DESIGNING MULTI-OBJECTIVE REVERSE LOGISTICS NETWORKS USING GENETIC ALGORITHMS

by

SANYA YIMSIRI

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2009

Copyright © by SANYA YIMSIRI 2009

All Rights Reserved

ACKNOWLEDGEMENTS

I would like to thank Dr. Rogers, my dissertation advisor, for giving me guidance, support and assistance during my Ph.D. study. I also would like to thank my committee members, Dr. Liles, Dr. Kim and Dr. Baker for their times and suggestions.

No words can describe my gratitude to my parents, Boonruam and Suthat Yimsiri for supporting, encouraging and raising me up.

I would also like to thank my wife, Oh, and my daughter, Minmin, for love and support in both good and bad times during this long Ph.D. journey. This dissertation would not have been completed without their supports and sacrifices.

I would like to express my gratitude to my parent-in-laws, grandmother-in-law, aunt Hua and uncle Kao for their assistance, and my two brothers for their advice.

Thanks to Jo An Weddle at Small Business Development Center (SBDC) for Enterprise Excellence who gave me advice, support and invaluable work experiences through graduate research assistantship. She will always be in my memory.

To all other people that their names are not mentioned here who contribute to this graduate study; I would like to thank them all.

April 16, 2009

ABSTRACT

DESIGNING MULTI-OBJECTIVE REVERSE LOGISTICS NETWORKS
USING GENETIC ALGORITHMS

Sanya Yimsiri, PhD.

The University of Texas at Arlington, 2009

Supervising Professor: K. Jamie Rogers

Reverse logistics (RL) involves management of activities that include collection, sort/storage, transportation, recovery, disposal and re-distribution. The product return process is more complicated than forward logistics due to presence of multiple reverse distribution channels, individualized returns with small quantities, extended order cycles associated with product exchanges and a variety of recovery and disposition options. Reverse logistics has been gaining interest from many sectors due to rising costs, environmental concern and tougher regulations. As a result, good reverse logistics network design can help business save costs and meet their bottom lines in this competitive global environment. Most of the previous research in reverse logistics network design has focused on minimizing total costs. However, focusing on minimizing total costs alone is not adequate as there are other environmental and economical factors that contribute to increasing total costs. Therefore, in real life, such design problems usually have multiple objectives to be satisfied. In this research, not only total costs but also total transportation are minimized using non-conventional optimization algorithms. Evolutionary Computation (EC)

has been gaining attention due to its effectiveness and robustness in searching for a set of non-inferior solutions for a multi-objective problem. Genetic Algorithms (GA) are considered the most well known class of EC. They are stochastic search techniques that mimic the natural evolution process, and do not require prior domain knowledge.

The purpose of this research has been to develop a technique to solve the reverse logistics network design problem with multiple objectives approach. The Pareto based Multi-objective Genetic Algorithm (MOGA) was used to obtain non-dominated solutions. A case study was conducted and sensitivity analysis was performed to compare robustness and stability between typical aggregation based multi-objective genetic algorithms and the Pareto based genetic algorithms developed in this research. The outcome shows that the Pareto based genetic algorithms technique provides an efficient design tool for the reverse logistics network design problem with multiple objectives.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview

Reverse logistics (RL) as a research topic has been increasingly interesting to many sectors due to environmental concern and regulations. Product design and logistics network design have to be not only economically feasible but also ecologically friendly. With improved product design paradigms such as Design for Disassembly, Design for Disposal and Post Mass Production Paradigms (Umeda, Nonomura et al. 2000), along with network designs and Life Cycle Assessment/analysis, firms can increasingly satisfy environmental requirements and regulations with lower cost. In addition, reverse logistics can reduce costs by reusing products, components and materials instead of simply disposing of them into landfills which negatively impacts the environment.

According to ReturnBuy (ReturnBuy.com 2000), the total value of returned merchandise was $62 billion in 1999, representing $10–$15 billion in losses to retailers in the United States, while the cost of handling these product returns was estimated to be $40 billion. This high cost has led some companies to consider improving the process of reverse logistics involving product returns thus creating opportunities for cost savings and improved customer services. These companies include internet retailers that have grown with increases in online sales, but were often overwhelmed by the amount of returns, scope and complexity of sending returned products back to distributors or manufacturers for credit. Return rates for online sales are significantly higher than traditional bricks-and-mortar retail sales, reaching 20–30% in certain categories. Even worse, Rogers and Tibben-Lembke (Rogers, Tibben-Lembke et al. 1999) reported that an average return rate for the magazine publishing industry was as high as 50%.

With rising costs of product returns and shrinking profit margins, the optimal handling of product returns can be a competitive advantage since firms can save a substantial amount of transportation, inventory, and warehousing costs associated with product returns. These firms also gain benefits from reuse, remanufacturing and recycling products associated with the reverse chain.

Shear et al. (Shear, Speh et al. 2003) noted that handling costs associated with product returns could reach $50 per item, and could be three times higher than costs in the forward chain. Poirier (Poirier 2003) observed that firms that operate an efficient supply chain network achieved 40% more cost savings, 33% more inventory reductions, and 44% higher customer services than those in the inefficient supply chain network. Therefore, a focus on reverse logistics is needed to allow firms to gain more competitive advantages for survival especially in difficult economic times.

This research focuses on the inherent complexity of the reverse logistics by designing a multi-objective reverse logistics network that satisfies both cost and environmental objectives. Total cost associated with the RL network is minimized along with a reduction in total transportation distance. It was originally planned to obtain a feasible solution to the RL network model by Ant Colony Optimization (ACO) methodology. However, during the research period, it was found that it would be more applicable to use Genetic Algorithm (GA) which has proven performance in dealing with multi-objective problems in forward supply chain logistics.

In Ant Colony Optimization, an artificial ant represents each agent. The main task of ants is to collaboratively find the optimal path on a graph using pheromone trails as a communication medium. Each ant uses probabilistic values, which are a function of pheromone and heuristic information, to determine the next move. An ant will deposit a pheromone on its selected path that links the present and the next node. Any path with a high concentration of pheromone will have high probability to be chosen by the following ants. Consequently, the optimal path will be generated.

In multi-objective optimization, Evolutionary Computation (EC) has been gaining attention due to its effectiveness and robustness in searching for a set of non-inferior solutions. Genetic Algorithms (GA) is considered the most well known class of EC. They are stochastic search techniques that mimic natural evolution process. Genetic algorithms deal with coding of the problem instead of decision variables. Therefore, they do not require prior domain knowledge, only the payoff information or fitness function.

The Multi-objective Genetic Algorithm (MOGA) based on Pareto optimality is proposed to solve the multi-objective reverse logistics network design problems which are considered to be one of the hard combinational optimization problems (COP).

1.2 Background

1.2.1 Reverse Logistics Definitions

What is reverse logistics?

