppliers $(s_1, s_2, \ldots, s_m)$ according to their y-coordinate values; let $s_{(i)}$ be the ith supplier after the sort. Set $i = 1$ and $k = 1$. Open a cluster $C_k = \{\ \ \}$ and set $Q_k = 0$ where Q_k is total pick-up quantity assigned to cluster k.

3. (Clustering): If $Q_k + d_{(i)} < C$, assign $s_{(i)}$ to C_k (i.e set $C_k = C_k \subset \{s_i\}$); set $Q_k = Q_k + d_{(i)}$ and $a_{(i)k} = d_i$. Otherwise set $a_{(i)k} = C - Q_k$ and $Q_k = C$; set $k = k + 1$ and open a new cluster $C_k = \{\ \ \}$, assign $s_{(i)}$ to C_k; set $a_{(i)k} = d_i - a_{(i)k-1}$ and $Q_k = a_{(i)k}$. If $i > m$, set $k = 1$ and go to Step 4; otherwise, set $i = i_1$ and repeat Step 3.

4. (Routing): Sort the suppliers within cluster C_k according to their x-coordinate values. Let ${s_{(i)}}^k$ be the ith supplier in cluster C_k after the sort. If the x-coordinate of the supplier $m + 1$ is greater than or equal to the x-coordinate of supplier 0, form a route that starts at depot s_0, visits supplier ${s_{(1)}}^k, {s_{(2)}}^k, \ldots, {s_{(|C_k|)}}^k, s_{m+1}$ and finally returns to s_0. Otherwise form a route that starts at supplier s_0, visit its supplier ${s_{(|C_k|)}}^k, {s_{(|C_k|-1)}}^k, \ldots, {s_{(1)}}^k, s_{m+1}$ and returns to s_0. If $k > J_t$, STOP; otherwise set $k = k + 1$ and go to Step 4.

4.2 Problem Definition and Assumptions

Our model consists of a depot, an assembly plant and geographically dispersed *N* suppliers/customers. The problem addressed in this work is based on a finite horizon, multi-period, multi-suppliers, single warehouse where a fleet of capacitated vehicles housed at a depot, transports products from the suppliers to meet the demand specified by the assembly plant/customers for each period. At the end of the delivery trip, the vehicles will return to the depot.

The other assumptions of the model are listed below:

1. No shortages is allowed since it will incur excessive cost,
2. An unlimited number of capacitated and identical vehicles are available at the depot; all the vehicles have to return to the depot upon completion of a route,
3. The locations of the assembly plant, the suppliers and the depot are given and fixed,
4. The route length for any truck may not exceed a user-specified limit,
5. The transportation cost per trip consists of a fixed charge incurring for each trip plus a variable cost proportional to the travel distance,
6. A supplier may be visited by one or more trucks in any given period,
7. The planning horizon is finite and given.

There are two types of distribution networks that have been studied that are inbound and outbound logistics networks. Thorough explanation on these two networks has been given in Chapter 2. The mathematical formulation for Inventory Routing

Problems specifically for inbound logistics network is given in Section 2.2. However some of the constraints must be reformulated to cater for the outbound logistics network. From Section 2.2, constraints (2.8)-(2.11) can be changed into the following constraint for outbound logistics problems.

$$x_{i,m+1,k,t} = 0, \quad \forall i \in \{1,\ldots,m+1\}, \forall k, \forall t \tag{4.8}$$

$$x_{0,j,k,t} = 0, \quad \forall j \in \{1,\ldots,m\}, \forall k, \forall t \tag{4.9}$$

$$x_{m+1,0,k,t} = 0, \quad \forall k, \forall t \tag{4.10}$$

$$x_{0,m+1,k,t} \geq x_{ijkt}, \quad \forall i, j \in \{0,1,\ldots,m,m+1\}, \forall t \tag{4.11}$$

In general, the route for outbound logistics problem starts when the trucks depart from the depot and go straight to the assembly plant to collect the products. From the assembly plant, the trucks will deliver the products to the customers and finally go back to the depot. Constraint (4.8) above indicates that there is no direct visits from the customer i to the assembly plant by truck k in period t. Constraint (4.9) ensures that there are no direct visits from the depot to the customers. All the trucks must go through the assembly plant first before going to the customers. The assembly plant however are not permitted to go directly to the depot and this is given by constraint (4.10) and constraint (4.11) ensures that there is at least one visit from the depot to the assembly by truck k in period t.

4.3 Datasets

In this study, the datasets were downloaded from http://mie.utoronto.ca/labs/ilr/IRP and are used by Lee et al.[1]. However, the paperwork was unpublished and some of the results violate the route length constraint.

There are four original datasets that is S12T14, S20T21, S50T21 and S98T14 that comprises of (12 customers/suppliers, 14 periods), (20 customers/suppliers, 21 periods), (50 customers/suppliers, 21 periods) and (98 customers/suppliers, 14 periods) respectively. Note that SNTt refers to an instant with N customers/suppliers and t periods.

11 more datasets are created from the original 4 datasets by varying the number of periods to represent small, medium and large size problems. The location of the suppliers for S12T14, S20T21 and S50T21 are generated randomly in a square of 100×100. The locations of the suppliers for the S20T14 are extended from the S12T14 datasets by adding 10 new suppliers. Similarly, the S50T21 suppliers are extended from the S20T21 locations and generating randomly the locations of an additional 30 suppliers. Datasets S98T14 is based on a real life data and the suppliers are closely located.

All the datasets with the exception of S50T21, have demands in every period. Some suppliers in the S50T21 dataset may not receive the demand for their product until the later periods. As some of the datasets are extracted from S50T21, it is possible that the demand for some of the product is zero. Datasets S50T5 and S50T10 consist of products with zero demand. The suppliers of these products can be effectively eliminated from the representation as they will never be visited in the planning horizon.

We note that the demands for S98T14 are given in real values and the amount varies significantly between each product. The cost per unit distance, fixed cost and the vehicles' capacity are increased to 50, 200 and 400 respectively. It is also noted that the demand for each product for this data set is constant as this is a common feature in the automotive industry. In our experiment, the number of generations, crossover rate and mutation rate are fixed at 300, 0.9, and 0.01 respectively for all the problems. The population size is fixed at 200 individuals. Each data set is executed 10 times.

Table 4.1 shows the characteristics of the four original datasets.

Table 4.1: Characteristics of the datasets.

Dataset	S12T14	S20T21	S50T21	S98T14
Fixed Cost (F)	20	20	20	200
Cost per unit Travelling Distance	1	1	1	50
Vehicle Capacity	10	10	10	400
Maximum Route Length	140	140	140	150
Range of Holding Costs	[3,27]	[3,27]	[1,9]	[1,44]
Range of Demand for Each Product	[1,4]	[1,4]	[0,9]	[0.04,393.33]
Coordinate of Depot	(0,0)	(0,0)	(0,0)	(40,-80)
Coordinate of Assembly Plant	(10,20)	(10,20)	(10,20)	(42.31,-83.17)

Figure 4.1(a)-(d) show the distribution of suppliers or customers in dataset S12T14, S20T21, S50T21, S98T14 respectively. From Figure 4.1(a)-(c), the distribution of suppliers/customers in dataset S12T14, S20T21 and S50T21 are fairly dispersed. It is because these datasets are generated randomly and dataset S20T21 and S50T21 are extended from S12T14. However, dataset S98T14 in Figure 4.1(d) are based on a real life data and therefore the distribution of the suppliers/customers are located closely with each other.

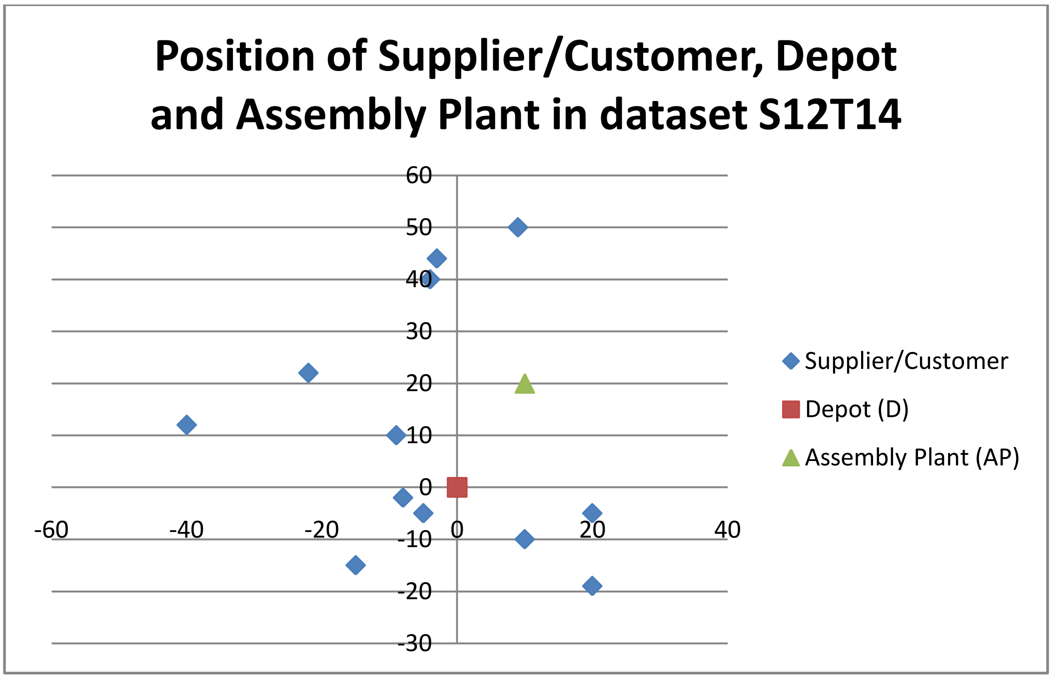

Figure 4.1 (a): The distribution of customers/suppliers in dataset S12T14

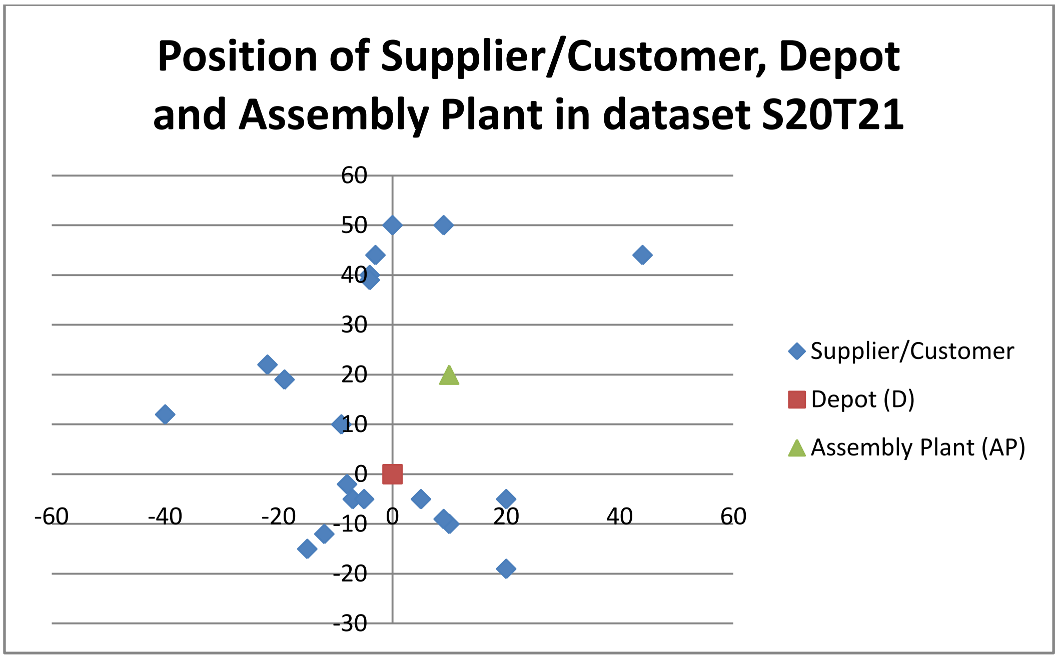

Figure 4.1 (b): The distribution of customers/suppliers in dataset S20T21

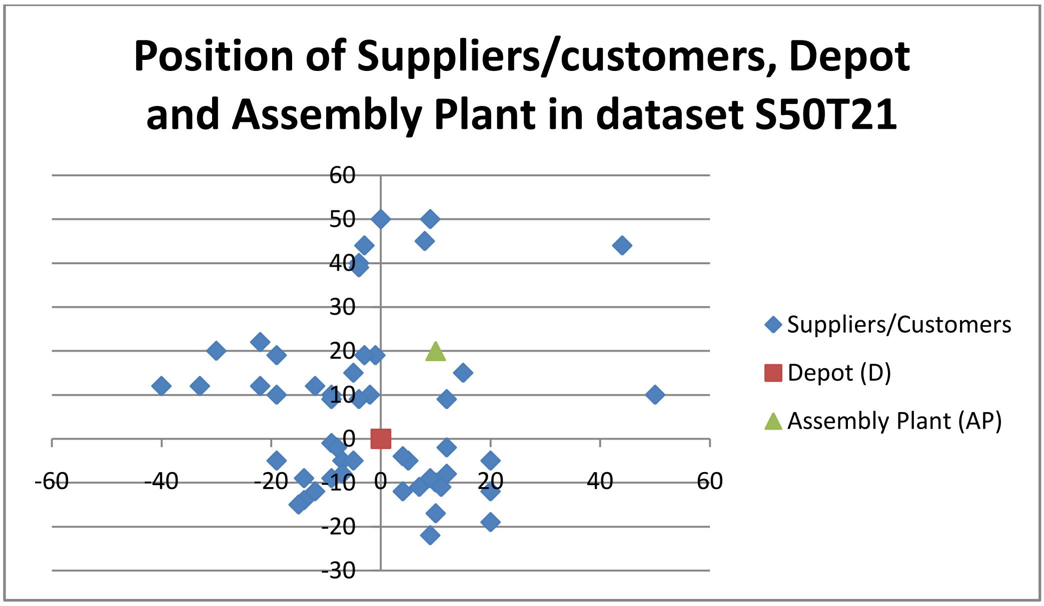

Figure 4.1 (c): The distribution of customers/suppliers in dataset S50T21

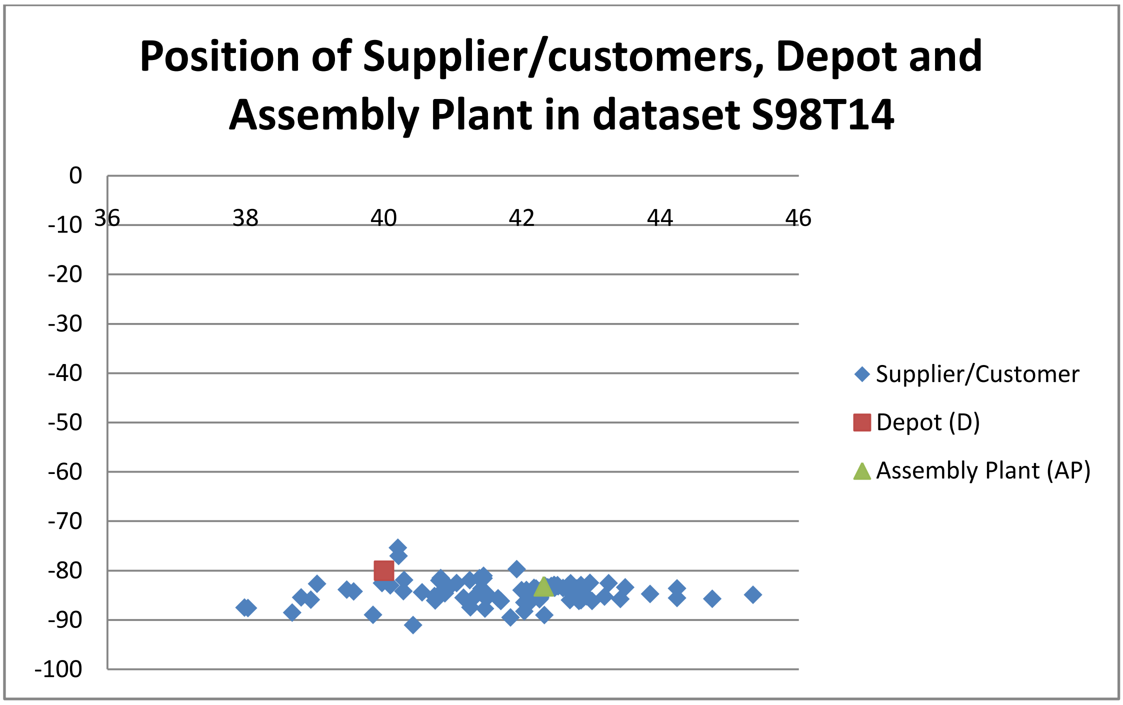

Figure 4.1(d): The distribution of customers/suppliers in dataset S98T14.