There are many definitions of reverse logistics (RL), but in general reverse logistics stands for all operations related to the reuse of products and materials as opposed to the typical forward supply chain logistics that is concerned with movement from raw material to manufacturing to distribution to consumer. The management of these operations can be referred to as Product Recovery Management (PRM). PRM is concerned with the care for products and materials after they have been used. Some of these activities are, to some extent, similar to those occurring in case of internal returns of defective items due to quality problems in production processes. Reverse logistics refers to all logistic activities to collect, disassemble and process used products, product parts, and/or materials in order to ensure a environmentally friendly recovery (Revlog).

Definitions of reverse logistics (RL) have been proposed by various authors indicated as follow. The American Reverse Logistics Executive Council (Rogers, Tibben-Lembke et al. 1999) defined RL as “The process of planning, implementing, and controlling the efficient, cost-effective flow of raw materials, in-process inventory, finished goods, and related information

from the point of consumption to the point of origin for the purpose of recapturing value or proper disposal."

The European Working Group on Reverse Logistics, RevLog (Revlog), gives a definition of RL as "The process of planning, implementing and controlling flows of raw materials, in process inventory, and finished goods, from a manufacturing, distribution or use point to a point of recovery or point of proper disposal"

"Reverse logistics can be viewed as an evolution of traditional forward logistics in an environmental-conscious industry or due to other commercial drives; it encompass all the logistics activities and management functions necessary for reintroducing valued-objects, which have finished or are not suitable to perform their primary function any more, into certain recovery systems for either recapturing their value or proper disposal" (Bostel, Dejax et al. 2005).

In the past, manufacturers did not feel responsible for their products after consumer use. The bulk of used products were dumped or incinerated with considerable damage to the environment. Today, consumers and authorities expect manufacturers to reduce the waste generated by their products. Therefore waste management has received increasing attention. Lately, due to new waste management legislation especially in Germany, the emphasis has been shifting towards recovery, due to the high costs and environmental burdens of disposal. Firms become more and more responsible for collecting, dismantling and upgrading of used products and packaging materials. The main reasons to become active in reverse logistics are

1. Environmental laws that force firms to take back their products and take care of further treatment,
2. Economic benefits of using returned products in the production process instead of paying high disposal costs, and
3. Growing environmental consciousness of consumers. (Revlog)

Min, Ko et al (Min, Ko et al. 2006) has summarized differences between reverse and forward logistics as in the table 1.1.

Table 1.1 Comparison between reverse logistics and forward logistics (Min, Ko et al. 2006)

	Reverse logistics	Forward logistics
Quantity	Small quantities	Large quantities of standardized items
Information tracking	Combination of automated and manual information systems used to track items	Automated information systems used to track items
Order cycle time	Medium to long order cycle time	Short order cycle time
Product value	Moderate to low product value	High product value
Inventory control	Not focused	Focused
Priority	Low	High
Cost elements	More hidden	More transparent
Product flow	Two way ("push and pull")	One way ("pull")
Channel	More complex and diverse (multi-echelon)	Less complex (single or multi-echelon)

1.2.2 Reverse Logistics Processes

While the traditional forward logistics process involves material supply, production, distribution and consumption, Fleischmann (Fleischmann 2000) defined reverse logistics process as shown in figure 1.1

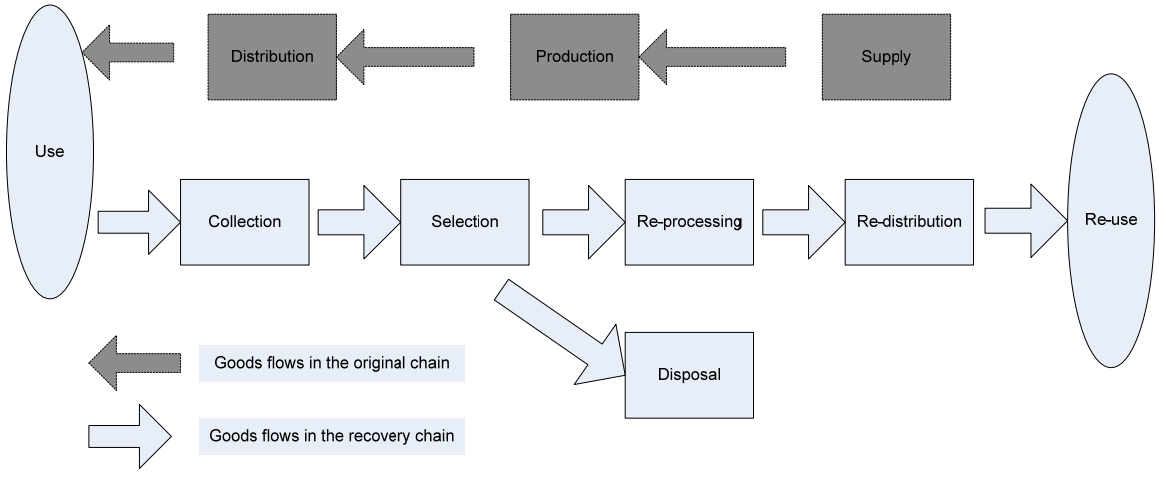

Figure 1.1 The recovery chain (Fleischmann 2000)

- Collection refers to all activities rendering used products available and physically moving them to some point for further treatment. Collection may include purchasing, transportation and storage activities.
- Inspection / separation denotes all operations determining whether a given product is in fact reusable and in which way. Thus, inspection and separation result in splitting the flow of used products according to re-use (and disposal) options. Inspection and separation may encompass disassembly, shredding, testing, sorting and storage steps.
- Re-processing means the actual transformation of a used product into a usable product/component/material again. This transformation may take different forms including recycling, repair and remanufacturing. In addition, activities such as cleaning, replacement and re-assembly may be involved.
- Disposal is required for products that cannot be re-used for technical or cost reasons. This applies, e.g., to product rejected at the separation level due to excessive repair requirements but also to products without satisfactory market potential, e.g., due to obsolescence. Disposal may include transportation, land filling and incineration steps.
- Re-distribution refers to directing re-usable products to a potential market and to physically moving them to future users. This may include sales, transportation and storage activities.

De Britto and Dekker (De Brito and Dekker 2002) proposed Recovery activities, which are related to re-processing, disposal and re-distribution processes above, as:

- Product Recovery: Product may be recycled directly into the original market or into secondary market, or repaired and sent back to the user under conditions of warranty,
- Component Recovery: products are dismantled and parts can be remanufactured into the same kind of product or different products,
- Material Recovery: materials are recuperated and recycled into raw materials like metal, paper or glass,

- Energy Recovery: products are incinerated to generate energy.

1.2.3 Closed Loop Logistics Network

A closed loop logistics network, comprised of both forward and reverse network, can be represented in the figure 1.2 forward flows are drawn in solid line while reverse flows are in dotted lines. Activities of a reverse logistics system include collection, cleaning, disassembly, testing, sorting, storage, transporting and recovery operations (Bostel, Dejax et al. 2005).

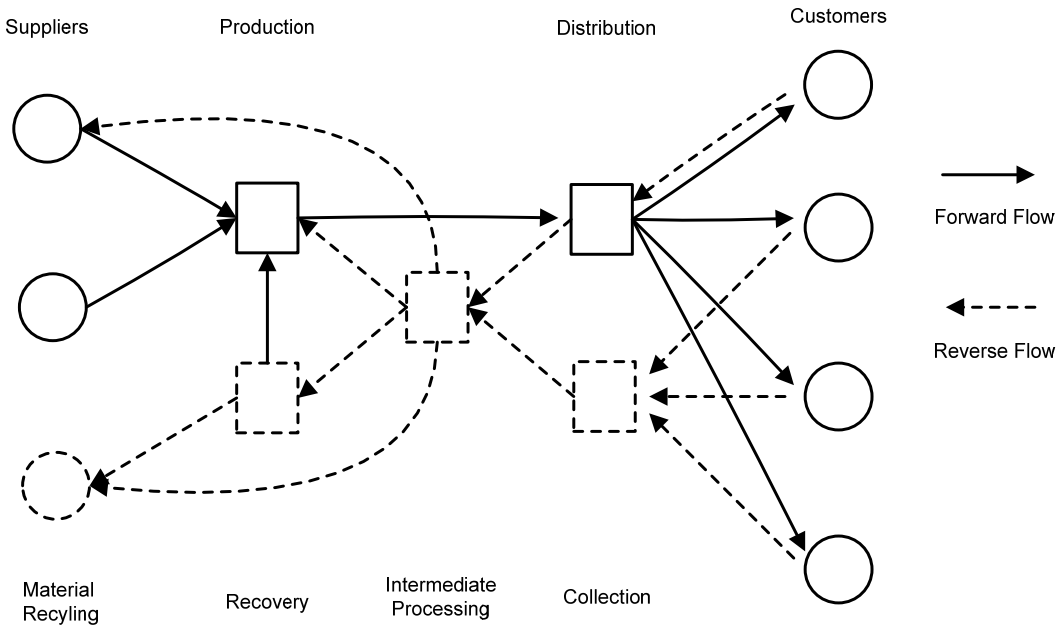

Figure 1.2 Generic forward and reverse logistics network (Bostel, Dejax et al. 2005)

The management of logistics networks and their activities is very complex because of their large dimensionality, wide variety of decisions of different scopes, focus and time horizon, and disturbance factor (Harhalakis, Nagi et al. 1993). However, the structure of decisions for such systems presents a hierarchy of interconnections (Anthony 1965). These decisions can be classified into three categories which are strategic planning, tactical planning and operational planning decisions. In this dissertation, we will focus on strategic planning aspect of RL.

Strategic planning is concerned with the design of the network, the establishment of managerial policies and the development of resources to satisfy external requirements in the way that complies with organization goals. Lu (Lu, Bostel et al. 2005) proposes a hierarchy framework for RL planning in the strategic planning as follow:

- Determination of the number and location of all types of logistics facilities such as production plants, warehouses, distribution centers in the forward logistics channel, and the corresponding facilities in the reverse channel, that are collection and sorting centers, recovery centers, etc.
- Determination of capacities and resources needed for all of above facilities
- Allocation of services areas to each facilities for distribution and collection activities

1.2.4 Reverse Logistics Network Design

The design and management of a reverse logistics network is more complex than that of forward logistics networks with direct flows. Two factors that cause these difficulties are:

- The simultaneous existence and mutual impact of the two types of flow: the possible coordination / integration and interfering constraints between forward and reverse flows must be considered;
- The existence of numerous uncertainties about the return flows such as choice of recovery options, quality of return objects, quantity and reprocessing time (Jayaraman, Guide Jr et al. 1999).

Reverse logistics systems can be classified into four major categories considering types of return items (Fleischmann, Bloemhof-Ruwaard et al. 1997) and the main options of recovery (Thierry 1993) These four classes are directly reusable network (DRN), remanufacturing network (RMN), repair service network (RSN) and recycling network (RN). Each network has their own characteristic as described below:

- Directly reusable network (DRN): This network involves new unopened returned items and reusable containers such as pallets, trays, boxes, and standard containers. They

can be directly reused without major operations on them. Only inspection, cleaning and minor maintenance are to be done. Forwards and reverse flows are closely associated. Thus, this system is closed loop.

- Remanufacturing networks (RMN): Products (or components) at the end of life or maintenance cycle are returned, and some parts or components are remanufactured to be used like new parts. This network is a closed loop system because remanufacturing is usually performed by the original manufacturer. The examples of this are copy machines and air craft engines.
- Repair service network (RSN): defective products, such as durable goods or electronics equipment, are returned and repaired in service centers. There are few links between FL and RL, so it is considered an opened-loop system.
- Recycling network (RN): raw materials, for example metal, glass and paper, are recycled usually by third parties recyclers. This can be considered an opened-loop system. This type of network also includes waste collection and elimination (Langevin and Riopel 2005).

Reverse logistics network design is a relatively new research field, so publications in this field are quite limited. The literatures can be divided in two major categories which are qualitative analysis based on case studies and quantitative analysis based on optimization models. In this case, we will focus on quantitative side and Pareto Based Genetic Algorithms will be used to obtained solutions.

Most of the studies found in literatures suggest extending classical facility location-allocation model to support the analysis of design problem of RL systems. Some literatures are devoted to specific application while others deal with generic models. The quantitative models can be summarized in the table below.

Table 1.2 Summary of quantitative models for reverse logistics systems
(adapted from Bostel, Dejax and Lu, 2005)

Authors	Model	Solution	Application
Crainic et al. (1989, 1993)	Deterministic uncapacitated MIP	Branch-and-bound	Container transport planning
Bloemhof-Ruwaad et al. (1996)	Deterministic capacitated MIP	Linear, Lagrangian relaxation	Breeding farm
Spengler et al. (1997)	Deterministic capacitated MIP	Standard package GAMS	Recycling of by-products in steel production
Barros et al. (1998)	Deterministic capacitated MIP	Linear relaxation + heuristics	Recycling sand
Marlin and Pelegrin (1998)	Deterministic uncapacitated MIP	Lagrangian decomposition based heuristics	Generic model
Jayaraman (1999)	Deterministic capacitated MIP	Standard package GAMS	Remanufacturing of electronics products
Realff et al. (1999)			
Fleischmann (2000)	Deterministic uncapacitated MIP	Standard package CPLEX	Copier remanufacturing and paper recycling
Shih (2001)	Deterministic capacitated MIP	Not indicated	Recycling electrical appliances
Luo et al. (2001)			
Lu (2003) in case of recycling	Deterministic capacitated and uncapacitated MIP	Lagragian heuristics	Generic model
Lu (2003) in case of repair service	Deterministic capacitated and uncapacitated MIP	Lagragian heuristics	Generic model
Lu et al. (2004) in case of direct use	Deterministic capacitated and uncapacitated MIP	Lagragian heuristics	Generic model
Lu (2003) in case of remanufacturing	Deterministic capacitated and uncapacitated MIP	Lagragian heuristics	Generic model
Kusumastuti (2005)	Deterministic capacitated MIP	Genetic algorithm	PC refurbishment