4.4 Hybrid Genetic Algorithms (HGA)

The first approach in this study is Hybrid Genetic Algorithms (HGA). In this method, the genetic representation is in binary form. This method uses the single point crossover and uniform at random bit flip mutation.

4.4.1 Binary Matrix Representation

In this type of representation, a binary matrix of size $N \times (T-1)$ where N is the number of customers while T defines the number of periods. A 1 at position (i,j) in the chromosome indicates that customer i will be visited at period j. The amount to be delivered depends on whether there will be delivery in the subsequent period or not. Since backordering is not allowed, the total delivery from customer i in period j is $\sum_{l-j}^{k-1} d_{il}$, the sum of all the demands in period $j, j+1, \ldots, k-1$ where the next delivery will be made in period k. As the initial inventory, s_{i0} for $i = 1,2, \ldots, N$ is assumed to be zero, the values in the first column consist of all ones, thus ignored from the representation. However, the algorithm can be adjusted accordingly if the initial inventory at customer i is given.

Table 4.2 : Binary chromosome representation for 5 suppliers in 5 periods.

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	1	0	0	1	0
2	1	1	1	0	0
3	1	1	0	0	1
4	1	0	1	1	0
5	1	0	1	1	1

4.4.2 Crossover operator

In this approach, the crossover operator that has been employed is a two-dimensional uniform crossover that is modified to suit the matrix representation. A binary mask of size $(N \times T)$ is generated randomly for each pair of parents. The position of ones in the binary mask determines the values in the first parent that are transferred to the first offspring and the elements in the position zeros are obtained from the second parent. A complimentary mask is used to deduce the second offspring. Figure 4.2 below shows the illustration of the modified uniform crossover.

Supplier	Period				
	1	2	3	4	5
1	1	0	1	1	0
2	1	0	0	1	1
3	1	1	0	1	0
4	1	1	1	0	0
5	1	0	1	0	1

Parent 1

Supplier	Period				
	1	2	3	4	5
1	1	1	1	0	1
2	1	0	1	1	0
3	1	1	0	1	0
4	1	0	1	0	0
5	1	0	0	1	1

Parent 2

Supplier	Period				
	1	2	3	4	5
1	1	0	1	0	1
2	0	1	0	1	0
3	1	0	0	0	1
4	0	1	0	1	0
5	0	0	0	1	1

Crossover Mask

Supplier	Period				
	1	2	3	4	5
1	1	1	1	0	0
2	1	0	1	1	0
3	1	1	0	1	0
4	1	1	1	0	0
5	1	0	0	0	1

Child 1

Supplier	Period				
	1	2	3	4	5
1	1	0	1	1	1
2	1	0	0	1	1
3	1	1	0	1	0
4	1	0	1	0	0
5	1	0	1	1	1

Child 2

Figure 4.2 : The illustration of uniform crossover.

4.4.3 Mutation operator

Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. In the flip bit mutation operator, which is adopted in this study, the selected bit in the chromosomes will be inverted. For example, if the genome bit is 1, then it will be mutated to 0 and vice versa. Normally this bit will be chosen randomly. The mutation rate has been set as 0.01 which represent the frequency of a mutation to occur in a generation. By taking child 1 from Figure 4.2, Figure 4.3 shows the example of bit flip mutation operator process.

	Child 1				
Before mutation	0	0	1	1	1
After mutation	1	0	0	0	1

Figure 4.3: Example of bit flip mutation operator.

4.4.4 Overall Hybrid Genetic Algorithms

The following algorithm shows the overall hybrid Genetic Algorithm (HGA).

STEP 1: Generate an initial population.

STEP 2: Decode the chromosome according to the representation procedure previously described. Perform STEP 3.

STEP 3a: {Double Sweep Algorithm} For each period $j, j = 1,2, \ldots, T$, arrange the suppliers/customers $(s_1, s_2, \ldots, s_m)$ and the assembly plant (s_{m+1}) around the depot such that the y-coordinate of the assembly plant is the same as the y-coordinate of the depot. Sort the suppliers/ customers $(s_1, s_2, \ldots, s_m)$ in ascending order according to their new y-coordinate values. Let $s_{(i)}$ be the ith supplier after the sort. Set $i = 1$ and $k = 1$. Open a route $R_k = \{ \quad \}$ and set $Q_k = 0$, where Q_k is the total pick-up/delivery quantity assigned to cluster k.

STEP 3b: {Clustering} If $Q_k + d_{ij} \leq C$, assign $s_{(i)}$ to route R_k. Set $Q_k = Q_k + d_{ij}$ and $a_{ik} = d_{ij}$. Otherwise set $a_{ik} = C - Q_k$ and $Q_k = C$. Set $k = k + 1$ and open a new route $R_k = \{ \quad \}$. Assign $s_{(i)}$ to route R_k. Set $a_{ik} = d_{ij} - a_{ik-1}$ and $Q_k = a_{ik}$. If $i > m$, set $k = 1$ and go to STEP 3c. Otherwise, set $i = i + 1$ and repeat STEP 3b.

STEP 3c:{Routing} Sort the suppliers within route R_k according to their x-coordinate values in ascending order. Let $s_{(i)}^k$ be the ith supplier in route R_k after the sort. Form a route that starts at the depot s_0, visit suppliers/customers $s_{(1)}^k, s_{(2)}^k, \ldots, s_{(|R_k|)}^k, s_{m+1}$ and returns to s_0. Evaluate the total objective function value as given in Equation 2.1 in Section 2.2.

STEP 4: Perform crossover and mutation.

STEP 5: Repeat STEP 2- STEP 3 until the maximum number of generations is attained.

4.4.5 Results and Discussion

The algorithms were written in C++ using Genetic Algorithms Library (GALIB) to run the program. 14 datasets as described in Section 4.3 were used to run this program. In our experiment, the number of generations, crossover rate and mutation rate are fixed at 300, 0.9 and 0.01 respectively for all the problems. The population size is fixed at 200 individuals. Each dataset is executed ten times. Table 4.3 summarizes the best total objective, the mean and standard deviation of total objective for 10 runs, and the number of vehicle for the best total objective. Although the same datasets are used by Lee et. al [1], their results are incomparable to our results as their work has never been published and some of their results violate the route length constraint.

Table 4.3: The best results, mean, standard deviation and number of vehicles for each datasets over 10 runs with HGA.

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S12T5				
Best Objective	2575.3	2813.6	3297.1	3921.9
Mean	2627.11	2874.05	3676.7	3980.35
Std. Deviation	36.04021	33.75241	35.7246	35.52959
No. of Vehicle	14	15	20	19
S12T10				
Best Objective	3028.1	5138.7	4021.4	4017.4
Mean	3028.1	5254.36	4133.45	4133.41
Std. Deviation	0	69.75472	46.66308	79.03118
No. of Vehicle	14	31	31	31
S12T14				
Best Objective	6347	6453.6	6864.6	6818.2
Mean	6500.83	6524.57	6970.48	6961.63
Std. Deviation	79.5637	62.04733	74.04554	77.56649
No. of Vehicle	41	42	44	45
S20T5				
Best Objective	4030.3	4019.9	6457.8	6468.6
Mean	4110.39	4130.33	6617.86	7843.79
Std. Deviation	67.82293	64.64048	99.57333	1312.784
No. of Vehicle	26	24	34	32

Table 4.3 (cont.)

Dataset	Inbound Logistics		Outbound Logistics	
	Split Delivery	Un-split Delivery	Split Pick-up	Un-split Pick-up
S20T10				
Best Objective	5410.8	9530.5	13259	13400.9
Mean	5477.83	9760.74	13659.45	16241.28
Std. Deviation	58.17915	139.1531	236.7183	2935.066
No. of Vehicle	23	53	62	63
S20T14				
Best Objective	11035.5	10921	13155	13046
Mean	11803.58	11339.43	13417.31	13000.6
Std. Deviation	1849.737	267.8344	209.3352	1152.021
No. of Vehicle	73	70	79	78
S20T21				
Best Objective	15076	15254	15709	16008
Mean	15225.9	15572.9	16085.2	20823
Std. Deviation	128.0117	167.2951	424.2025	4945.593
No. of Vehicle	105	104	100	104
S50T5				
Best Objective	5461.4	5729.6	9841	9455.1
Mean	5567.98	5808.87	9971.03	9759.14
Std. Deviation	69.71548	57.28983	136.9567	232.7855
No. of Vehicle	46	47	60	59
S50T10				
Best Objective	11817	12081.1	16891	16763
Mean	11921.37	12342.23	17138.4	17045.7
Std. Deviation	93.08041	156.4575	126.3762	205.9283
No. of Vehicle	101	100	110	112
S50T14				
Best Objective	16936	17521	18082	17476
Mean	17216.6	17727.9	18638.5	17994.8
Std. Deviation	196.1576	127.3529	486.7275	297.7276
No. of Vehicle	141	143	142	147
S50T21				
Best Objective	25283	27189	34165	33251
Mean	26518.3	27415.6	34505.3	33508.3
Std. Deviation	641.0322	160.1653	207.5037	228.3054
No. of Vehicle	223	220	228	239
S98T5				
Best Objective	40720	41832.7	411865	304810
Mean	43802.46	45655.06	418222.1	406942
Std. Deviation	2338.858	3314.367	6311.975	36688.76
No. of Vehicle	57	66	34	34

Table 4.3 (cont.)

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S98T10				
Best Objective	83419.1	87522.5	1219360	1203650
Mean	87471.27	93132.9	1239270	1227920
Std. Deviation	3741.721	5172.044	13690.67	12187.09
No. of Vehicle	115	133	102	99
S98T14				
Best Objective	119450	125951	1994890	1968740
Mean	124529.8	131308.7	2030847	2011311
Std. Deviation	3562.157	4731.59	31193.15	30712.89
No. of Vehicle	154	190	164	162

In general, it is found that the inbound logistics produces better results compared to the outbound logistics. It can also be seen that most split delivery/pick-up problems give lower total objective cost than un-split problems. This is already expected as in the un-split problems, the transportation cost will increase due to the additional number of vehicle.

From Table 4.3, it can be seen that the standard deviation increases consistently with the size of the data. The solutions of the small instances are fairly distributed within these 10 runs. However, for the large instances, the standard deviations are comparatively large which can be due to the maximum number of generations being not sufficiently large. As mentioned earlier, the maximum number of generation is set to be 300. For small instances, this number of generation is sufficient to get the function converge to the total objective cost. For the large instances, on the other hands, even after the maximum number of generation, the function is still converging to the objective value.

Table 4.4 tabulates the characteristics of the best results obtained in Table 4.3. It gives the total distance cost, inventory costs, the number of vehicles involved, total objective costs and the CPU time in milliseconds. From the table, we can see that generally, most split delivery problem results gives slightly better results compared to the un-split delivery. However, the difference between these two types of inventory can be considered as small to be noticed. The outbound logistics also gives slightly larger results compared with the inbound logistics. A significant difference of the two types of logistics is shown clearly in the large dataset of S98T5, S98T10 and S98T14. This difference on the Inventory Holding Costs and the Distance costs has the major effect on the total objective costs.

Table 4.4: The characteristics of the best results given in Table 4.3

	Inbound Logistics		**Outbound Logistics**	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S12T5				
Distance Cost	2058.3	1748.6	1854.1	2497.9
Inventory Cost	237	765	1043	1044
Num. of Veh.	14	15	20	19
Total Cost	2575.3	2813.6	3297.1	3921.9
CPU Time	754	1033	836	691
S12T10				
Distance Cost	2748.1	3852.7	3062.4	3010.4
Inventory Cost	0	666	339	387
Num. of Veh.	14	31	31	31
Total Cost	3028.1	5138.7	4021.4	4017.4
CPU Time	858	1422	1087	1064
S12T14				
Distance Cost	4981	4932.6	5234.6	5052.2
Inventory Cost	546	681	750	846
Num. of Veh.	41	42	44	46
Total Cost	6347	6453.6	6864.6	6818.2
CPU Time	1317	2039	879	1331

Table 4.4 (cont.)

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S20T5				
Distance Cost	2502.3	2543.9	3860.8	3797.6
Inventory Cost	1008	996	1917	2031
Num. of Veh.	26	24	34	32
Total Cost	4030.3	4019.9	6457.8	6468.6
CPU Time	1121	977	1691	1050
S20T10				
Distance Cost	4824.8	6181.5	7972.1	8063.9
Inventory Cost	126	2289	4047	4077
Num. of Veh.	23	53	62	63
Total Cost	5410.8	9530.5	13259	13400.9
CPU Time	1694	1960	1825	2242
S20T14				
Distance Cost	7502.5	7733.4	8815.4	8558
Inventory Cost	2073	1788	2760	2928
Num. of Veh.	73	70	79	78
Total Cost	11035.5	10921	13155	13046
CPU Time	2313	3050	2168	3538
S20T21				
Distance Cost	10675	11239	11228	11144
Inventory Cost	2301	1935	2481	2784
Num. of Veh.	105	104	100	104
Total Cost	15076	15254	15709	16008
CPU Time	3468	3651	3828	3613
S50T5				
Distance Cost	4220.4	4336.6	7367	7008.1
Inventory Cost	321	453	1274	1267
Num. of Veh.	46	47	60	59
Total Cost	5461.4	5729.6	9841	9455.1
CPU Time	3269	2559	2690	2365
S50T10				
Distance Cost	8574	8868.1	12691	12163
Inventory Cost	1223	1213	2000	2360
Num. of Veh.	101	100	110	112
Total Cost	11817	12081.1	16891	16763
CPU Time	5075	3630	2502	4951
S50T14				
Distance Cost	12074	12727	12644	11528
Inventory Cost	2042	1934	2598	3007
Num. of Veh.	141	143	142	147
Total Cost	16936	17521	18082	17476
CPU Time	7839	7405	3170	6563

Table 4.4 (cont.)

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S50T21				
Distance Cost	17496	19508	24349	23679
Inventory Cost	3327	3281	5256	4792
Num. of Veh.	223	220	228	239
Total Cost	25283	27189	34165	33251
CPU Time	9104	8708	3743	6875
S98T5				
Distance Cost	514.56	476.35	5674.1	5676
Inventory Cost	3592	4815.2	121360	14210
Num. of Veh.	57	66	34	34
Total Cost	40720	41832.7	411865	304810
CPU Time	9890	9613	12500	8673
S98T10				
Distance Cost	1034	971.57	17585	17020
Inventory Cost	8719.1	12344	319710	332850
Num. of Veh.	115	133	102	99
Total Cost	83419.1	87522.5	1219360	1203650
CPU Time	13892	13505	10053	11110
S98T14				
Distance Cost	1453.4	1388	28549	28170
Inventory Cost	15980	18551	534640	527840
Num. of Veh.	154	190	164	162
Total Cost	119450	125951	1994890	1968740
CPU Time	17399	18131	15007	9066

4.5 Knowledge-based Hybrid Genetic Algorithms (KBHGA)

The second method studied in this research is Knowledge-based Hybrid Genetic Algorithms. Unlike HGA, the chromosomes in this approach are represented with the real-valued integer which denotes the inventory collection or delivery. Therefore, in this approach, new crossover operator and mutation operator are introduced to handle the real-valued chromosomes.