1.2.5 Reverse Logistics Network Design Model

Fleischman (Fleischmann 2000) noted that for most of the reverse logistics network models, facility location models based on mixed integer linear programming (MILP) have become a standard approach. There are models from simple un-capacitated plant location models to more complex capacitated multi-level multi-commodity models. A variety of solution algorithms proposed relied on combinatorial optimization theory. The author referred to

Michandani and Francis (1989) and Danskin (1995) for a detailed overview of models and solution techniques. As shown in the previous table, most of the models have a characteristics of single-period, multi-level, capacitated, discrete location models. The models have some similarity to facility location models. All aspects in production and distribution models can also be found in reverse logistics network models. However, production and distribution models are demand driven (pull) while reverse logistics network designs are not only pull driven but also supply pushed from either the disposer market or the reuse market or both. Therefore, it means that the network flows may be subjected to, possibly conflicting, constraints on the supply and on the demand side. The model can be closed loop or opened loop depending on the relationship between producers and customers.

Fleischman (Fleischmann 2000) concluded that current models for reverse logistics network design are quite similar to traditional facility location-allocation models, especially to the multi-level warehouse location models. The main differences are additional flow constraints reflecting supply restrictions concerning the disposer market. The models include supply-push constraints rather than entirely demand-pull driven. Other modifications include multiple return flow disposition and possible interaction between forward and reverse channel. As a result, most of the models have characteristics of multi-commodity flow. All models are deterministic and only treat uncertainty in a conventional way by using scenario and parametric analysis.

A general recovery network design model can be described as below:

Minimize:

$$\sum_{i\in I} f_i^p Y_i^p + \sum_{j\in J} f_j^w Y_j^w + \sum_{l\in L} f_l^r Y_l^r + \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} c_{ijk}^f d_k \, X_{ijk}^f$$

$$+ \sum_{k\in K}\sum_{l\in L}\sum_{i\in I_0} c_{kli}^r r_k \; X_{kli}^r + \sum_{k\in K} c_k^u d_k \, U_k \; + \sum_{k\in K} c_k^w r_k \; W_k$$

Subjected to:

$$\sum_{i\in I}\sum_{j\in J} X_{ijk}^{f} = 1 - U_k \qquad \forall k \in K \qquad (x.1)$$

$$\sum_{l\in L}\left(\sum_{i\in I} X_{kli}^{r} + X_{kl0}^{r}\right) = 1 - W_k \qquad \forall k \in K \qquad (x.2)$$

$$\gamma \sum_{i\in I_0} X_{kli}^{r} \le X_{kl0}^{r} \qquad \forall k \in K, l \in L \qquad (x.3)$$

$$\sum_{k\in K}\sum_{l\in L} r_k \; X_{kli}^{r} \le \sum_{j\in J}\sum_{k\in K} d_k \, X_{ijk}^{f} \qquad \forall i \in I \qquad (x.4)$$

$$\sum_{j\in J} X_{ijk}^{f} \le Y_i^{p} \qquad \forall i \in I, k \in K \qquad (x.5)$$

$$\sum_{i\in I} X_{ijk}^{f} \le Y_j^{w} \qquad \forall j \in J, k \in K \qquad (x.6)$$

$$\sum_{i\in I_0} X_{kli}^{r} \le Y_l^{r} \qquad \forall k \in K, l \in L \qquad (x.7)$$

$$Y_i^{p} Y_j^{w} Y_l^{r} \in \{0,1\} \qquad \forall i \in I, j \in J, l \in L \qquad (x.8)$$

$$0 \le X_{ijk}^{f}, X_{kli}^{r}, U_k \; W_k \; \le 1 \qquad \forall i \in I, j \in J, l \in L \qquad (x.9)$$

Where:

Index sets

$I = \{1, \dots, N_p\}$ *set of potential plant locations*

$I_0 = I \cup \{0\}$, *where* 0 *denotes the disposal option*

$J = \{1, \dots, N_w\}$ *set of potential warehouse locations*

$K = \{1, \dots, N_c\}$ *set of fixed customer locations in disposer and reuse market*

$L = \{1, \dots, N_r\}$ *set of potential disassembly locations*

Variables

$X_{ijk}^{f} = forward\ flow: fraction\ of\ demand\ of\ customer\ k\ to\ be\ served\ from\ plant\ i\ and\ warehouse\ j; i \in I, j \in J, k \in K$

$X_{kli}^{r} = reverse\ flow: fraction\ of\ returns\ from\ customer\ k\ to\ be\ returned\ via\ disassembly\ center\ l\ to\ plant\ i; k \in K, l \in L, i \in I_0$

$U_k = fraction\ of\ unsatisfied\ demand\ of\ customer\ k; k \in K$

$W_k = fraction\ of\ uncolllected\ returns\ of\ customer\ k; k \in K$

$Y_i^p = indicator\ opening\ plant\ i; i \in I$

$Y_j^w = indicator\ of\ opening\ warehouse\ j; j \in J$

$Y_l^r = indicator\ opening\ disassembly\ center\ l; l \in L$

Costs

$c_{ijk}^{f} = unit\ variable\ cost\ of\ serving\ demand\ of\ customer\ k\ from\ plant\ i\ and\ warehouse\ j, including\ transportation, production\ and\ handling\ cost; i \in I, j \in J, k \in K$

$c_{kli}^{r} = unit\ variable\ cost\ of\ returns\ from\ customer\ k\ via\ disassembly\ center\ l\ to\ plant\ i; including\ transportation\ and\ handling\ cost\ minus\ production\ cost\ saving\ at\ plant\ i; k \in K, l \in L, i \in I$

$c_{kl0}^{r} = unit\ variable\ cost\ of\ disposing\ returns\ from\ customer\ k\ via\ disassembly\ center\ l, including\ collection, transportation, handling\ and\ disposal\ cost; k \in K, l \in L$

$c_k^u = unit\ penalty\ cost\ for\ not\ serving\ demand\ of\ customer\ k; k \in K$

$c_k^w = unit\ penalty\ cost\ for\ not\ collecting\ returns\ of\ customer\ k; k \in K$