4.5.1 Real-valued Matrix Representation

The chromosomes in this representation encode the delivery matrices (the amount to be collected/delivered) in the form of a 2-dimensional NxT matrix. The initial real-valued chromosome is constructed through a procedure known as preprocessing. This procedure will use a combination of a random binary representation and the demand matrix.

In this procedure, a binary matrix of size NxT where the elements in the first column are all ones is randomly generated. The amount to be delivered to supplier i in period j is generated randomly in the interval of $\left(\sum_{l=j}^{k-1} d_{il}, \sum_{l=j}^{k} d_{il}\right)$ where d_{il} is the demand at supplier i in period l and the next delivery to supplier i is in period k. This is to allow the flexibility of satisfying parts of the demands in a given period if the transportation cost is reduced. As mentioned earlier, there will always be delivery in the first period for all suppliers as the initial inventory, s_{i0} for $i = 1,2,\ldots,N$ is assumed to be zero.

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	4	2	4	4	4
2	2	2	2	2	2
3	2	1	2	2	2
4	4	1	4	4	4
5	2	1	2	2	2

(a) The demand matrix

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	1	0	0	1	0
2	1	1	1	0	0
3	1	1	0	0	1
4	1	0	1	1	0
5	1	0	1	1	1

(b) The binary matrix

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	11	0	0	7	0
2	2	2	6	0	0
3	2	5	0	0	1
4	7	0	2	8	0
5	4	0	1	2	2

(c) The real valued chromosome representation

Figure 4.4 : Preprocessing steps to produce initial real-valued chromosome representation

Figure 4.4 gives an illustration of the construction of a real-valued chromosome through the preprocessing steps. Figure 4.4(a) shows the binary chromosome representation while Figure 4.4(b) is the demand matrix. After the preprocessing steps, the resultant collection/delivery matrix becomes the initial real-valued chromosome as showed in Figure 4.4(c).

4.5.2 Crossover Operator

Since this approach is using the real-valued chromosome, a new crossover operator is proposed for this method. It is based on exchanging the delivery schedules for a selected set of periods, which is chosen randomly between the two parents. At the same time, it will ensure that the resultant child does not violate either the demand or the vehicle's capacity constraints. This crossover operator is similar to the one used by

AbdelMaguid and Dessouky [25] on a different problem. In their study, they allowed the infeasible solutions and penalized it in their objective function.

In our model, all constraints are treated as hard constraints, thus to restore feasibility a repair mechanism must be designed. However, for this particular model, the repair mechanism is found to be too time consuming and costly.

Firstly, a mask of size *N*x1 is randomly generated. The position of the ones in the binary mask determines the values in the first parent that are transferred to the first offspring and the elements in the position zeros are obtained from the second parent.

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	11	0	0	7	0
2	2	2	6	0	0
3	2	5	0	0	1
4	7	0	2	8	0
5	4	0	1	2	2

Parent 1

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	4	9	0	5	0
2	5	0	1	4	0
3	8	0	0	0	1
4	7	0	10	0	0
5	2	3	0	2	2

Parent 2

Mask =[1 0 0 1 0]

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	11	0	0	7	0
2	5	0	1	4	0
3	8	0	0	0	1
4	7	0	2	8	0
5	2	3	0	2	2

Child 1

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	4	9	0	5	0
2	2	2	6	0	0
3	2	5	0	0	1
4	7	0	10	0	0
5	4	0	1	2	2

Child 2

Figure 4.5: The crossover operator

Figure 4.5 illustrates the crossover operator by using mask. From the figure, the first element from the mask vector (that is 1) indicates that the first row for Child 1 will be copied from the first row of Parent 1. Meanwhile, for Child 2, the first row will be taken from Parent 2. The second element in the mask vector (that is 0) means that the second row for Child 1 will be taken from the second row of Parent 2 and for Child 2, the second row will be taken from Parent 1. The rest of the child will be constructed based on these mask vector.

4.5.3 Mutation Operator

From the observation, a slightly higher inventory holding costs are produced from this type of representation. Hence, a new mutation operator has been designed to overcome this problem. This mutation operator transfer some amount of the product picked up/delivered in the previous period to the current selected period. If the selected period happens to be the first period, then the amount will transferred from the selected period to the succeeding period. By doing this, the inventory holding cost can be reduced. The algorithm for the mutation process is given below:

STEP 1: Select randomly the gene that will undergo the mutation process. Let this gene be $gene(i,j)$ where i and j denote supplier/customer and period respectively.

STEP 2: If $j \neq 1$, go to STEP 3. Otherwise, set $q = 1$ where q is the number of periods before the next collection/delivery. If $gene(i, j+q) = 0$ set $q = q+1$ and repeat the process until $gene(i, j+q) \neq 0$. Next, let $r = \sum_{k=1}^{j+q} a_{ik} - \sum_{k=1}^{j+q} d_{ik}$ where a_{ij} and d_{ij} is the collected amount and the demand in period j respectively. Generate the amount to be transferred (say V) randomly in the interval $(0, r)$. Let $gene(i, j) = gene(i, j) - V$ and $gene(i, j + q) = gene(i, j + q) + V$.

STEP 3: Set $p = 1$ where p is the number of periods since the last collection. If $gene(i, j - p) = 0$, set $p = p + 1$. Repeat the process until $gene(i, j - p) \neq 0$. Let $r = \sum_{k=j-p}^{j} a_{ik} - \sum_{k=j-p}^{j} d_{ik}$ and generate randomly $V \in (0, r)$. Set $gene(i, j - p) = gene(i, j - p) - V$ and $gene(i, j) = gene(i, j) + V$.

The illustration for the mutation operator is given in Figure 4.6 below.

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	11	0	0	7	0

1. Randomly select the gene to be mutated i.e. $gene(1,1) = 11$
2. Determine when next delivery/collection will occur i.e. $gene(1,4) = 7$.
3. Find $r = \sum_{k=1}^{j+q} a_{ik} - \sum_{k=1}^{j+q} d_{ik}$

$$\sum_{j=1}^{4} a_{1k} = 11 + 0 + 0 + 7 = 18$$

   Based on the demand matrix in Figure 4.5a), calculate the demand

$$\sum_{k=1}^{4} d_{1k} = 4 + 2 + 4 + 4 = 14$$

   There $r = 18 - 14 = 4$

4. Generate the random amount to be transferred

$$V = rand(0,4) = 3$$

5. Mutate $gene(1,1)$ and $gene(1,4)$

$$gene(1,1) = 11 - 3 = 8$$
$$gene(1,4) = 7 + 3 = 10$$

6. The new matrix after mutation is given as follows:

	Period				
Supplier	**1**	**2**	**3**	**4**	**5**
1	8	0	0	10	0

Figure 4.6: Illustration of mutation operator for real-valued representation

4.5.4 Results and discussion

Just like in the Hybrid Genetic Algorithms method, the algorithms for Knowledge-based Genetic Algorithms were also written in C++ using Genetic Algorithms Library (GALIB) to run the program. We used the same datasets and parameter to run these algorithms where each dataset is executed ten times. Table 4.5 below summarizes the best total objective, the mean and standard deviation of total objective for 10 runs, and the number of vehicle for the best total objective.

Table 4.5: The best results, mean, standard deviation and number of vehicles for each datasets over 10 runs with KBGA.

	Inbound Logistics		**Outbound Logistics**	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S12T5				
Best Obj.	2773.1	2979	2933.1	3986.3
Mean	2851.58	3138.55	3129.46	4209.33
Std. Dev.	67.85541	83.40893	125.76911	105.629
No. of Veh.	16	15	15	20
S12T10				
Best Obj.	5104	5063.8	5154.2	4165.6
Mean	5190.92	5229.08	5247.7	4246.51
Std. Dev.	66.96616	93.41276	88.979689	70.74679
No. of Veh.	29	30	30	32
S12T14				
Best Obj.	6498.6	6229.9	6455.4	7068.6
Mean	6644.1	6397.91	9670.34	7246.82
Std. Dev.	124.066	151.3162	7088.3253	147.5165
No. of Veh.	43	39	40	45
S20T5				
Best Obj.	4235.2	4055.2	4164.7	6600.2
Mean	4363.62	4195.97	4248.36	6664.1
Std. Dev.	94.45603	82.56646	94.322468	52.35008
No. of Veh.	22	27	26	34

Table 4.5 (cont.)

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S20T10				
Best Obj.	9816.7	5063.8	9576.1	13981
Mean	9945.18	7459.17	9793.38	14200.9
Std. Dev.	88.83567	2380.817	148.76595	124.8131
No. of Veh.	56	30	58	67
S20T14				
Best Obj.	11450	11520	11513	13711
Mean	11603.8	11796.92	11663.2	14083.4
Std. Dev.	99.8609	195.829	102.51683	217.817
No. of Veh.	79	77	74	79
S20T21				
Best Obj.	15561	15666	15579	16419
Mean	15950	16020.46	15916.4	16644.3
Std. Dev.	313.0343	228.8699	248.27565	149.5222
No. of Veh.	101	107	103	108
S50T5				
Best Obj.	6354	5698.9	9847	9677
Mean	6406.2	6008.93	10161	10265.57
Std. Dev.	38.38229	233.2013	250.42863	657.5434
No. of Veh.	45	47	61	59
S50T10				
Best Obj.	17829	17965	17813	17346
Mean	18292.6	18895.3	18303.6	17759.4
Std. Dev.	328.2214	2020.988	319.90905	166.1099
No. of Veh.	116	117	116	110
S50T14				
Best Obj.	24162	23936	18420	17963
Mean	24626	24825.9	22188.2	18423.4
Std. Dev.	428.184	365.9971	3374.9366	237.0819
No. of Veh.	158	158	151	151
S50T21				
Best Obj.	34108	33575	33833	36627
Mean	34240.4	33866.3	34420.8	36890.3
Std. Dev.	193.908	233.8727	655.21844	217.5424
No. of Veh.	233	241	236	252
S98T5				
Best Obj.	69021	515992	495820	406280
Mean	69333.2	528572.7	510834	417299
Std. Dev.	2395.145	9056.137	10537.264	5163.751
No. of Veh.	58	74	72	68

Table 4.5 (cont.)

	Inbound Logistics		Outbound Logistics	
Dataset	**Split Delivery**	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S98T10				
Best Obj.	155460	1009560	998840	834040
Mean	157386	1033530	1030248	850397
Std. Dev.	1316.598	21927.84	21007.725	10082.4
No. of Veh.	116	140	137	128
S98T14				
Best Obj.	227700	1478000	1202800	1172200
Mean	230232	1539854	1396820	1199350
Std. Dev.	1713.029	120452.3	109103.97	14486.72
No. of Veh.	160	201	180	177

Generally in the Inbound Logistics, most un-split delivery problems seems to have lower total costs compared to split delivery problems. However, this situation seems to change when the size of the instances increases for example in dataset S98T5, S98T10 and S98T14, there are very large difference in the amount of total costs between split and un-split delivery problems.

A different situation occurs in the outbound logistics where lower results in small and medium instances are located within the split pick-up problems and when the size of the instances increases, lower total costs are located in the un-split pick-up problems.

An observation on the standard deviation shows that there is no consistency in the results spread. Therefore, by comparing the KBGA and HGA methods, we can say that the binary representation gives better and steady solutions compared to the real-valued integer representation.

Table 4.6 below shows the characteristics of the best results given in Table 4.4. It gives the distance costs, total inventory holding costs, number of vehicle and the best total objective costs for 10 runs.

Table 4.6: The characteristics of the best results given in Table 4.5

Dataset	Inbound Logistics		Outbound Logistics	
	Split Delivery	**Un-split Delivery**	**Split Pick-up**	**Un-split Pick-up**
S12T5				
Distance Cost	1826.1	1748.6	1790.1	1782
Inventory Cost	627	765	843	897
Num. of Veh.	16	15	15	15
Total Cost	2773.1	2813.6	2933.1	2979
CPU Time	62	1033	56	72
S12T10				
Distance Cost	3435	3852.7	3720.2	3566.8
Inventory Cost	1089	666	834	897
Num. of Veh.	29	31	30	30
Total Cost	5104	5138.7	5154.2	5063.8
CPU Time	463	1422	501	516
S12T14				
Distance Cost	4825.6	4932.6	4797.4	4609.9
Inventory Cost	813	681	858	840
Num. of Veh.	43	42	40	39
Total Cost	6498.6	6453.6	6455.4	6229.9
CPU Time	525	2039	1072	689
S20T5				
Distance Cost	2736.2	2543.9	2669.7	2534.2
Inventory Cost	1059	996	975	981
Num. of Veh.	22	24	26	27
Total Cost	4235.2	4019.9	4164.7	4055.2
CPU Time	335	977	416	398
S20T10				
Distance Cost	6422.7	6181.5	6628.1	3566.8
Inventory Cost	2274	2289	1788	897
Num. of Veh.	56	53	58	30
Total Cost	9816.7	9530.5	9576.1	5063.8
CPU Time	802	1960	1415	516

Table 4.6 (cont.)

Dataset	Inbound Logistics		Outbound Logistics	
	Split Delivery	Un-split Delivery	Split Pick-up	Un-split Pick-up
S20T14				
Distance Cost	8018.7	7733.4	8175.6	7993.9
Inventory Cost	1851	1788	1857	1986
Num. of Veh.	79	70	74	77
Total Cost	11450	10921	11513	11520
CPU Time	2437	3050	1816	2732
S20T21				
Distance Cost	11213	11239	11539	11678
Inventory Cost	2328	1935	1980	1848
Num. of Veh.	101	104	103	107
Total Cost	15561	15254	15579	15666
CPU Time	2269	3651	2246	3125
S50T5				
Distance Cost	4803	4336.6	6994	4163.9
Inventory Cost	651	453	1633	595
Num. of Veh.	45	47	61	47
Total Cost	6354	5729.6	9847	5698.9
CPU Time	2104	2559	4293	3842
S50T10				
Distance Cost	13140	8868.1	12903	13291
Inventory Cost	2369	1213	2590	2334
Num. of Veh.	116	100	116	117
Total Cost	17829	12081.1	17813	17965
CPU Time	5184	3630	3256	6435
S50T14				
Distance Cost	17490	12727	12866	17436
Inventory Cost	3512	1934	2534	3340
Num. of Veh.	158	143	151	158
Total Cost	24162	17521	18420	23936
CPU Time	5566	7405	4489	3905
S50T21				
Distance Cost	24242	19508	24561	23654
Inventory Cost	5206	3281	4552	5101
Num. of Veh.	233	220	236	241
Total Cost	34108	27189	33833	33575
CPU Time	6415	8708	9634	5057
S98T5				
Distance Cost	878.64	476.35	8899.1	9138.9
Inventory Cost	13489	4815.2	36464	44247
Num. of Veh.	58	66	72	74
Total Cost	69021	41832.7	495820	515992
CPU Time	10424	9613	13155	13764

Table 4.6 (cont.)

Dataset	Inbound Logistics		Outbound Logistics	
	Split Delivery	Un-split Delivery	Split Pick-up	Un-split Pick-up
S98T10				
Distance Cost	2139.8	971.57	17248	17136
Inventory Cost	25274	12344	109050	124760
Num. of Veh.	116	133	137	140
Total Cost	155460	87522.5	998840	1009560
CPU Time	15429	13505	31978	12713
S98T14				
Distance Cost	3111.1	1388	20183	25251
Inventory Cost	40149	18551	157700	175250
Num. of Veh.	160	190	180	201
Total Cost	227700	125951	1202800	1478000
CPU Time	19622	18131	17374	20132

In the inbound logistics, the distance costs for split delivery problem is higher than un-split delivery problem. Considerably large amounts of difference between the distance costs of split delivery problem and un-split delivery problem are noticed especially in large instances such as S98T5, S98T10 and S98T14. The same case did not occur in outbound logistics where the distance costs between split and un-split pick-up problem for this type of logistics did not vary much.

These results however, cannot be compared with Lee et. al [1] because in their results, some of the routes violate the route length constraints.

4.6 Conclusion

Two methods, namely Hybrid Genetic Algorithms (HGA) and Knowledge-based Genetic Algorithms (KbGA) are introduced in this chapter. In this study, Double Sweep Algorithms (DSA) has been used to arrange the customers before and after clustering. There is also description about the dataset used in this study. Originally, there are 4 datasets. Based on these datasets, 11 more are created to vary the number of periods to represent small, medium, and large size problem. The characteristics of datasets are also described in this chapter. The coordinates for dataset with 12 suppliers, 20 suppliers, 50 suppliers and 98 suppliers are plotted to show the position of the suppliers with the Depot and the Assembly Plant.

In HGA, the delivery matrix is represented in the form of binary matrix where 0 indicates that there is no delivery and 1 indicates otherwise. The crossover and mutation operator used in this method is the default operator in Genetic Algorithm that is single point crossover and flip bit mutation.

In KBGA, real-valued matrix is used to represent the delivery matrix instead of binary matrix. New crossover operator is designed for this method using a mask vector. Since the results show slightly higher inventory holding costs, a new mutation operator is designed to transfer some amount of the product delivery in previous period to current selected period.

CHAPTER 5

MODIFIED HYBRID GENETIC ALGORITHMS (MHGA)

5.0 Introduction

From Chapter 4, it is observed that the last vehicle in each period normally utilizes less than half of the vehicle's capacity. This increases the number of vehicles used unnecessarily and indirectly increases the transportation costs. Therefore in this chapter, we propose a new formulation in order to maximize the vehicle's utilization. Since Hybrid Genetic Algorithms (HGA) seems to outperform Knowledge-based Genetic Algorithms (KBGA) in Chapter 4, then the modification is done on the HGA problems and is referred to as the Modified Hybrid Genetic Algorithms (MHGA).

In the second part of this chapter, some post-optimization is done to do the routing within each cluster by using 2-opt that is originally proposed by Croes [43]. The results for the un-split delivery case are then compared with the results from Variable Neighborhood Search algorithm.

Lastly, CPLEX is used to find the lower bound for each data set and the results for the split delivery problem are compared with the best integer from the CPLEX.

5.1 Modified Hybrid Genetic Algorithms (MHGA)

The main purpose of MHGA is to maximize the vehicle's utilization by improving the coordination of transportation. This is achieved by examining the total amount to be collected by the last vehicle in each period. However, this will consequently result in having to construct the route again and also very costly in a GA platform.

We propose that the excess collection in each period (with respect to the vehicle capacity) is approximated instead. If the excess is less than $K\%$, then the amount of excess is transferred to the preceding period. However, certain limit of transfer period must be set to ensure that the inventory holding costs is not be too high. In our study, the transfer period is limited to 2 periods only.

By referring to the overall Hybrid Genetic Algorithms (HGA) in Section 4.4.4, the modification of MHGA is done in Step 3A-3C as shown as follows:

Step 1: Generate an initial population.

Step 2: For each supplier $i, i = 1, 2, 3, \dots, N$, construct a total collection/delivery matrix, $D_{ij} = \sum_{j=1}^{p-1} d_{ij}$ where d_{ij} is the demand for supplier i in period j and p is the period for the next visit to supplier i.

Step 3A: {MHGA}:For each period $j, j = 1, 2, \dots, T$, let $DP_j = \sum_{i=1}^{|P_j|} D_{ij}$ where $P_j \subseteq N$ is the set of suppliers visited in period j and DP_j is the total collection in period j. Calculate the delivery/collection excess (with respect to vehicle capacity) in each period as, $R_j = rem(\mathrm{DP_j}, \mathrm{C})$ where C is the vehicle capacity.

Step 3B: Starting from $j = T$ (the last period), if $R_j < K$, sort in ascending order of the holding costs, the supplier $\{s_1, s_2, s_3, \dots, s_m\} \in P_j$ and let $s_{\{k\}}$ be the kth supplier after the sort. Otherwise set $j = j - 1$ and repeat Step 3B.

Step 3C: Set $q = 1, sum = 0$ and $s = 1$ where q ($q \leq 2$) is the number of periods since the last collection/delivery, sum is the total amount to be transferred to the preceding period(s) and $s(s \leq |P_j|)$ is the number of suppliers that will be visited in period j. Starting from $s_{\{k\}}$, $k = 1$, if $a_{kj} > 0$ and $a_{k(j-q)} > 0$, then if $a_{kj} > d_{kj}$, set $r = random\left(1, min\{d_{ij}, (R_j - sum)\}\right)$. Otherwise, let $r = random\left(0, min\{a_{kj}, (R_j - sum)\}\right)$. Set $sum = sum + r$, $a_{kj} = a_{kj} - r$, $a_{k(j-q)} = a_{k(j-q)} + r$, $s = s + 1$ and repeat until $sum = R_j$.

Step 4: Perform Step 3 (Steps 3a-3c) – Step 5 as in section 4.4.4.

From the algorithm above, we note that the maximum amount to be transferred for supplier k in period j is not more than d_{kj}. This will ensure that the resultant inventory holding cost will not increase drastically. If the remaining $a_{kj} = 0$, then the chromosome will be modified accordingly.

Other than generating the amount to be transferred (r) randomly as above algorithm, we have also investigated the effect of choosing r deterministically by following the equation $r = min\{d_{kj}\ , (R_j - sum)\}$ for $a_{kj} > d_{kj}$ and $r = min\{a_{kj}\ , (R_j - sum)\}$ otherwise where sum refers to the total amount to be transferred to the preceding period.

5.2 The illustrations of MHGA

To illustrate the MHGA, there are two important matrices that we have to consider that is the Demand Matrix and Delivery/Collection Matrix. In our study, the vehicle capacity, C is set at 10 and $K\%$ is set at 40% and 60%. For this particular example, we will be using example of S5T5 where the number of customers are 5 in period 5 and $K\%$ is set at 40%.

From the Delivery/Collection Matrix in Table 5.2, calculate the Total Delivery/Collection amount for each period (denoted as DP_j) and the percentage excess (with respect to ehicle capacity) which is denoted as R_j. Table 5.3 below shows the value for DP_j and R_j.

Table 5.1: The Demand Matrix

	1	**2**	**3**	**4**	**5**
1	4	2	4	4	4
2	2	2	2	2	2
3	2	1	2	2	2
4	2	1	2	2	2
5	1	2	1	1	1

Table 5.2: The Delivery/Collection Matrix

	1	**2**	**3**	**4**	**5**
1	6	0	8	0	4
2	2	4	0	2	2
3	5	0	0	4	0
4	3	0	2	4	0
5	1	3	0	1	1

Table 5.3: The total delivery/pick-up for each period (DP_j) and excess in comparison to the vehicle capacity (R_j)

	1	2	3	4	5
1	6	0	8	0	4
2	2	4	0	2	2
3	5	0	0	4	0
4	3	0	2	4	0
5	1	3	0	1	1
$\boldsymbol{DP_j}$	**17**	**7**	**10**	**11**	**7**
$\boldsymbol{R_j(\%)}$	**70**	**70**	**0**	**10**	**70**

In the algorithm, the excess delivery (with respect to the vehicle's capacity), R_j for each period j is evaluated by comparing it to a certain value $K\%$. If R_j is less than $K\%$, then the amount will be transferred to the preceding period to be delivered. To avoid a great increase in the inventory holding cost, we limit the number of periods to be transferred to 2. From Table 5.3, starting from $T = 5$, we can see that when $T = 4$, $R_j < 40\%$. Therefore some delivery/collection amount in period $T = 4$ will be shifted to the preceding periods. However, the selection of the suppliers will be done based on the inventory holding cost for each supplier. Table 5.4(a) shows the list of the suppliers with their inventory holding costs while Table 5.4(b) shows the sorted suppliers in ascending order based on their inventory holding costs.

Table 5.4(a): The list of suppliers with their inventory holding costs.

Supplier	Holding Cost
1	24
2	15
3	3
4	9
5	12

Table 5.4(b): The list of assorted suppliers in ascending order based on their inventory holding costs.

Supplier	Holding Cost
3	3
4	9
5	12
2	15
1	24

Table 5.5 shows the period where the amount will be shifted. By setting the preceding period, $q = 1$, it means that some amount from period 4 will be shifted into period 3. We start with the supplier that has the smallest inventory holding cost. From Table 5.5b), the first supplier is supplier 3. Unfortunately, there is no delivery/collection for supplier 3 in period 3 so the amount in period 4 cannot be shifted into period 3. The second lowest inventory holding cost is supplier 4 and there is delivery/collection for supplier 4 in period 3. Hence, the shifted amount can be shifted from period 4 to period 3 for supplier 4.

Table 5.5: The total delivery/pick-up matrix with the sorted supplier.

		Period					
Sorted supplier		1	2	3	**4**	5	HC
s_5	1	6	0	8	**0**	4	24
s_4	2	2	4	0	**2**	2	15
s_1	3	5	0	0	**4**	0	3
s_2	4	3	0	***2***	**4**	0	9
s_3	5	1	3	0	**1**	1	12
	$\boldsymbol{DP_j}$	**17**	**7**	**10**	**11**	**7**	
	$\boldsymbol{R_j}$	**7**	**7**	**0**	**1**	**7**	

In MHGA, there are two ways to determine the amount to be shifted. The first method is to get the deterministic amount and the second method is to randomly generate the amount to be shifted. In this example, we use the random generation amount to be shifted. Since total delivery/collection, $a_{44} > d_{44}$, and the random number is $r = 1$. This will give us $a_{44} = 4 - 1 = 3$ and $a_{43} = 2 + 1 = 3$. The new total delivery/collection matrix is shown in Table 5.6.

Table 5.6: The total delivery/pick-up matrix after the application of MHGA.

		Period					
Sorted supplier		1	2	3	4	5	HC
s_5	1	6	0	8	**0**	4	24
s_4	2	2	4	0	**2**	2	15
s_1	3	5	0	0	**4**	0	3
s_2	4	3	0	***3***	**3**	0	9
s_3	5	1	3	0	**1**	1	12
	$\boldsymbol{DP_j}$	**17**	**7**	**11**	**10**	**7**	
	$\boldsymbol{R_j}$	**7**	**7**	**1**	**0**	**7**	

We note that the maximum amount to be shifted for each customer in each period is not more than $a_{ij} - 1$. This will ensure that the resultant inventory holding cost will not increase in great amount. Besides, if the customer is visited, the delivery/collection amount will not be 0. The original chromosome has to be modified if it is not subjected to this constraint, resulting undesirable and unnecessary cost in terms of computational time.

5.3 Results and Discussions

The algorithms were written in C++ using Genetic Algorithms Library (GALIB) to run the program. The same 14 datasets that have been used for the previous 3 models were used to run this program. In our experiment, the number of generations, crossover rate and mutation rate are fixed at 300, 0.9 and 0.01 respectively for all the problems. The population size is fixed at 200 individuals. Each dataset is executed ten times. Table 5.7 summarizes the best total objective, the mean and standard deviation of total objective for 10 runs, and the number of vehicle for the best total objective for the inbound logistics problems. The best routes and the list of the best objectives for each generation are saved for observation of the converging pattern.

Table 5.7 : The best total objective, the mean and standard deviation of total objective for 10 runs, and the number of vehicle for the best total objective for the inbound logistics problems with MHGA.

Dataset	Inbound Logistics							
	Split Delivery				Un-split Delivery			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S12T5								
Best Obj.	2614.4	2669.7	2771.3	2732.2	2588.6	2643	2652.2	2681.2
Mean	2729.84	2784.27	3026.13	2822.48	2788.66	2832.4	2797.32	2779.22
Std. Dev.	93.5801	142.072	166.737	82.054	198.057	133.165	101.363	63.8421
No. of Veh.	15	16	18	17	16	17	17	16
S12T10								
Best Obj.	5382	5807.4	5162.8	5499.6	5336.5	5377.2	5198.6	5361.3
Mean	5547.2	5926.14	5428.68	5703.86	5438.95	5482.62	5445.66	5396.53
Std. Dev.	187.852	110.097	262.114	203.39	79.9029	96.8379	208.216	261.328
No. of Veh.	33	36	33	33	33	33	34	33
S12T14								
Best Obj.	7273.9	7430.2	7077	7357.7	7100.8	7618	7007.3	7587.4
Mean	7451.09	7611.52	7228.2	7646.87	7229.88	7845.5	7150.73	7675.54
Std. Dev.	130.711	146.048	138.383	137.54	76.7626	143.875	82.551	100.078
No. of Veh.	45	46	46	45	46	48	46	49
S20T5								
Best Obj.	4066.9	4051.7	4035.2	4100.2	4004.5	4039	4053.7	3988.7
Mean	4127.99	4211.17	4164.02	4271.42	4216.75	4366.3	4298.27	4250.27
Std. Dev.	42.243	129.652	95.94	107.775	132.33	236.022	201.479	275.703
No. of Veh.	26	24	25	26	25	24	26	25
S20T10								
Best Obj.	10110.9	10580.3	9382.8	9923.7	9756.8	11163.9	10455.3	9217.6
Mean	10201.3	10760.3	9634.98	10077	9975.78	11444	10553.1	9413.86
Std. Dev.	93.9481	142.514	147.011	143.526	128.054	222.203	113.246	162.755
No. of Veh.	55	59	54	56	55	60	58	52
S20T14								
Best Obj.	10116.2	12122.4	10599.1	10700	10714.3	11441.1	11385.9	11295.1
Mean	10276.6	12288.2	10747.1	10904.1	10931.4	11580.6	11613.4	11664.6
Std. Dev.	150.526	125.088	172.76	115.719	209.201	177.108	173.048	239.6
No. of Veh.	67	68	68	67	73	73	77	75

Table 5.7 (cont.)

Dataset	Inbound Logistics							
	Split Delivery				Un-split Delivery			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S20T21								
Best Obj.	15920	17287	16894	16659	16574	17955	16571	17259
Mean	16268.3	17339.6	17156.7	17006.1	16812.9	18280.8	16751.2	17580.4
Std. Dev.	207.718	125.916	139.13	248.622	230.416	246.861	168.926	195.936
No. of Veh.	102	103	103	106	106	108	110	108
S50T5								
Best Obj.	8199.3	8563.6	8713.2	8682.1	8274.2	8313.3	8399.2	8423.1
Mean	8421.53	8802.26	8830.92	8835.41	8432.75	8695.43	8607.62	8798.51
Std. Dev.	168.429	134.679	139.975	125.714	107.902	273.228	133.096	253.537
No. of Veh.	55	56	59	61	57	57	58	58
S50T10								
Best Obj.	16579	16990	17842	18681	17485	18420	18155	18381
Mean	16766.9	17219.9	17997.2	18870.6	17869.1	18636.5	18332.5	18869.2
Std. Dev.	187.473	111.832	86.1933	223.802	268.189	242.676	127.327	339.402
No. of Veh.	116	120	114	122	117	115	120	124
S50T14								
Best Obj.	21503	22174	24863	25901	24869	26531	25814	25620
Mean	21766.7	22372	25134.7	26001.7	25154.4	26813	26178.2	25941.7
Std. Dev.	125.092	190.517	165.091	97.3528	272.82	234.375	224.588	247.753
No. of Veh.	167	161	158	166	163	170	170	173
S50T21								
Best Obj.	31944	32273	31475	32150	31840	32289	31743	32235
Mean	32250.2	32520.8	31812.4	32217.1	32159.7	32519.8	31917.6	32385.4
Std. Dev.	318.559	207.622	360.389	46.1626	247.474	200.781	165.013	92.5361
No. of Veh.	228	229	227	230	238	246	236	241
S98T5								
Best Obj.	63234.5	67968	62446	73392.5	89771.5	79244	70444	99037
Mean	67710.5	73489.7	67517.7	82960	93293.8	80286.4	76605.3	100620
Std. Dev.	3587.87	3998.79	4049.44	4272.33	5192.89	1229.73	3986.95	1302.49
No. of Veh.	20	24	32	28	36	37	32	35
S98T10								
Best Obj.	150403	131050	234004	175237	240474	168517	200331	187293
Mean	152185	131552	236798	177204	250271	178523	208934	225950
Std. Dev.	1037.9	601.566	1882.89	732.586	9865.83	7008.31	7529.77	17686.9
No. of Veh.	10	10	78	50	69	72	71	72
S98T14								
Best Obj.	275823	345099	1064408	1098795	294420	241050	1165740	1193520
Mean	276324	346734	1070488	1104433	306004	247533	1171130	1211484
Std. Dev.	473.483	1062.84	3349.53	5020.52	7559.25	5623.26	4472.86	8315.51
No. of Veh.	14	14	143	147	134	141	157	156

Table 5.8 : The best total objective, the mean and standard deviation of total objective for 10 runs, and the number of vehicle for the best total objective for the outbound logistics problems with MHGA.