$f_i^p = fixed\ cost\ for\ opening\ plant\ i; i \in I$

$f_j^w = fixed\ cost\ for\ opening\ warehouse\ j; j \in J$

$f_l^r = fixed\ cost\ for\ opening\ disassembly\ center\ l; l \in L$

Parameters

$d_k = demand\ of\ customer\ k\ in\ the\ reuse\ market; k \in K$

$r_k = returns\ from\ customer\ k\ in\ the\ disposer\ market; k \in K$

$\gamma = minimum\ disposal\ fraction$

1.2.6 Objectives in Reverse Logistics Network Design

In previous literature, most of the reverse logistics network design models have only one objective minimizing total costs. Total costs include fixed costs and variable costs. However, environmental impact is becoming increasingly important. This is partly due to stakeholders demanding environmental improvement. (Azzone, Noci et al. 1996) Unfortunately, the intrinsic complexity associated with environmental issues means it is difficult to understand all feasible actions available to a firm for reducing its impact on the environment. Life Cycle Assessment (LCA) is regarded as one of the most important environmental management tools today. So far, supply chain models and LCA seem to live in separate worlds. (Krikke, Bloemhof-Ruwaard et al. 2003)

Table 1.3 Overview of model characteristics (Krikke, Bloemhof-Ruwaard et al. 2001)

Author	type of optimization	type of supply chain	decision variables
reverse supply chain cost models			
Spengler et al., 1997	Costs	reverse, open loop	location-allocation
Ossenbruggen and Ossenbruggen, 1992	Costs	reverse, open loop	Allocation
Pugh, 1993	Costs	reverse, open loop	Allocation
Barros et al., 1998	Costs	reverse, open loop	location-allocation
Gottinger, 1988	Costs	reverse, open loop	location-allocation
Krikke et al., 1998	Costs	reverse, closed loop	location-allocation
Krikke et al., 1999	costs	reverse, open loop	location-allocation
Louwers et al., 1997	costs	reverse, open loop	location-allocation
Ammons et al., 1997	costs	reverse, open loop	location-allocation
Marks, 1969	costs	reverse, open loop	location-allocation
Jaramayan et al., 1997	costs	reverse, open and closed loop	location-allocation

Table 1.3 – continued

closed loop supply chain cost models			
Berger and Debaille, 1997	costs	forward and reverse, closed loop	location-allocation
Kroon and Vrijens, 1995	costs	forward and reverse, closed loop	location-allocation
Thierry, 1997	costs	forward and reverse, closed loop	Allocation
LCA oriented models			
Sasse et al., 1999	Multi-objective with simple LCA (energy and waste)	reverse, open loop	location-allocation
Berger et al., 1998	flexible, cost and environmental	reverse, open loop	location-allocation
Caruso et al., 1993	multiobjective with environmental indicators	reverse, open loop	location-allocation
Bloemhof-Ruwaard, 1996	LCA	forward and reverse, closed loop	allocation and product mix
Daniel et al., 1999	LCA	reverse, open loop	product design
Guelorget et al., 1993	LCA	forward and reverse, open loop	product design

One of the approaches to model a multi-objective problem is to define an aggregated objective function as a weighted sum of the objectives. Single objective optimization algorithms can then be applied, without any changes to the algorithm, to find optimum solutions. For aggregation methods, the multi-objective problem is redefined as:

Minimize $\sum_{k=1}^{n_k} \omega_k * f_k(x)$

Subject to $g_m(x) \leq 0, \quad m = 1, \dots, n_g$

$h_m(x) = 0, \quad m = n_g + 1, \dots, n_g + n_h$

$X \in [X_{min}, X_{max}]^{n_x}$

$\omega_k \geq 0, \quad k = 1, \dots, n_k$

It is normally assumed that $\sum_{k=1}^{n_k} \omega_k = 1$. Parsopoulos and Vrahatis (Parsopoulos and Vrahatis 2002; Parsopoulos and Vrahatis 2002) successfully applied the aggregation method for Particle Swarming Optimization to a number of standard benchmarking functions. However, the aggregation approach has some problems that the weight values are problem dependent and difficult to obtain the best values.

The other approaches to multi-objective optimization problems were classified by Cohon and Marks (Cohon and Marks 1975) as A priori Preference Articulation (decide then search), A posteriori Preference Articulation (search then decide), and Progressive Preference Articulation (decide and search).

The main objective of multi-objective optimization has changed from single objective optimization. It is to find a set of solutions which optimally balances the trade-offs among objectives. Engelbrecht (Engelbrecht 2005) concluded approaches to multi-objective optimization into three categories as:

- Aggregation based, where the objective function is a weighted of the objective functions
- Criteria based, where the objective function calculation switches between different objectives in different stages of the optimization process
- Pareto dominance based, where objective function value is proportional to the dominance rank of solutions. Most dominance ranking schemes make use of an archive of non-dominating solutions.

These topics will be mentioned in detail again in Genetic Algorithm and multi-objective optimization section.

CHAPTER 2

LITERATURE REVIEW

2.1 Optimization

2.1.1 Meta-heuristics and hard combinatorial optimization problems

Most of the logistics problems have been known to have characteristics of being combinatorial. As a result, all possible combinations of the decisions and variables must be explored to find the optimum solution. The time required to solve the problem becomes extremely long as the number of variables increase to more than hundreds. Therefore, heuristics methods have been used to obtain reasonably good solution in realistic time. Heuristics methods explore only parts of the search space, concentrating in the parts that appear to promise a possible improvement of the solutions, thus reducing the time required to obtain a solution, which is often sub-optimal, but already a good improvement from the starting situation. A heuristic makes use of peculiar characteristics of a problem and exploits them to find a solution. Other empirical methods do not exploit only the problem characteristics but especially the analogy with other optimization methods found in Nature.

These heuristic methods are called meta-heuristics. Ant-Colony Optimization (ACO) is one of the examples of meta-heuristics. Ant Colony Optimization was originated by observing ants find the optimal path between a food source and their nest. An algorithm has been developed based on ants' behaviors. It has been applied to various problems, ranging from the travelling salesman problem, to the sequential ordering problem and the vehicle routing problem. Other well known meta-heuristic methods are Genetic Algorithms, Simulated Annealing, Tabu Search (IDSIA). Ant Colony Optimization algorithms have been implemented for commercial usage by companies such as Southwest Airlines and AntOptima.

2.1.2 Exact Algorithm

Exact algorithms are guaranteed to find the optimal solution and to prove its optimality for every finite set of instance of a combinatorial optimization problem within an instance-dependent run time. In the case of nondeterministic polynomial-time hard (NP-hard) problem, exact algorithms need exponential time to find the optimum in worst case. Some exact algorithms have been improved significantly to be able to obtain the optimum in timely manner. (Applegate, Bixby et al. 1995) However, for most NP-hard problems, the performance of exact algorithms is not satisfactory. For example, the quadratic assignment problem (QAP) is limited to around 30 in dimension when being solved by state-of-the art exact algorithms (Anstreicher, Brixius et al. 2002). Another example is ste36a from QAPLIB that took about 180 hrs of CPU time on a Pentium III, comparing to Ant Colony Optimization which requires an average time about 10 seconds to find the optimal solution for the same instance. Exact algorithms are also suffered from a strong rise in computation time when the problem size increases, rendered them infeasible. (Dorigo and Stutzle 2004)

2.1.3 Approximate / Heuristics Algorithm

Approximate algorithms seek to obtain good near-optimal solutions and require low computation. They are also called heuristics methods. Approximate algorithm can be classified as constructive and local search methods.

2.1.3.1 Constructive Algorithm

Constructive algorithms generate solutions from scratch by iteratively adding solution components to an initially empty solution until the solution is complete. (Dorigo and Socha 2006) An example of these algorithms can be demonstrated by applying Greedy algorithm to Travel Salesman Problem (TSP) by randomly choose the beginning city. Then, the next city, that has minimum distance from the current city, is repeatedly added until all cities in the network are explored. Constructive algorithms are faster than local search, but usually give lower quality solution than local search algorithms.

2.1.2.2 Local Search Methods

Local search begins with a complete solution, and tries to improve the quality of the current solution by local changes.

2.1.2.3 Meta-heuristic

Dorigo and Stutzle (Dorigo and Stutzle 2004) defined meta-heuristic as a set of algorithmic concepts that can be used to define heuristic methods applicable to a wide set of different problems. In other words, a meta-heuristic can be described as a general-purpose heuristic method designed to guide an underlying problem specific heuristic such as a local search algorithm or a construction heuristic towards promising regions of the search space containing high-quality solutions. A meta-heuristic is therefore a general algorithmic framework which can be applied to different optimization problems with relatively few modifications to make them adapted to a specific problem.

2.1.2.4 Model based methods

Most of the classic search methods may be considered instance-based, since they generate new candidate solutions using solely the current solution or the current “population” of solutions.(Dorigo and Stutzle 2004) Typical representatives of this class are evolutionary computation algorithms (Forgel et al., 1996; Holland, 1975; Rechenberg, 1973; Schwefel, 1981; Goldberg, 1989) or local search and its variants. The examples of these are simulated annealing and iterated local search.

2.1.3 Ant Colony Optimization Algorithm and its extensions

Miller (2007) noted in National Geographic that “A single ant or bee isn't smart, but their colonies are. The study of swarm intelligence is providing insights that can help humans manage complex systems, from truck routing to military robots.”

Table 2.1 A list of successful ant colony optimization algorithms
(Dorigo, Birattari, and Stutzle 2006)

ALGORITHM	AUTHORS	YEAR
ANT SYSTEM (AS)	DORIGO ET AL.	1991
ELITIST AS	DORIGO ET AL.	1992
ANT-Q	GAMBARDELLA & DORIGO	1995
ANT COLONY SYSTEM	DORIGO & GAMBARDELLA	1996
MAX -MIN AS	STUTZLE & HOOS	1996
RANK-BASED AS	BULLNHEIMER ET AL.	1997
ANTS	MANIEZZO	1999
BWAS	CORDON ET AL.	2000
HYPER-CUBE AS	BLUM ET AL.	2001

2.1.3.1 Ant Colony Optimization Algorithm Characteristics

Ant Colony Optimization algorithms are population based-stochastic search algorithms, designed to solve specific types of combinatorial optimization problems. These problems are generally characterized by:

- The search space is discrete
- A set of finite constraints
- A solution is comprised of an ordered sequence of components
- A cost function
- A finite set of components from which solutions are constructed
- A finite set of possible transitions among the components
- A finite set of sequences of components (Engelbrecht 2005)

Ant Colony Optimization and its variety extension are similar in structural procedures. However, they are differences in how a pheromone trail is updated within the algorithms.

Ant Colony Optimization procedures

procedure *ACO algorithm for static combinatorial problems*

Set parameters, initialize pheromone trails

while (termination condition not met) **do**

ConstructSolutions

ApplyLocalSearch (optional)

UpdateTrails

end

end

2.1.3.2 Ant System (AS), Max-Min Ant System (MMAS) and Ant Colony System (ACS)

There are several variants of Ant Colony Optimization algorithms in literatures. The three most successful ones are nt System (AS), Max-Min Ant System (MMAS) and Ant Colony System. Ant System was the first implementation of an Ant Colony Optimization algorithm, followed by Max-Min Ant System and Ant Colony System. Dorigo illustrated the differences between them by using the example of the Travel Salesman Problem as follow.

Ant System

Ant System was the first Ant Colony Optimization algorithm proposed by Dorigo (Dorigo, Maniezzo et al. 1996) in the literature. Its main characteristic is that the pheromone values are updated by all the ants that have completed the tour. The pheromone update for τ_{ij}, that is, for edge joining cities i and j , is performed as follows:

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

where ρ is the evaporation rate, m is the number of ants, and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone er unit length laid on edge (i, j) by the kth ant:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} \; if \; ant \; k \; used \; edge \; (i,j) in \; its \; tour \\ 0, otherwise \end{cases}$$

where Q is a constant anL_k is the tour length of the kth ant.

When constructing the solutions, the ants in AS traverse a construction graph and make a probabilistic decision at each vertex. The transition probability p_{ij}^{k} of the kth ant moving from city i to city j is given by

$$= \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l \in allowed_k} \tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}} \; if \; j \in allowed_k \\ 0, otherwise \end{cases}$$

where $allowed_k$ is the list of cities not yet visited by the kth ant, and α and β are the parameters that control the relative impotance of the pheromone versus the heuristic information η_{ij} given by

$$\eta_{ij} = \frac{1}{d_{ij}}$$

where d_{ij} is the length of edge (i, j).

Several implementations of the AS algorithm have been applied to different combinatorial optimization problems. They range from traveling salesman problem (TSP), quadratic assignment problem (QAP), job-shop scheduling problem (JSP), vehicle routing problem (VRP) and shortest common supersequence problem (SCS).

MAX-MIN Ant System

MAX-MIN Ant Systemis an improvement over the original AS idea. MMAS was proposed by Stutzle and Hoos (St¨utzle and Hoos 2000), who introduced a number of changes of which the most important are the following:

- only the best ant can update the pheromone trails, and
- the minimum and maximum values of the pheromone are limited.

The pheromone update takes the following new form:

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} + \Delta \tau_{ij}^{best}$$

where $\Delta\tau_{ij}^{best}$ best is the pheromone update value defined by

$$\Delta\tau_{ij}^{best} = \begin{cases} \dfrac{1}{L_{best}} \; if \; the \; best \; ant \; used \; edge \; (i,j) \; in \; its \; tour \\ 0, otherwise \end{cases}$$

L_{best} is the length of the tour of the best ant. This may be, subject to the algorithm designer decision, either the best tour found in the current iteration—iteration best, L_{ib}—or the best solution found since the start of the algorithm—best-so-far, L_{bs} —or a combination of both.