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	40%	60%	40%	60%	40%	60%	40%	60%
S12T5								
Best Obj.	3990.4	4048.1	3945.4	4127	3953.3	3968.4	3860	4081.3
Mean	4111.5	4123.43	4156.69	4316.54	4004.2	4013.88	4049.54	4173.23
Std. Dev.	57.342	46.4352	123.688	151.456	23.8453	31.0028	118.593	78.2818
No. of Veh.	19	20	22	21	19	19	19	21
S12T10								
Best Obj.	4343.7	4390	4116.5	4512.7	4106	4308.1	4008.4	4432.5
Mean	4484.6	4512.53	4727.19	4835.34	4162.02	4362.21	4603.64	4718.35
Std. Dev.	146.34	95.953	277.53	274.212	55.3543	64.0504	345.551	204.864
No. of Veh.	33	32	32	31	31	32	31	33
S12T14								
Best Obj.	6813	7179.6	6895.3	55247	6899.6	7188	6881	7116
Mean	7619.43	7408.75	7159.23	7423.69	7475.41	7300.42	7022.11	7280.53
Std. Dev.	204.543	164.23	147.665	152.395	337.994	72.1709	80.5914	135.701
No. of Veh.	44	52	42	48	45	47	44	46
S20T5								
Best Obj.	6471.3	6504.2	6825.4	6837.8	6461.2	6517	6662.6	6765.8
Mean	6810.48	6852.77	7221.28	6976.23	6691.6	6733.92	7072.37	6846.34
Std. Dev.	262.274	210.221	256.453	86.3925	179.75	123.222	223.986	78.9283
No. of Veh.	31	32	33	31	33	33	33	32
S20T10								
Best Obj.	13771	13937.6	8117.4	7714.1	13566.7	13869.9	7999.1	7765
Mean	12628.7	14379.4	8314.25	8033.35	12487.1	14235.5	8165.21	7925.21
Std. Dev.	402.235	206.364	131.372	137.384	4391.44	236.154	104.694	116.77
No. of Veh.	63	59	49	44	63	60	49	46
S20T14								
Best Obj.	13612.1	14910.2	13887.7	14546.5	13349.2	14726	13644.5	14405
Mean	13786.5	14928.3	14310.2	14949.3	13676.9	14794.4	13801.5	14508.7
Std. Dev.	216.395	78.503	129.352	157.23	163.758	56.2294	119.518	77.7756
No. of Veh.	77	79	77	85	77	81	78	81
S20T21								
Best Obj.	16171.5	17895.3	16615.5	17067.2	16163	17715	16506	17203
Mean	16804.4	18195.3	16954.3	17509.6	16702.2	18076.1	16835.9	17395.4
Std. Dev.	235.938	305.869	307.23	171.112	396.731	262.099	295.488	162.073
No. of Veh.	107	110	108	110	105	108	107	112

Table 5.8 (cont.)

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S50T5								
Best Obj.	9787.2	10033.4	8545.1	8531.6	9612.7	9818.3	8427	8324
Mean	9858.24	10293.2	8706.32	8726.86	9714.57	9952.44	8583.7	8589.2
Std. Dev.	161.93	98.3405	307.348	172.675	93.1292	79.5741	210.165	142.976
No. of Veh.	62	60	58	60	59	59	57	59
S50T10								
Best Obj.	18724	17162	18141	18643	17565	17133	17920	18489
Mean	19902.6	17722.6	18437	19912.2	17764.3	17621.9	18287.6	18787.2
Std. Dev.	223.334	98.349	362.325	263.323	132.798	364.907	321.71	182.677
No. of Veh.	112	112	114	126	112	112	117	128
S50T14								
Best Obj.	16703	18097	25576	25514	17679	18422	25536	25640
Mean	18008.2	18786.7	27231.2	26021.4	17883.2	18638.6	26099.6	25871.2
Std. Dev.	186.348	158.958	410.134	274.115	124.767	131.406	329.366	193.667
No. of Veh.	149	152	167	170	151	154	170	171
S50T21								
Best Obj.	32900	33972	31487	31426	34077	34205	31550	31733
Mean	34316.2	34706.2	32016	32651.3	34205.5	34572.3	31892.3	32526.4
Std. Dev.	95.483	408.356	215.42	470.325	60.6305	323.651	215.919	659.831
No. of Veh.	230	234	225	227	241	244	230	234
S98T5								
Best Obj.	78153	81109	66695	97408	85339	86323	70210	102035
Mean	87958.4	92843.2	72021.3	99948.1	90834.9	92706.9	71890.8	103809
Std. Dev.	2483.13	2245.35	2549.24	2484.84	3786.18	4061.71	861.07	2385.39
No. of Veh.	23	25	31	32	36	37	35	35
S98T10								
Best Obj.	206581	161345	175005	178063	231126	164292	206924	161092
Mean	236668	173146	191953	198857	236520	173042	211825	168745
Std. Dev.	4728.73	5378.84	2820.12	6619.24	4725.64	5373.01	2762.82	8616.9
No. of Veh.	54	63	82	84	63	72	182	79
S98T14								
Best Obj.	222075	225755	265063	235608	222790	222862	266271	236793
Mean	265112	247527	277090	280741	264979	247403	276952	280615
Std. Dev.	12564.2	17126.4	4574.23	26015.3	22464	17120.1	5550.26	25962.4
No. of Veh.	51	49	132	40	52	48	135	41

Generally, from Table 5.7 and Table 5.8 above, the standard deviation of the total objectives increases steadily from the small-sized datasets to the medium-sized datasets. There seems to be not so much different in the best objectives, mean and the number of vehicles. For most results, when we set $K = 40\%$, the best objectives seems to be lower than when we set $K = 60\%$. The purpose of setting K is to limit the size of the amount to be transferred to the previous period. Therefore, the least we shift the amount, the lower the inventory holding costs would be.

For some small instances such as S12T5, S20T5, the best objectives for the un-split delivery and pick-up cases are lower compared with the Hybrid Genetic Algorithms results. However, as the size of instances increases, the number of difference of the best objectives between HGA and MHGA method seem to be quite large.

In large instances (S98T5, S98T10, S98T14), the standard deviations for the objective functions are large. As explained in Chapter 4, this is probably because the maximum number of generation is not sufficient enough for the algorithm to converge.

Table 5.9 and Table 5.10 below show the characteristics of the best results given in Table 5.7 and Table 5.8. It gives the distance costs, total inventory holding costs, number of vehicle and the best total objective costs for 10 runs.

Table 5.9: The distance costs, total inventory holding costs, number of vehicle and the best total objective costs for 10 runs from Table 5.7.

Dataset	Inbound Logistics							
	Split Delivery				Un-split Delivery			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S12T5								
Distance	1765.4	1812.7	1976.3	1747.2	1863.6	1898	1841.2	1806.2
Inventory	549	537	435	645	405	405	471	555
Tot Veh	15	16	18	17	16	17	17	16
Tot Obj	2614.4	2669.7	2771.3	2732.2	2588.6	2643	2652.2	2681.2
time	463	437	446	439	443	467	465	446
S12T10								
Distance	3966	4268.4	3965.8	3801.6	3950.5	3946.2	3966.6	3882.3
Inventory	756	819	537	1038	726	771	552	819
Tot Veh	33	36	33	33	33	33	34	33
Tot Obj	5382	5807.4	5162.8	5499.6	5336.5	5377.2	5198.6	5361.3
time	715	694	728	667	698	697	750	687
S12T14								
Distance	5239.9	5247.2	5419	5242.7	5379.8	5518	5286.3	5425.4
Inventory	1134	1263	738	1215	801	1140	801	1182
Tot Veh	45	46	46	45	46	48	46	49
Tot Obj	7273.9	7430.2	7077	7357.7	7100.8	7618	7007.3	7587.4
time	837	838	880	845	932	845	885	875
S20T5								
Distance	2733.9	2497.7	2710.2	2653.2	2640.5	2578	2786.7	2603.7
Inventory	813	1074	825	927	864	981	747	885
Tot Veh	26	24	25	26	25	24	26	25
Tot Obj	4066.9	4051.7	4035.2	4100.2	4004.5	4039	4053.7	3988.7
time	681	655	692	661	672	654	694	689
S20T10								
Distance	6400.9	6775.3	6292.8	6397.7	6613.8	6648.9	6595.3	6131.6
Inventory	2610	2625	2010	2406	2043	3315	2700	2046
Tot Veh	55	59	54	56	55	60	58	52
Tot Obj	10110.9	10580.3	9382.8	9923.7	9756.8	11163.9	10455.3	9217.6
time	978	929	1012	1000	978	888	968	1022
S20T14								
Distance	7096.2	7129.4	7079.1	7038	7187.3	7086.1	7697.9	7509.1
Inventory	1680	3633	2160	2322	2067	2895	2148	2286
Tot Veh	67	68	68	67	73	73	77	75
Tot Obj	10116.2	12122.4	10599.1	10700	10714.3	11441.1	11385.9	11295.1
time	1393	1195	1343	1345	1250	1200	1363	1342

Table 5.9 (cont.)

Dataset	Inbound Logistics							
	Split Delivery				Un-split Delivery			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	40%	60%	40%	60%	40%	60%	40%	60%
S20T21								
Distance	11294	11195	11255	11554	11331	11001	11821	11697
Inventory	2586	4032	3579	2985	3123	4794	2550	3402
Tot Veh	102	103	103	106	106	108	110	108
Tot Obj	15920	17287	16894	16659	16574	17955	16571	17259
time	1834	1653	1788	1833	1732	1541	2247	2217
S50T5								
Distance	5881.3	6163.6	6323.2	6401.1	5807.2	5806.3	5992.2	6026.1
Inventory	1218	1280	1210	1061	1327	1367	1247	1237
Tot Veh	55	56	59	61	57	57	58	58
Tot Obj	8199.3	8563.6	8713.2	8682.1	8274.2	8313.3	8399.2	8423.1
time	1873	1849	1816	1831	1760	1748	1824	1832
S50T10								
Distance	11418	11604	12487	13474	12412	12494	12820	13450
Inventory	2841	2986	3075	2767	2733	3626	2935	2451
Tot Veh	116	120	114	122	117	115	120	124
Tot Obj	16579	16990	17842	18681	17485	18420	18155	18381
time	2534	2418	2348	2422	2288	2214	2355	2360
S50T14								
Distance	14807	14717	17391	18657	17449	18239	18888	18517
Inventory	3356	4237	4312	3924	4160	4892	3526	3643
Tot Veh	167	161	158	166	163	170	170	173
Tot Obj	21503	22174	24863	25901	24869	26531	25814	25620
time	3188	2977	2732	2975	2699	2634	2849	2842
S50T21								
Distance	22144	21950	21904	22573	22046	21561	21920	22714
Inventory	5240	5743	5031	4977	5034	5808	5103	4701
Tot Veh	228	229	227	230	238	246	236	241
Tot Obj	31944	32273	31475	32150	31840	32289	31743	32235
time	3937	3825	3903	4042	4340	3751	4188	5192
S98T5								
Distance	245.69	280.1	318.94	293.09	361.55	328.4	315.54	372.86
Inventory	46950	49163	40099	53138	64494	55424	48267	73394
Tot Veh	20	24	32	28	36	37	32	35
Tot Obj	63234.5	67968	62446	73392.5	89771.5	79244	70444	99037
time	9911	9883	8897	8786	11021	11194	9387	9130

Table 5.9 (cont.)

Dataset	Inbound Logistics							
	Split Delivery				Un-split Delivery			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	40%	60%	40%	60%	40%	60%	40%	60%
S98T10								
Distance	61.45	62.6	904.08	580.13	649.07	712.33	719.61	754.65
Inventory	145330	125920	173200	136230	194220	118500	150150	135160
Tot Veh	10	10	78	50	69	72	71	72
Tot Obj	150403	131050	234004	175237	240474	168517	200331	187293
time	16539	16378	14185	15267	17182	23499	17450	18034
S98T14								
Distance	89.86	95.98	13763	14335	1256.2	1350	15642	15900
Inventory	268530	337500	347658	352645	204810	145350	352240	367320
Tot Veh	14	14	143	147	134	141	157	156
Tot Obj	275823	345099	1064408	1098795	294420	241050	1165740	1193520
time	22642	23578	18976	17685	23083	23595	19787	19744

From Table 5.9 above, we can see that in large instances, there seem to be inconsistency in the number of vehicle used in split delivery problem where the number varies quite considerably between $K = 40\%$ and $K = 60\%$. However, for the un-split delivery problem, the numbers of vehicle used are within the acceptable range.

Table 5.10: The distance costs, total inventory holding costs, number of vehicle and the best total objective costs for 10 runs from Table 5.8.

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	40%	60%	40%	60%	40%	60%	40%	60%
S12T5								
Distance	2494.4	2570.1	2624.4	2721	2370.3	2520.4	2528	2671.3
Inventory	1116	1078	881	986	1203	1068	952	990
Tot Veh	19	20	22	21	19	19	19	21
Tot Obj	3990.4	4048.1	3945.4	4127	3953.3	3968.4	3860	4081.3
time	354	418	526	552	431	404	592	565

Table 5.10 (cont.)

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S12T10								
Distance	3224.7	3227	3043.5	3295.7	3024	3083.1	3085.4	3214.5
Inventory	459	523	433	597	462	585	303	558
Tot Veh	33	32	32	31	31	32	31	33
Tot Obj	4343.7	4390	4116.5	4512.7	4106	4308.1	4008.4	4432.5
time	762	784	859	792	724	729	934	869
S12T14								
Distance	5235	5451.6	5388.3	53304	5366.6	5477	5291	5298
Inventory	698	688	667	983	633	771	710	898
Tot Veh	44	52	42	48	45	47	44	46
Tot Obj	6813	7179.6	6895.3	55247	6899.6	7188	6881	7116
time	1143	1076	1216	1111	1191	1122	1194	1069
S20T5								
Distance	4068.3	4017.2	4135.4	3946.8	4082.2	3976	4073.6	3800.8
Inventory	1783	1847	2030	2271	1719	1881	1929	2325
Tot Veh	31	32	33	31	33	33	33	32
Tot Obj	6471.3	6504.2	6825.4	6837.8	6461.2	6517	6662.6	6765.8
time	717	637	726	675	639	623	706	644
S20T10								
Distance	8421	7866.6	5688.4	5583.1	8331.7	7857.9	5594.1	5532
Inventory	4090	4891	1449	1251	3975	4812	1425	1313
Tot Veh	63	59	49	44	63	60	49	46
Tot Obj	13771	13937.6	8117.4	7714.1	13566.7	13869.9	7999.1	7765
time	868	806	1069	1044	906	863	1112	1059
S20T14								
Distance	9093.1	9144.2	8815.7	9067.5	8917.2	8944	8652.5	8934
Inventory	2979	4186	3532	3779	2892	4162	3432	3851
Tot Veh	77	79	77	85	77	81	78	81
Tot Obj	13612.1	14910.2	13887.7	14546.5	13349.2	14726	13644.5	14405
time	1403	1294	1409	1454	1374	1224	1316	1466
S20T21								
Distance	11487.5	11472.3	11234.5	11596.2	11477	11304	11231	11600
Inventory	2544	4223	3221	3271	2586	4251	3135	3363
Tot Veh	107	110	108	110	105	108	107	112
Tot Obj	16171.5	17895.3	16615.5	17067.2	16163	17715	16506	17203
time	2457	2514	2143	2290	2486	2575	2212	2312

Table 5.10 (cont.)

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S50T5								
Distance	7238.2	7294.4	6139.1	6159.6	7119.7	7210.3	6025	5947
Inventory	1309	1539	1246	1172	1313	1428	1262	1197
Tot Veh	62	60	58	60	59	59	57	59
Tot Obj	9787.2	10033.4	8545.1	8531.6	9612.7	9818.3	8427	8324
time	1950	1776	1928	1739	1959	1826	1965	1722
S50T10								
Distance	14069	12335	12733	13726	12922	12275	12541	13538
Inventory	2415	2587	3128	2397	2403	2618	3039	2391
Tot Veh	112	112	114	126	112	112	117	128
Tot Obj	18724	17162	18141	18643	17565	17133	17920	18489
time	2596	2449	2117	2468	2528	2376	2195	2411
S50T14								
Distance	10903	11504	18492	18377	11770	11794	18505	18524
Inventory	2820	3553	3744	3737	2889	3548	3631	3696
Tot Veh	149	152	167	170	151	154	170	171
Tot Obj	16703	18097	25576	25514	17679	18422	25536	25640
time	2903	3888	2700	2766	2960	3831	2768	2754
S50T21								
Distance	23054	23099	21785	22399	23934	23194	21829	22477
Inventory	5246	6193	5202	4487	5323	6131	5121	4576
Tot Veh	230	234	225	227	241	244	230	234
Tot Obj	32900	33972	31487	31426	34077	34205	31550	31733
time	5233	4822	4146	5035	5230	4868	4185	4992
S98T5								
Distance	269.3	321.6	262.4	328.7	359.5	379	299	431
Inventory	60088	60029	47375	74573	60164	59973	48260	73485
Tot Veh	23	25	31	32	36	37	35	35
Tot Obj	78153	81109	66695	97408	85339	86323	70210	102035
time	10140	10744	9462	9230	10125	10679	9470	9205
S98T10								
Distance	444	603.9	893.2	753.6	481	627	1731.5	834.03
Inventory	173581	118550	113945	123583	194476	118542	83949	103590
Tot Veh	54	63	82	84	63	72	182	79
Tot Obj	206581	161345	175005	178063	231126	164292	206924	161092
time	17079	23787	12495	12622	17084	23750	12498	12561

Dataset	Outbound Logistics							
	Split Pick-up				Un-split Pick-up			
	Fixed shifted amount		Random shifted amount		Fixed shifted amount		Random shifted amount	
K%	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**	**40%**	**60%**
S98T14								
Distance	476.9	504.3	1009.6	378.5	488.59	449.64	1022	397.65
Inventory	188030	190740	188183	208683	187960	190780	188171	208710
Tot Veh	51	49	132	40	52	48	135	41
Tot Obj	222075	225755	265063	235608	222790	222862	266271	236793
time	22608	23073	19579	25021	22678	23097	19651	24922

In fixed shifted amount, generally the distance costs and inventory costs for $K = 40\%$ are lower than when $K = 60\%$. This is to be expected because the size to be shifted for $K = 40\%$ is normally lower than $K = 60\%$. Thus, there will be fewer amounts to be held in the preceding period and less travelling is done to send the items. However this is not necessarily true for the random shifted amount where the amount shifted does not constitute a full amount.

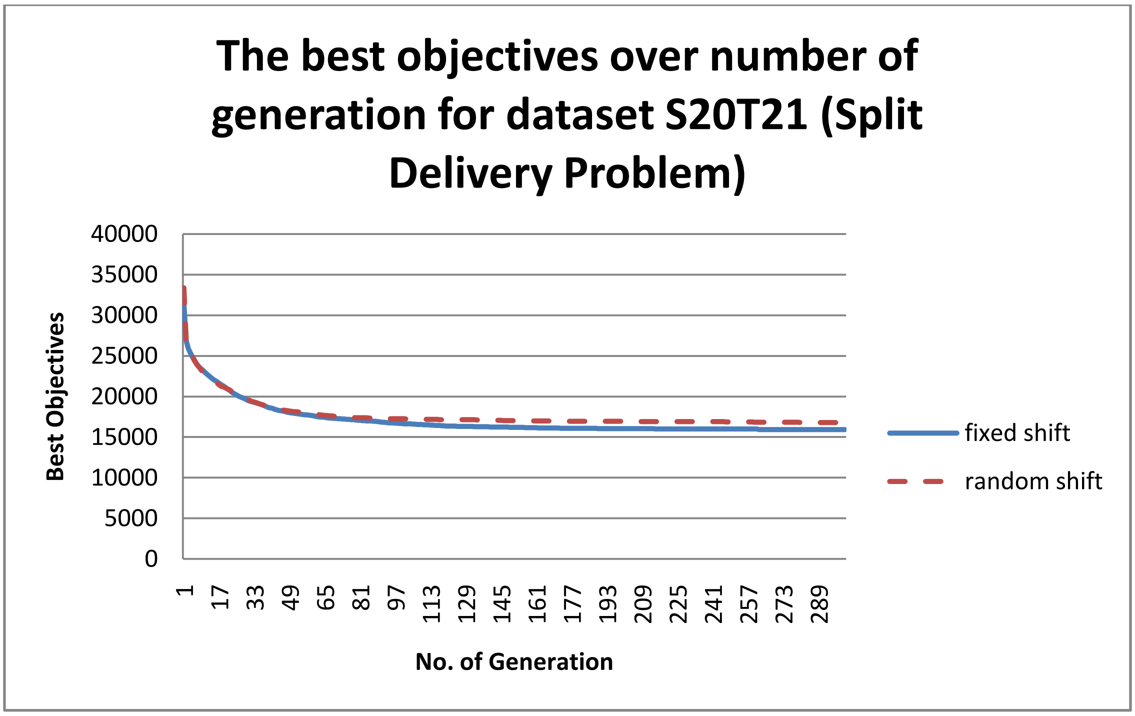

Figure 5.1: The best objectives over the number of generation for dataset S20T21 for Split Delivery Case

Figure 5.1 shows graph of the best objectives over the number of generation for dataset S20T21 for split delivery case. The graph has been plotted based on the fixed shift algorithm and the random shift algorithm. This graph is an example of convergence graph for small datasets. Based on the graph, the fixed shift and the random shift algorithm did not show much difference in terms of converging to get the objective value. Basically, after 250 generations, the algorithm finally finds the optimum objective value for the dataset.

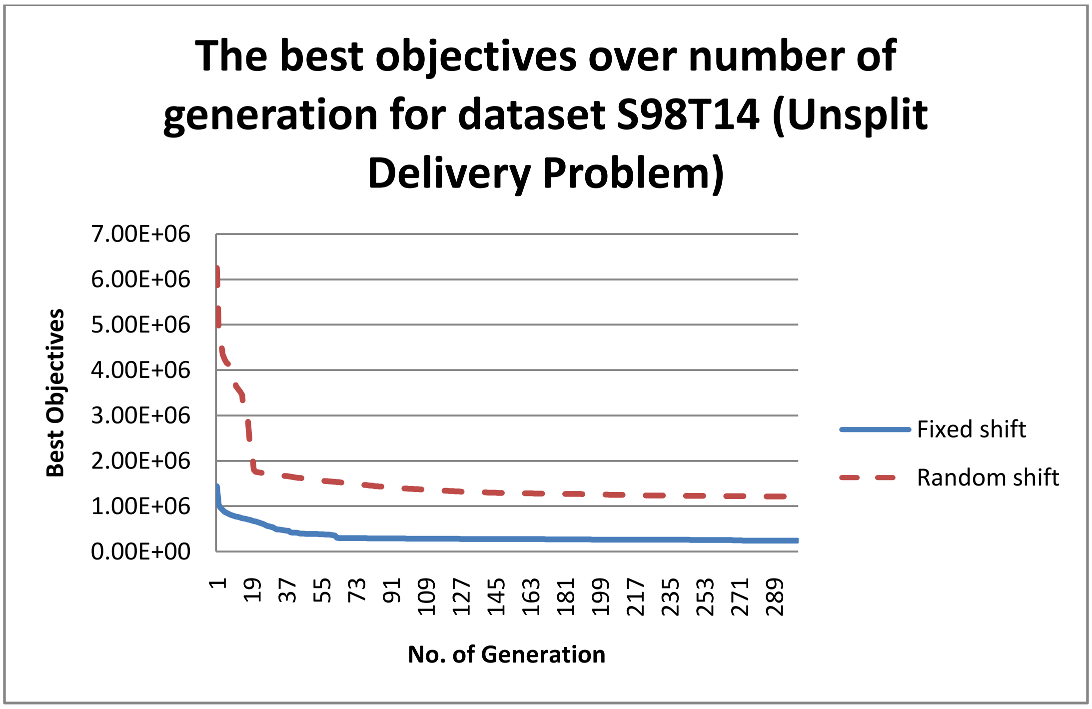

Figure 5.2: The best objective over the number of generation for dataset S98T14 for un-split delivery problem.

On the other hand, Figure 5.2 shows the convergence graph for dataset S98T14 with un-split delivery case. It can be seen that with Genetic Algorithms, the algorithms are quick to find the lower objective value. However, in order to converge into the optimal objective value, the algorithm needs to have more number of generations.

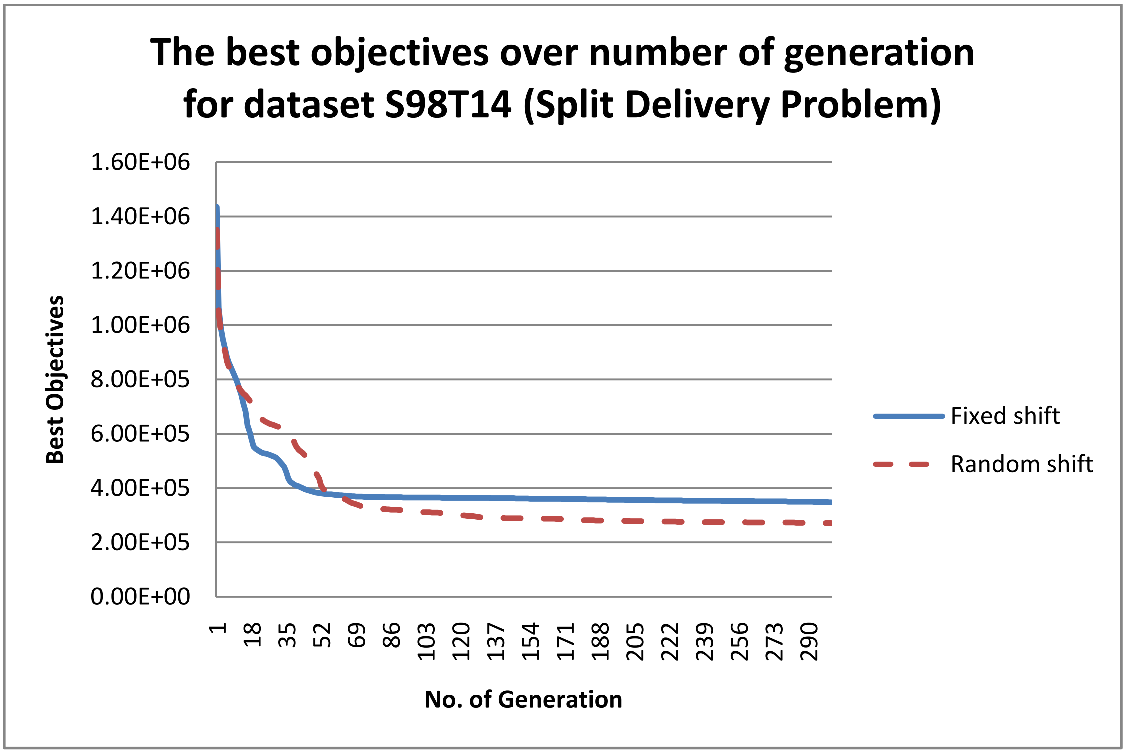

Figure 5.3: The best objectives over number of generations for dataset S98T14 with split delivery case.

For the split delivery problem with large dataset, Figure 5.3 above shows that the convergence graph for the fixed shift and random shift algorithm. It can be seen that in the split delivery case, the random shift algorithms find the lower objective value faster than the fixed shift algorithm.

It can be concluded that for small instances, the number of generation is sufficient to find the objective value. On the other hand, for large datasets, we will need bigger number of generation for the algorithm to converge to find the objective value. This however will be time consuming.

5.4 Post-Optimization

The routing within each cluster can obviously be improved using existing refinement procedures. In this study, a simple local search 2-opt which is originally proposed by Croes in 1958 [43] is performed on the best chromosome found. The general concept of 2-opt procedure is to obtain a new tour by eliminating two edges and reconnecting the paths in different way. This procedure is repeated until the shortest tour is found. This additional task however, does not contribute significantly to the cpu times since the number of suppliers within a route is relatively small due to the capacity constraint.

5.4.1 Results and discussions

In this study, the 2-opt optimization is applied on the results from HGA method in Chapter 4. This result is then compared to the work done by Moin et. al [6] that uses Variable Neighborhood Search (VNS) Algorithm on the same datasets used in this study. VNS were written in MATLAB 7.1.

VNS method is based on exploration of a systematic neighborhood model. The principles of VNS is that different neighborhoods generate different search topologies [35, 44]. The systematic change of neighborhood is applied within a local search algorithm that can be applied repeatedly in order to move from the incumbent solution. There are several ways that can be used to define the neighborhood structure, for example: 1-interchange, symmetric difference between two solutions, Hamming distance, vertex deletion or addition, node based or path based and *k*-edge exchange.

Table 5.11: The comparison of HGA results after 2-opt and VNS results.

DATA SET	METHOD	TOTAL COST	HOLDING COST	NUM OF VEH	DISTANCE	TIME
S12T5	GA	2385.9	378	15	1707.9	41.2467
	VNS	**2116.7**	261	14	1575.7	184.596
S12T10	GA	4657.04	507	30	3550.04	81.5261
	VNS	**4400.44**	369	31	3411.44	437.661
S12T14	GA	6882.31	621	43	5401.31	114.146
	VNS	**6301.09**	498	45	4903.09	533.867
S20T5	GA	**3210.89**	198	22	2572.89	61.402
	VNS	3214.66	345	24	2389.66	1200.77
S20T10	GA	6890.2	537	48	5393.2	117.219
	VNS	**6689**	690	49	5019	2384.88
S20T14	GA	9716.69	570	70	7746.69	167.42
	VNS	**9575.98**	1005	68	7210.98	6901.06
S20T21	GA	14672.4	933	106	11619.4	257.464
	VNS	**14498.3**	1425	106	10953.3	6029.84
S50T5	GA	5729.6	453	47	4336.6	2559
	VNS	**5448.8**	270	46	4258.8	15921
S50T10	GA	12081.1	1213	100	8868.1	3630
	VNS	**11493.6**	545	102	8908.58	57054.7
S50T14	GA	17521	1934	143	12727	7405
	VNS	**16699.2**	650	148	13089.2	70968.1
S50T21	GA	27189	3281	220	19508	8708
	VNS	**25520.6**	1005	225	20015.6	47699.9
S98T5	GA	**614787**	7294.89	63	11897.8	226.623
	VNS	624073	1210	65	12431.2	4560.32

Table 5.11 (cont.)

DATA SET	METHOD	TOTAL COST	HOLDING COST	NUM OF VEH	DISTANCE	TIME
S98T10	GA	1223764	16767.21	125	23639.9311	461.701
	VNS	1238072	3024.5	129	24649.3569	11125.7
S98T14	GA	1722226	20997.02	176	33320.577	653.441
	VNS	1737358	3467.75	181	34605.408	24589.9

From Table 5.11 above, it is observed that VNS seems to outperform GA in small and medium sized problems. However, for the large data size S98T5 – S98T14, GA after post-optimization seems to give better results compared to VNS. The CPU time for GA are extremely lower compared as compared to VNS. It is interesting to note that although VNS emphasizes on reducing the travelling costs, it generally produces a slightly higher travelling distance as compare to GA especially on the large instances data size. For small instances, only small difference can be seen in the number of vehicles used in GA and VNS but in large data size, GA generates slightly less number of vehicles than its counterparts.