Concerning the limits on the minimal and maximal pheromone values allowed, respectively τ_{min} and τ_{max} , Stutzle and Hoos suggest that they should be chosen experimentally based on the problem at hand. The maximum value τ_{max} may be calculated analytically provided that the optimum ant tour length is known. In the case of the TSP, τ_{max} is given by

$$\tau_{max} = \frac{1}{\rho} \cdot \frac{1}{L^*}$$

Where L^* is the length of the optimal tour. If L^* is not known, it can be approximated by L_{bs}. The minimum pheromone value τ_{min} should be chosen with caution as it has a rather strong influence on the algorithm performance. St¨utzle and Hoos present an analytical approach to finding this value based on the probability p_{best} that an ant constructs the best tour found so far. This is done as follows. First, it is assumed that at each construction step an ant has a constant number k of options available. Therefore, the probability that an ant makes the right decision (i.e., the decision that belongs to the sequence of decisions leading to the construction of the best tour found so far) at each of n steps is given by $p_{dec} = \sqrt[n-1]{p_{best}}$.

The analytical formula they suggest for finding τ_{min} is

$$\tau_{min} = \frac{\tau_{max} \cdot (1 - p_{dec})}{k \cdot p_{dec}}$$

For some problems the choice of an appropriate τ_{min} value is more easily done experimentally than analytically.

The process of pheromone update in MMAS is concluded by verifying that all pheromone values are within the imposed limits:

$$\tau_{ij} = \begin{cases} \tau_{min}, & if\ \tau_{ij} < \tau_{min} \\ \tau_{ij}, if\ \tau_{min} \leq \tau_{ij} \leq \tau_{max} \\ \tau_{max}, & if\ \tau_{ij} > \tau_{max} \end{cases}$$

MAX-MIN Ant System provided a significant improvement over the basic AS performance. While the first implementations focused on the TSP, it has been later applied to

many other problems such as the QAP, the generalized assignment problem (GAP), and the set-covering problem (SCP.)

Ant Colony System

Another improvement over the original AS was Ant Colony System (ACS), introduced by Gambardella and Dorigo (Dorigo and Gambardella 1997). The most significant improvement of ACS is the introduction of a local pheromone update in addition to the pheromone update performed at the end of the construction process, called here offline pheromone update.

The local pheromone update is performed by all the ants after each construction step. Each ant applies it only to the last edge traversed:

$$\tau_{ij} = (1 - \varphi) \cdot \tau_{ij} + \varphi \cdot \tau_0$$

where $\phi \in (0, 1)$ is the pheromone decay coefficient, and τ_0 is the initial value of the pheromone.

The main goal of the local update is to diversify the search performed by subsequent ants during one iteration. In fact, decreasing the pheromone concentration on the edges as they are traversed during one iteration encourages subsequent ants to choose other edges and hence to produce different solutions. This makes less likely that several ants produce identical solutions during one iteration.

The offline pheromone update, similar to MMAS, is applied at the end of each iteration by only one ant, either the one that found the best solution in the iteration or the best-so-far. However, the update formula is slightly different:

$$\tau_{ij} \leftarrow \begin{cases} (1 - \rho) \cdot \tau_{ij} + \rho \cdot \Delta\tau_{ij} \; if \; edge \; (i,j) \; belongs \; to \; the \; tour \; of \; the \; best \; ant \\ \tau_{ij}, \;\; otherwise \end{cases}$$

and in case of the TSP, $\Delta\tau_{ij} = \frac{1}{L_{best}}$ .In the same manner as in MMAS, L_{best} can be set to either L_{ib} or L_{bs}.

Another important difference between AS and ACS is in the decision rule used by the ants during the construction process. Ants in ACS use the so-called “pseudorandom proportional rule:” The probability for an ant to move from city i to city j depends on a random

variable q uniformly distributed over [0, 1], and a parameterq_0; if $q \leq q_0$, then $j = argmax_{l \in N(s^p)}\{\tau_{il}\, \eta_{il}^{\beta}\}$, otherwise p_{ij}^{k} from AS is used.

Ant Colony System has been initially developed for the TSP, but it was later used to tackle various combinatorial optimization problems, including vehicle routing, sequential ordering, and timetabling problems(Dorigo and Stutzle 2004).

2.1.4 Ant Colony Optimization with local search

Meta-heuristic algorithms sometimes encounter the state in which it is impossible to improve the solution quality by itself. To solve this problem, various researches suggested that a promising approach to extract high-quality solutions is to employ a local search mechanism within the meta-heuristics framework. Local search algorithms operate on the solutions found by meta-heuristics by introducing some modification to obtain local optimal solutions. (St¨utzle and Hoos. 2005) Furthermore, the consideration of local search algorithm as a standalone technique for combinatorial optimization problems suffers from the problem of finding good starting solutions. As a result, coupling meta-heuristic algorithms such as Ant Colony Optimization with local search algorithms can be explained as follow. The meta-heuristic algorithms provide a high quality initial solution. Then, local search algorithms operate on these quality solutions to provide an even improved quality solution.

Ant Colony Optimization framework has the flexibility of coupling local search in the definition. Once ants complete their solution construction phase, the local search algorithms are allowed to refine their solutions to yield a higher quality solution. Afterwards, the pheromones are updated on the arcs with respect to the improved solutions found by local search procedures. Dorigo and Stutzle (Dorigo and Stutzle 2004) proposed that combining the ant's solution construction phase with the local search procedures is a promising approach and there is a very high possibility and probability that the local search procedures can improve the solution constructed by the ants.

2.1.5 Ant Colony Optimization for Multi-Objective Problems

Engelbrecht (Engelbrecht 2005) stated that using Ant Colony Optimization algorithms to solve multi-objective optimization problems (MOP) can be done as defining a single objective function using an aggregation approach where the single objective is a weighted sum of the sub-objectives, or by combining objectives using a fuzzy logic approach. However, due to the problems of aggregation methods, the behaviors of artificial ants can be modified such that their collective behavior results in finding a set of non-dominated solutions.

2.2 Life Cycle Assessment / Analysis (LCA)

LCA stands for Life Cycle Analysis or Assessment. It is a method that is used to measure the impact that an industry has on the environment. It is also known as eco-balance, environmental impact analysis or even cradle-to-grave analysis. LCA aims to evaluate the environmental burden associated with a product, process or activity by identifying and quantifying energy and materials used and wastes released to the environment; to assess the impact of energy and materials used and wastes released to the environment and to identify and evaluate opportunities to affect environmental improvements (SETAC 1993). LCA can be defined as an input–output analysis of resources or materials and energy requirements in each phase of the life cycle of a product. By definition, LCA only considers environmental issues. In reality, economic and technical issues cannot be ignored in any decision. Therefore, LCA should be seen in a broader context, as a tool that provides information on the product's environmental impacts for decision-making (Mietinen and Hamalainen 1997). LCA models for production, use and disposal of products have been used since the late 1960s. Well-known examples are the life cycle studies on cotton diapers versus paper diapers and porcelain tea sets versus plastic cups (Guinee et al. 1993). LCA has also been applied to waste management, proving that some waste management options, although optimal from an economic view, harm the environment rather than support sustainability (Rose 1994).