5.5 Reformulation of the IRP Model

We reformulate the IRP model by Lee et al [1] (given in Section 2.2) using the maximal flow approach with the route length constraint (constraint 2.12) removed from the formulation. The reformulation is done in order to reduce the dimension of the formulation because from our trial runs, CPLEX is not able to run with the current formulation. CPLEX 9.1 is used to generate the lower bound for the data sets used in this study. A new mathematical formulation is shown below:

Indices

$S = \{1,2,\dots,N\}$	A set of suppliers where supplier i ($i \in S$) supplies product i only.
$D = \{0\}$	Depot
$P = \{N+1\}$	Assembly plant
$\tau = \{1,2,\dots,T\}$	Period index

Parameters

C	Vehicle's capacity
F	Fixed vehicle cost per trip (assumed to be the same for all periods)
V	Travel cost per unit distance
M	Size of the vehicle fleet and it is assumed to be unlimited
d_{it}	Demand for product from supplier i (at the assembly plant) in period t
c_{ij}	Travel distance between supplier i and j where $c_{ij} = c_{ji}$ and the triangle inequality, $c_{ik} = c_{kj} \geq c_{ij}$, holds for any i, j and k with $i \neq j$, $k \neq i$ and $k \neq j$
h_i	Inventory carrying cost at the assembly plant for product from supplier i per unit product per unit time
I_{i0}	Initial inventory level of product from supplier i (at the assembly plant) at the beginning of period 1

Variables

a_{it}	Total amount to be picked-up at supplier i in period t
I_{it}	Inventory level of product from supplier i at the assembly plant at the end of period t
q_{ijt}	Quantity transported through the directed arc (i, j) in period t
x_{ijt}	Number of times that the directed arc (i, j) is visited by vehicles in period t

Objective Function

$$Z = \min \underbrace{\sum_{i \epsilon S} h_i \left(\sum_{t \epsilon \tau} I_{it} \right)}_{A}$$

$$+ \underbrace{V \left(\sum_{\substack{j \epsilon S \\ j \neq i}} \sum_{i \epsilon S \cup D} c_{ij} \left(\sum_{t \epsilon \tau} x_{ijt} \right) + \sum_{i \epsilon S} c_{i,N+1} \left(\sum_{t \epsilon \tau} x_{i,N+1,t} \right) \right)}_{B} \tag{5.1}$$

$$+ \underbrace{\left(F + c_{N+1,0} \right) \sum_{t \epsilon \tau} \sum_{i \epsilon S} x_{0it}}_{C}$$

Subject to

$$I_{it} = I_{i,t-1} + a_{it} - d_{it}, \quad \forall\, i \in S, \qquad \forall\, t\, \epsilon\, \tau \tag{5.2}$$

$$\sum_{\substack{i \in S \cup D \\ i \neq j}} q_{ijt} + + a_{jt} = \sum_{\substack{i \in S \cup D \\ i \neq j}} q_{jit}, \quad \forall j \in S, \qquad \forall t \in \tau \tag{5.3}$$

$$\sum_{i \in S} q_{i,N+1,t} = \sum_{i \in S} a_{it}, \qquad \forall t \in \tau \tag{5.4}$$

$$\sum_{\substack{i \in S \cup D \\ i \neq j}} x_{ijt} = \sum_{\substack{i \in S \cup D \\ i \neq j}} x_{jit}, \quad \forall j \in S, \qquad \forall t \in \tau \tag{5.5}$$

$$\sum_{j \in S} x_{ijt} = \sum_{j \in S} x_{jkt}\ , \qquad i \in D, k \in P, \forall t \in \tau \tag{5.6}$$

$$q_{ijt} \le C x_{ijt}\,, \qquad \forall i \in S, \qquad \forall j \in S \cup P, \qquad i \ne j, \qquad \forall t \in \tau \tag{5.7}$$

$$I_{it} \ge 0, \qquad \forall\, i \in S, \qquad \forall\, t \,\epsilon\, \tau \tag{5.8}$$

$$a_{it} \ge 0, \qquad \forall\, i \in S, \qquad \forall\, t \,\epsilon\, \tau \tag{5.9}$$

$$x_{ijt} \in \{0,1\}, \qquad \forall i, j \in S, \qquad \forall t \in \tau \tag{5.10}$$

$$x_{0jt} \ge 0,\ \text{and integer}, \forall j \in S,\ \forall t \in \tau \tag{5.11}$$

$$x_{i,N+1,t} \ge 0,\ \text{and integer}, \forall i \in S,\ \forall t \in \tau \tag{5.12}$$

$$x_{ijt} = 0, \qquad i \in D, j \cup P, \forall t \in \tau \tag{5.13}$$

$$x_{ijt} = 0, \qquad i \in S, j \cup D, \forall t \in \tau \tag{5.14}$$

$$x_{ijt} = 0, \qquad i \in P, j \cup S, \forall t \in \tau \tag{5.15}$$

$$q_{ijt} \ge 0, \qquad \forall i \in S, \forall j \in S \cup P, \forall t \in \tau \tag{5.16}$$

$$q_{0it} = 0, \qquad \forall i \in S, \quad \forall t \in \tau \tag{5.17}$$

The objective function (5.1) comprises both the inventory costs (A) and the transportation costs (variable travel costs (B) and vehicle fixed cost (C)). We note that the fixed transportation cost consists of the fixed cost incurred per trip and the constant cost of vehicles returning to the depot from the assembly plant. The number of trips in period t is $\sum_{i \in S} x_{0it}$. Equation (5.2) is the inventory balance equation for each product at the assembly plant whilst (5.3) is the product flow conservation equations, assuring the flow balance at each supplier and eliminating all subtours. (5.4) assures the accumulative picked-up quantities at the assembly plant and (5.5) and (5.6) ensure that the number of vehicles leaving a supplier, assembly plant or the depot is equal to the number of its arrival vehicles. We note that constraint (5.6) is introduced because each vehicle has to visit the plant before returning to the depot. (5.7) guarantees that the vehicle capacity is respected and gives the logical relationship between q_{ijt} and x_{ijt}

which allows for split pick-ups. (5.8) ensures that the demand at the assembly plant is completely fulfilled without backorder. The remaining constraints are the nonnegativity constraints imposed on the variables. We note that (5.13)-(5.15) ensure that there is no direct link from the depot to the plant, from supplier to the depot and from plant to the suppliers, respectively. We also note that this formulation does not impose the maximum fleet size. However, if an upper bound on the fleet size is known a priory for a given period t, say M, then the following constraint $\sum_{i \in S} x_{0it} \leq M$ can be added.

5.5.1 Results and discussion

For all the instances, we let CPLEX run for a time limit of 3600s when we record the lower bound and the best integer solution found. In the implementation of the GAs, the number of generations, the generation gap and the crossover rate are fixed at 300, 0.9, and 0.7 respectively, for all problems. The mutation rate for all the algorithms is fixed at 0.001 with the exception of the real representation. The mutation rate for this algorithm is fixed at 0.1 as our limited experiments indicate that this algorithm performs better with higher mutation rates. The population size is fixed at 200 individuals. The maximum number of generations for the data sets with 98 suppliers is increased to 600 because of the large data size. All other parameters were kept the same and 10 runs were performed on each data set.

Table 5.12 summarizes the best total costs, the number of vehicles and the cpu time for each of our algorithms along with the lower bound and the best integer solutions (upper bound) obtained from CPLEX.

Table 5.12: Best total costs, no. of vehicles and CPU (s) for all the algorithms

Data set	(N,τ)	CPLEX (after 3600s)			HGA1[a]			KBGA2[b]		
		LB	Best Integer	No. of Veh.	Best objective	No. of Veh	CPU (s)	Best objective	No. of Veh	CPU (s)
S12T5	(12,5)	1650	1881	14	2099.31	14	46.66	**2096.75**	14	58.48
S12T10	(12,10)	3218	3797	28	**4333.27**	29	111.23	4350.99	29	56.7
S12T14	(12,14)	4709	5645	40	**6115.19**	41	120.31	6172.04	41	157.17
S20T5	(20,5)	2607	2895	+	**3143.39**	21	31.81	3170.68	21	85.52
S20T10	(20,10)	5227	6080	+	6543.08	44	65.81	6720.64	44	169.42
S20T14	(20,14)	7181	8772	64	**9208.43**	61	360.33	9571.85	62	237.31
S20T21	(20,21)	10717	14093	+	**13948.41***	92	255.83	14462.34	96	362.02
S50T5	(50,5)	4547	5071	46	5681.58	45	105.63	5633.37	45	217.68
S50T10	(50,10)	9289	11910	102	11906.00*	95	213.92	11986.02	96	408.44
S50T14	(50,14)	13193	18264	150	17143.77*	136	307.93	17477.05*	137	303.75
S50T21	(50,21)	20185	29975	248	**26448.77***	209	496.72	27034.00*	210	723.95
S98T5	(98,5)	544036	604205	53	561592.59*	57	609.45	564531.95*	57	113.85
S98T10	(98,10)	NA	NA	NA	**1124797.57***	113	1307.26	1132874.15*	114	214.52
S98T14	(98,14)	NA	NA	NA	**1571652.32***	159	1589.71	1596783.40*	161	310.83

Data set	(N,τ)	MHGA1[c] (random shift)			MHGA2[d] (fixed shift)		
		Best objective	No. of Veh	CPU (s)	Best objective	No. of Veh	CPU (s)
S12T5	(12,5)	2099.31	14	48.63	2099.31	14	48.52
S12T10	(12,10)	**4333.27**	29	93.44	**4333.27**	29	101.09
S12T14	(12,14)	**6115.19**	41	123.01	6131.72	41	129.61
S20T5	(20,5)	3178.16	21	68.09	3175.46	21	133.33
S20T10	(20,10)	**6499.4**	43	126.03	6620.9	44	127.3
S20T14	(20,14)	9243.23	61	177.08	9287.64	62	179.45
S20T21	(20,21)	14028.48*	93	273.27	14024.35*	93	434.21
S50T5	(50,5)	**5618.09**	45	133.4	5705.55	45	125.57
S50T10	(50,10)	11940.23	95	269.23	**11642.00***	95	226.01
S50T14	(50,14)	17155.62*	135	340.24	**16987.00***	135	328.07
S50T21	(50,21)	26458.80*	209	563.48	26450.18*	208	506.32
S98T5	(98,5)	561899.63*	57	477.24	**561168.21***	57	476.77
S98T10	(98,10)	1125295.96*	114	1040.2	1125398.21*	114	1071.79
S98T14	(98,14)	1574542.60*	159	1297.93	1573987.75*	159	1300.5

[a] Hybrid GA with binary representation.
[b] Hybrid GA with real representation.
[c] Modified hybrid GA for binary representation with randomly generated amount for $K = 40$.
[d] Modified hybrid GA for binary representation with fixed amount for $K = 40$.
+ The algorithm terminates in less than one hour.

The solutions in bold in Table 5.12 are the best of the 4 algorithms and an '*' shows that the solutions are better than the upper bound obtained by CPLEX. The mean and standard deviation of the total cost over the 10 runs are also shown in Table 5.13 below.

Table 5.13: The mean and standard deviation of the total costs over 10 runs

Data set	HGA1		HGA2		MHGA1		MHGA2	
	Avg. Obj.	Std. Dev.	Avg. Obj.	Std. Dev.	Avg. Obj.	Std. Dev.	Avg. Obj.	Std. Dev.
S12T5	2122.5	17.14	2129.24	12.5	21.22.57	14.19	2124.04	13.9
S12T10	4358.07	20.41	4403.37	43.77	4360.4	21.61	4355.35	22.09
S12T14	6150.06	24.12	6221.2	31.42	6151.08	16.5	6173.24	24.25
S20T5	3238.61	64.56	3222.87	33.05	3260.82	75.63	3284.2	65.24
S20T10	6674.98	60.46	6784.5	70.13	6660.13	72.34	6706.88	44.68
S20T14	9347.35	78.83	9659.57	88.47	9373.75	85.74	9354.89	59.89
S20T21	14160.6	115.51	14659.93	158.35	14136.47	92.08	14163	78.79
S50T5	5831.02	91.73	5686.09	48.22	5711.63	79.82	5773.86	65.98
S50T10	12059.41	97.6	12168.11	113.83	12076.03	73.97	12128.5	81.16
S50T14	17294.73	99.98	17652.97	153.28	17321.55	114.46	17337.33	113.44
S50T21	26678.06	127.75	27294.92	126.47	26625.07	117.25	26591.69	85.04
S98T5	563839.8	1464.09	5567351.21	3666.85	563271	1381.35	563741.31	1795.23
S98T10	1129545.27	3028.44	1141130.82	5452.87	1128558.3	2700.65	1128072.18	2334.86
S98T14	1580344.45	3273.84	1600426.66	4241.72	1580339.9	1260.47	1581465.51	5548.88

From Table 5.12, the CPLEX terminates prematurely in 3 of the data sets (indicated by '+') due to memory usage. CPLEX did not provide any solution for the 2 large problems (S98T10 and S98T14), even after 7200s (2h) of CPU time. In addition, it was not able to find the optimum solution in any of the instances within the 3600 s time limit. It is observed that the gaps, calculated as the ratio of the difference between the upper bound and the lower bound to the lower bound, for all the solutions obtained by CPLEX are more than 10%. This ratio grows drastically as the number of periods and the number of suppliers increase. Therefore it is very hard to judge the quality of the lower bound obtained by CPLEX as this may be because the lower bound is really loose or the upper bound is rather poor.

In small instances the upper bound found by CPLEX within the time limit outperforms the GAs results. Good solutions (the gap between the upper bound and the lower bound is less than 15%) are obtained in cases where the number of periods is 5. However, the best solutions for all our algorithms were found in significantly less CPU times. As expected, GA based algorithms performed relatively much better for larger instances. In addition, in almost all problem instances except for the S12T5 data set, the binary representation produced better solutions than the ones generated using the real representation. The solutions obtained by the binary representation HGA1 and the modified algorithm (MHGA1 and MHGA2) are not significantly different from each other, with the modified algorithms outperforming HGA1 slightly on 5 instances (2 by MHGA1 and 3 by MHGA2). All the algorithms produced significantly good solutions as the gap between the best solution and the lower bound for S98T5 is less than 3.5% besides requiring a computational time which is less than 30 minutes (1800 s). Table 5.13 shows that the standard deviations over the 10 runs for the four algorithms are comparatively small except for the larger instances that can be due to the maximum number of generations being not sufficiently large.

5.6 Conclusion

It is observed that in HGA, most of the last vehicle only utilize less than half of the vehicle capacity. This unnecessarily causes an increment in the number of vehicle used and consequently led to the additional transportation costs. This chapter proposes a new formulation called Modified Hybrid Genetic Algorithms (MHGA) to tackle this issue. In the second part of this chapter, 2-opt is used to reroute the cluster. And lastly to find the lower bound for each dataset, CPLEX is used.

With the increase of data size especially for the problems with 98 suppliers, the performance of the GA based algorithms increases and the results were obtained in significantly less computational times. In these particular instances, the suppliers are closely located which is most appropriate for consolidated transportation strategy.

The current algorithms do not incorporate powerful route improvement procedure though a simple 2-opt procedure is implemented at the end. The current algorithms give more emphasis on the benefit of consolidating the transportation to reduce the overall transportation cost and the inventory cost. However, reducing the routing cost can significantly reduce the total cost as it constitutes a large part of the total cost. It may therefore be interesting to dynamically use post-optimization at various generations and on specific chromosomes. This adaptive strategy is worth exploring further. The studied problem and the developed GA based heuristics can also provide interesting insights for solving other problems, especially in an outbound logistics where the demand pattern from one period to the other changes significantly.

CHAPTER 6

CONCLUSION AND FUTURE RESEARCH

6.0 Conclusion

In this thesis, we have focused on the methods to solve the integration of inventory control and distribution management. We designed three algorithms based on Genetic Algorithm to solve Inventory Routing Problem (IRP). The algorithms are coded into C++ programming language and integrated with Genetic Algorithm Library (GAlib).

Chapter 1 briefly introduced Inventory Routing Problem, background of the problem, problem statement, scopes and objective of this study. The problem addressed in this study is based on a finite horizon, multi-period, multi-supplier, single warehouse, where a fleet of capacitated vehicles, housed at a depot, transports (collects) products to (from) the customers (suppliers) to meet the demand specified by the assembly plant (suppliers) for each period. The inventory holding costs are incurred at the assembly plant which also acts as the warehouse in the one-to-many (outbound) network. The holding cost at the suppliers is not taken into consideration in our models. The vehicles return to the depot at the end of the trip. No backordering/backlogging is allowed in this study. However, if the demand for more than one period is collected, then the inventory is carried forward subject to product-specific holding cost incurred at the assembly plant.

Each of the methods in this study considers various logistics network that is one-to-many (outbound) network and many-to-one (inbound) network. Chapter 2 explains and describes these logistic networks. For each type of the logistics network, we consider split inventory and un-split inventory cases. In reality, outbound and inbound network occurs simultaneously in an organization. However, the costs involved in these networks are different because of the nature of the material carried in the network. In the inbound network, the product (such as raw, unfinished product, spare parts, assembles) is moved into a firm and not away from it. The network design here does not require sophisticated transportation or warehouse system. On the other hand, outbound logistic network is a procedure that is related with the movement and storage of finished goods from the production line to the end user. This system certainly requires proper warehouse, transportation, materials handling and inventory control. Nonetheless, in this study, we use the same costs for both logistics network in order for comparison later. The mathematical formulation for IRP is given by the end of this chapter.

Chapter 3 contains literature review on metaheuristics. Metaheuristic now are becoming quite popular because it is a heuristic method that is applied in problems with no satisfactory solution such as combinatorial optimization. Among the metaheuristics used to solve IRP are Tabu Search (TS), Genetic Algorithms (GA), Simulated Annealing (SA) and Variable Neighborhood Search (VNS) algorithm. Unlike the classical heuristics, metaheuristics will not stop at the local optima. Instead, it will continue to explore the search space for more possible solutions. In this study, we designed algorithms based on Genetic Algorithms to solve IRP. One of the advantages of using GA is that GA search from a set of solution which is different than other metaheuristics such as Simulated Annealing and Tabu Search that start with a single solutions and move to another solution by some transition. GA will do a multidirectional search in the solution space and reducing the probability of finishing in a local optimum. GA also only require objective function values and not continuous searching space or existence of derivatives. This is more relevant because most real life examples generally have discontinuous search spaces. Lastly, GA is nondeterministics which make them more robust.

In Chapter 4, we designed algorithms based on Genetic Algorithm with binary and real-valued representation. Each representation is designed for the inbound and outbound Inventory Routing Problem by considering various logistics conditions. The first method is called Hybrid Genetic Algorithm (HGA) that uses the classical binary matrix representation to represent the delivery or collection matrix. In HGA, we employ a two dimensional uniform crossover that is modified to suit the matrix representation where a binary mask of size $N \times T$ is generated randomly for each pair of parents. Next, the chromosome has to go through the mutation process. In this study, we adopt the flip bit mutation operator.

The second method discussed in Chapter 4 is Knowledge-based Genetic Algorithm (KBGA) where the chromosomes are represented by the real-valued integer matrix that encodes the delivery (collection) matrix. There will be a pre-processing procedure to generate the initial real-valued integer matrix. This procedure will use a combination of a random binary representation and the demand matrix In KBGA, new crossover operator is proposed for this method to tackle the real-valued chromosome. It is based on exchanging the delivery schedules for a selected set of periods, which is chosen randomly between the two parents. At the same time, it will ensure that the resultant child does not violate either the demand or the vehicle's capacity constraints. From the observation, a slightly higher inventory holding costs are produced from this method. Hence, a new mutation operator has been designed to overcome this problem where it will transfer some amount of the product picked up/delivered in the previous period to the current selected period.

For testing purposes, we expanded the original 4 datasets that were downloaded from http://mie.utoronto.ca/labs/ilr/IRP to 11 more datasets. The data expansion was done by varying the number of periods to represent small, medium and large size problems. The same 4 original datasets have been used by Lee et.al [1] in their work. The original 4 datasets are S12T14, S20T14, S50T21 and S98T14 that comprises of (12 customers/suppliers, 14 periods), (20 customers/suppliers, 21 periods), (50 customers/suppliers, 21 periods) and (98 customers/suppliers, 14 periods) respectively. Note that SNTt refers to an instant with N customers/suppliers and t periods. Double Sweep Algorithms are used to do the clustering and routing of the customers/suppliers for both HGA and KBGA methods.

Generally for HGA method, most split delivery problems give slightly better results compared to the un-split delivery. However, the difference between these two types of inventory can be considered as small to be noticeable. It is also observed that for large instances such as S98T14, the algorithm is quick to find the lower objective value. However, in order to converge into the optimal objective value, the algorithm needs to have more number of generations. In KBGA, most un-split problems seem to have lower total costs compared to split delivery problems. However, this situation changes when the size of the instances increases. An observation on the standard deviation in KBGA shows that there is no consistency in the results spread. By comparing these two methods, HGA seems to perform better and give steady solutions compared to KBGA. Nonetheless, KBGA has more potential in finding better solution as it allows the algorithm to exploit the tradeoff between transportation and inventory holding costs.

In the first two methods above, it is observed that the last vehicle in each period utilizes less than half of the vehicle's capacity. This has increased the number of vehicle used unnecessarily and indirectly increases the transportation costs. Chapter 5 discusses the mechanism to overcome this problem which is also our second objective in this study. We proposed an inventory updating mechanism called the Modified Hybrid Genetic Algorithms (MHGA) where it coordinates the vehicle in order to maximize the vehicle utilization. We found out that for most datasets using MHGA, there are some improvement compared to the results with HGA. As an extension, some post-optimization using 2-opt has been done on the results from HGA. This post optimization is used to reroute the cluster. The results after post-optimization are then compared with results using Variable Neighborhood Search (VNS) algorithm on the same datasets. The results show that post-optimization gives better results on the large datasets such as S98T5, S98T10 and S98T14.

Chapter 5 also discusses the third objective of this study that is to propose a new reformulation of the IRP model in order to reduce the complexity of the problem. This can be achieved when the dimension of the problem is reduced by removing the route length constraint from the formulation. This new formulation is then solved using CPLEX to get the lower and upper bound for each dataset. In small instances the upper bound found by CPLEX within the time limit outperforms the GAs results. GA-based algorithms on the other hand, performed relatively much better for larger instances. The current algorithms give more emphasis on the benefit of consolidating the transportation to reduce the overall transportation cost and the inventory cost. However, reducing the routing cost can significantly reduce the total cost as it constitutes a large part of the total cost.

6.1 Future Research

Further improvement and extensions can be done in two basic ways. First, some modifications can be done on the procedure of this work in order to make the model more efficient and flexible. For example the current model can be extended by adding a post improvement at various generations and on specific chromosomes especially for the large instances dataset. This adaptive strategy is worth exploring further. The studied problem and developed GA based heuristics can also provide interesting insights for solving other problems, especially in an outbound logistics where the demand pattern from one period to the other changes significantly.

Secondly, for future research it may be useful to investigate the possibility of using or combining Genetic Algorithms with another local search heuristics such as Simulated Annealing, Tabu Search or Variable Neighborhood Search algorithms in the problems. In this study, an assumption to have no shortages has been set. An investigation can be done to see the effect if we relax the assumption. Eventhough it is assumed that by allowing the shortages, it will incur excessive cost but it is likely will reduce the transportation cost.

Third, the C++ problem can be enhanced by combining the program with CPLEX. For example, the routing part can be solved by C++ and the inventory assignment calculation can be solved in CPLEX. The combination of these two programming languages is expected to save the run time while at the same time giving the optimum solution to the problems.

REFERENCES

1. Lee, C.-G., Y.A. Bozer, and C.C. White III, *A heuristic approach and properties of optimal solutions to the dynamic inventory routing problems*, 2003, University of Toronto: Toronto, Canada.
2. Campbell , A.M. and M.W.P. Savelsbergh, *A decomposition approach for the inventory-routing problem.* Transportation Science, 2004. **38**(4): p. 488-502.
3. Ann, M.C., W.C. Lloyd, and W.P.S. Martin, *Inventory routing in practice*, in *The vehicle routing problem*. 2001, Society for Industrial and Applied Mathematics. p. 309-330.
4. Moin, N.H. and S. Salhi, *Inventory routing problems: a logistical overview.* Journal of Operational Research Society, 2007. **58**: p. 1185-1194.
5. Li, J.X., H.X. Chen, and F. Chu, *Performance Evaluation of Distribution Strategies for the Inventory Routing Problem.* . European Journal of Operational Research, 2008. **202**(2): p. 412-419.
6. Moin, N.H., et al., *Variable Neighborhood Search for the Multi Product Multi Period Inventory Routing Problem*, in *VIII Metaheuristic International Conference*2009: Hamburg.
7. Jung, J. and K. Mathur, *An efficient heuristic algorithm for a two-echelon joint inventory and routing problem.* Transportation Science, 2007. **41**(1): p. 55-73.
8. Li, J., F. Chu, and H. Chen, *Coordination of split deliveries in one-warehouse multi-retailer distribution systems.* Computers & Industrial Engineering, 2011. **60**: p. 291-301.
9. Yu, Y., H. Chen, and F. Chu, *A new model and hybrid approach for large scale inventory routing problems.* European Journal of Operational Research, 2008. **189**: p. 1022-1040.
10. Federgruen, A. and P. Zipkin, *A combined vehicle routing and inventory allocation problem.* Operations Research, 1984. **32**(5): p. 1019-1037.
11. Federgruen, A., G. Prastacos, and P.H. Zipkin, *An allocation and distribution model for perishable products.* Operations Research, 1986. **34**(1): p. 1986.
12. Chien, T.W., A. Balakrishnan, and R.T. Wong, *An integrated inventory allocation and vehicle routing problem.* Transportation Science, 1989. **23**(2): p. 67-76.

13. Fisher, M.L., et al., *Real-time scheduling of a bulk delivery fleet: practical application of Lagrangean relaxation*, in *Technical Report*1982, Department of Decision Sciences, The Wharton School, University of Pennsylvania.
14. Dror, M., M.O. Ball, and B. Golden, *Computational comparison of algorithms for the inventory routing problem.* Annals of Operations Research, 1985. **4**(1): p. 3-23.
15. Dror, M. and M.O. Ball, *Inventory/routing:Reduction from an annual to a short-period problem.* Naval Research Logistics, 1987. **34**(6): p. 891-905.
16. Dror, M. and L. Levy, *A vehicle routing improvement algorithm comparison of a 'greedy' and a matching implementation for inventory routing.* Computers & Operations Research, 1986. **13**(1): p. 33-45.
17. Anily, S. and A. Federgruen, *One warehouse multiple retailer systems with vehicle routing costs.* Management Science, 1990. **36**(1): p. 92-114.
18. Anily, S. and A. Federgruen, *A class of Euclidean routing problems with general route cost functions.* Mathematics of Operations Research, 1990. **15**: p. 269-285.
19. Anily, S. and A. Federgruen, *Two-echelon distribution systems with vehicle routing costs and central inventories.* Operations Research, 1993. **41**(1): p. 37-47.
20. Bramel, J. and D. Simchi-Levi, *A location based heuristic for general routing problems.* Operations Research, 1995. **43**(4): p. 649-660.
21. Gendreau, M. and J.-Y. Potvin, *Handbook of Metaheuristics*. International Series in Operations Research & Management Science. Vol. 146. 2010, New York: Springer.
22. Maniezzo, V., T. Stützle, and S. Voß, *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*. 2010, New York: Springer.
23. Ribeiro, R. and H.R.D. Lourenço, *Inventory-routing mode, for a multi-period problem with stochastic and deterministic demand.* UPF Economics and Business Working Paper No. 725 2003.
24. Abdelmaguid, T.F., *Heuristic Approaches for the Integrated Inventory Distribution Problem*, 2004, University of Southern California: Los Angeles.
25. Abdelmaguid, T.F. and M.M. Dessouky, *A Genetic Algorithm Approach to the Integrated Inventory-Distribution Problem.* International Journal of Production Research, 2006. **44**: p. 4445-4464.

26. Abdelmaguid, T.F., M.M. Dessouky, and F. Ordóñez, *Heuristic Approaches for the Inventory-Routing Problem with Backlogging.* Computers & Industrial Engineering, 2009. **56**(4): p. 1519-1534.
27. Savelsbergh, M.W.P. and J.-h. Song, *An optimization algorithm for the inventory routing problem with continuous moves.* Computers and Operations Research, 2008. **35**(7): p. 2266-2282.
28. Michel, S. and F. Vanderbeck, *A Column-Generation Based Tactical Planning Method for Inventory Routing.* Operations Research, 2012. **60**(2): p. 382-379.
29. Popović, D., M. Vidović, and G. Radivojević, *Variable Neighborhood Search heuristic for the Inventory Routing Problem in fuel delivery.* Expert Systems with Applications, 2012. **39**(18): p. 13390-13398.
30. Coelho, L.C. and G. Laporte, *A Brand-and-cut Algorithm for the Multi-product Multi-vehicle Inventory-Routing Problem.* International Journal of Production Research, 2013.
31. Waller, M., M.E. Johnson, and T. Davis, *Vendor-managed inventory in the retail supply chain.* Journal of Business Logistics, 1999. **20**(1): p. 183-203.
32. Sari, K., *Exploring the benefits of vendor managed inventory.* International Journal of Physical Distribution & Logistics Management, 2007. **37**(7): p. 529-545.
33. Dror, M. and P. Trudeau, *Split delivery routing.* Naval Research Logistics, 1990. **37**: p. 383-402.
34. Zhao, Q.-H., S.-Y. Wang, and K.K. Lai, *A partition approach to the inventory/routing problem.* European Journal of Operational Research, 2007. **177**(2): p. 786-802.
35. Blum, C. and A. Roli, *Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison.* ACM Computing Surveys, 2003. **35**(3): p. 268-308.
36. Glover, F., *Future Paths for Integer Programming and Links to Artificial Intelligence.* Computers and Operations Research, 1986. **13**: p. 533-549.
37. Back, T., D.B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. 1997, Oxford University: Computational Intelligence Library.
38. Eiben, A.E. and J.E. Smith, *Introduction to Evolutionary Computing*. 2003, Berlin: Springer.

39. Wikipedia. *Evolutionary algorithm - Wikipedia, the free encyclopedia*. 2011 [cited 2011 27 June]; Available from: http://en.wikipedia.org/wiki/Evolutionary_algorithm.
40. Cousineau-Ouimet, K. *A Tabu Search Heuristic for the Inventory Routing Problem*. in *37th Annual ORSNZ Conference*. 2002. University of Auckland.
41. Sindhuchao, S., et al., *An Integrated Inventory-Routing System for Multi-item Joint Replenishment with Limited Vehicle Capacity.* Journal of Global Optimization, 2005. **32**: p. 93-118.
42. Gillett, B.E. and L.R. Miller, *A Heuristic Algorithm for the Vehicle-Dispatch Problem.* Operations Research, 1974. **22**(2): p. 340-349.
43. Croes, G.A., *A Method For Solving Traveling-Salesman Problems.* Operations Research, 1958. **6**(6): p. 791-812.
44. Gendreau, M., A. Hertz, and G. Laporte, *New Insertion and Postoptimization Procedures for the Traveling Salesman Problem.* Operations Research, 1992. **40**(6): p. 1086-1094.