According to Azzone et al. (Azzone, Noci et al. 1996), indicators can be divided into qualitative and quantitative economic indicators and quantitative non-economic indicators. For a

company's effect on the environment, they suggest non-economic quantitative indicators focusing on the measurement of physical data, such as emissions, waste, energy and transportation. Given the complexity of LCA, it becomes regular practice to use these indicators instead of doing a full LCA. Energy use and residual waste provide an approximation, which, due to lower data requirements, is more practical, especially when it has to be combined with cost optimization (Emblemsvag and Bras 1999, Sasse et al. 1999, Umeda et al. 2000). Therefore, in this research, total transportation distance between facilities is assumed to be such indicator, and is included in one of the objective functions.

2.3 Waste Electrical and Electronic Equipment (WEEE) Returns

In the European Union (EU), waste from electrical and electronic equipment (WEEE) is now subject to regulation designed to prevent the disposal of such waste and to encourage prior treatment measures to minimize the amount of waste ultimately disposed. The objective of this regulation is to preserve, protect and improve the quality of the environment, protect human health, and utilize natural resources prudently. In particular, the EU WEEE Directive 2002/96/EC requires that Producers of electronic equipment be responsible for the collection, reuse, recycling and treatment of WEEE which the Producer places on the EU market after August 13, 2005. (www.sun.com)

2.4 Evolutionary Computation

After doing research in Ant Colony Optimization (ACO) techniques for multi-objective reverse logistic network design, it was found that there was not much published research in this field since Ant Colony optimization is rather a new technique. However, during that period, much published research in evolutionary computation relevant to multi-objective network design for logistic problem was found. In addition, Silva (Silva, Sousa et al. 2005) compared Ant Colony Optimization with Genetic Algorithm (GA) in logistic process applications, and found out that both algorithms performed equally well for logistics problems. Despite comparable performance, Genetic Algorithm is computationally faster than Ant Colony Optimization, and GA is also well suited for a multi-objective environment from its foundation. Therefore, the author decided to

use GA, a subcategory of Evolutionary computation which has many publications to prove that this technique is suit for multi-objective network design problems that is NP hard problem.

Evolutionary computation belongs in a sub category of computational intelligence involving combinatorial optimization problems. It has characteristics of iterative progress, population based, guided random search, parallel processing, and usually biologically inspired. Evolutionary computation uses computational models of evolutionary processes as a major element in the design and implementation of the system. There are variety of evolutionary computation models that simulate the evolution of individual structures via processes of selection and reproduction. These are usually called evolutionary algorithms. Evolutionary algorithms maintain a population of structures that evolve according to the rules of selection and other operators such as recombination and mutation. Each individual in the population is measured for its fitness in the environment. Individuals with high fitness are selected, and ones with low fitness are discarded to maintain the same population. Recombination and mutation of high fitness individuals provides general heuristics for exploration. The pseudo code of evolutionary algorithms is as follow:

```
Procedure Evolutionary Algorithm
        Time, t=0
        initialize population P(t)
        evaluate P(t)
        until(done){
                t=t+1
                parent selection P(t)
                recombine P(t)
                mutate P(t)
                evaluate P(t)
                survive P(t) }
```

A population of individual structures is initialized randomly and then evolved from generation to generation by repeated evaluation, selection, recombination, and mutation. The population size is normally constant on evolutionary algorithms.

Evolutionary computation techniques imitate the natural evolutionary principles into an algorithm that may be used to search for optimal or near optimal solutions to a problem. The main difference that distinguishes an evolutionary search algorithm from traditional heuristic algorithms is that it is population based. An evolutionary algorithm performs a directed search for solutions by modification of successive generations of a number of populations. Evolutionary search usually performs better than random search and is not susceptible to hill-climbing behavior found in gradient based search (Sumathi, Hamsapriya et al. 2008).

2.5 Genetic Algorithms (GA)

2.5.1 Overview of Genetic Algorithms

In the last decades, several Evolutionary Computation methodologies have emerged and gained popularity. These include evolutionary programming, evolution strategy, genetic programming and genetic algorithm. Genetic Algorithm was firstly introduced by J. H. Holland (1975) in 1975. The Genetic Algorithm has been applied to variety types of problems such as machine learning, optimization. Genetic Algorithm is a stochastic search techniques based on the process of natural selection and genetics. Genetic Algorithm is distinctive from conventional optimization techniques in the way that it is initialized by a set of random generated solutions called population. Each individual, i.e. one solution, in the population is called chromosome. A chromosome is subdivided into genes. A gene represents a single factor for a control factor. Each factor in the solution set corresponds to gene in the chromosome. The chromosome represents the genotype, i.e., raw genetic information. The phenotype is an expression of the chromosome in terms of an objective function as shown in figure 2.1

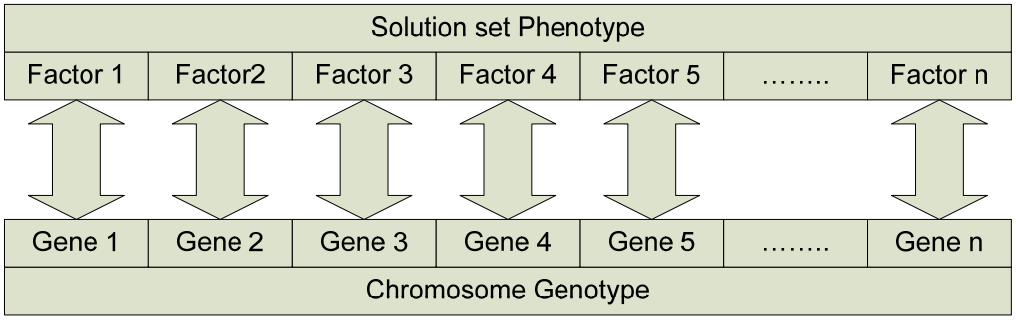

Figure 2.1 representation of genotype and phenotype of an individual chromosome adapted from Sivanandam and Deepa (Sivanandam and Deepa 2008)

A chromosome, in the final stage, will give out solutions to the objective function which is called fitness function in Genetic Algorithm. The chromosome is a string of variables that is usually, but not necessary, a binary string. The chromosome evolves through successive iterations which are called generations. During each generation, the chromosomes are evaluated their fitness. Some of the fittest chromosomes are selected to generate the next generation or offspring via recombination process.

The recombination process can be:

- Merging two chromosomes from the current generation using genetic crossover operator, or
- Modifying a chromosome using a genetic mutation operator

A new generation is created by selecting some of parents and offspring that have highest fitness. To keep the population size constant, some of individuals that have low fitness are discarded from the population. Fitter chromosomes have higher probability to be selected. After amount of generations, the algorithm converges and the chromosomes that have higher fitness value are obtained. These chromosomes in a final population represent the solutions to the objective function at hand. The figure below shows generic algorithm of Genetic Algorithm. P(t) is a population at time (t).

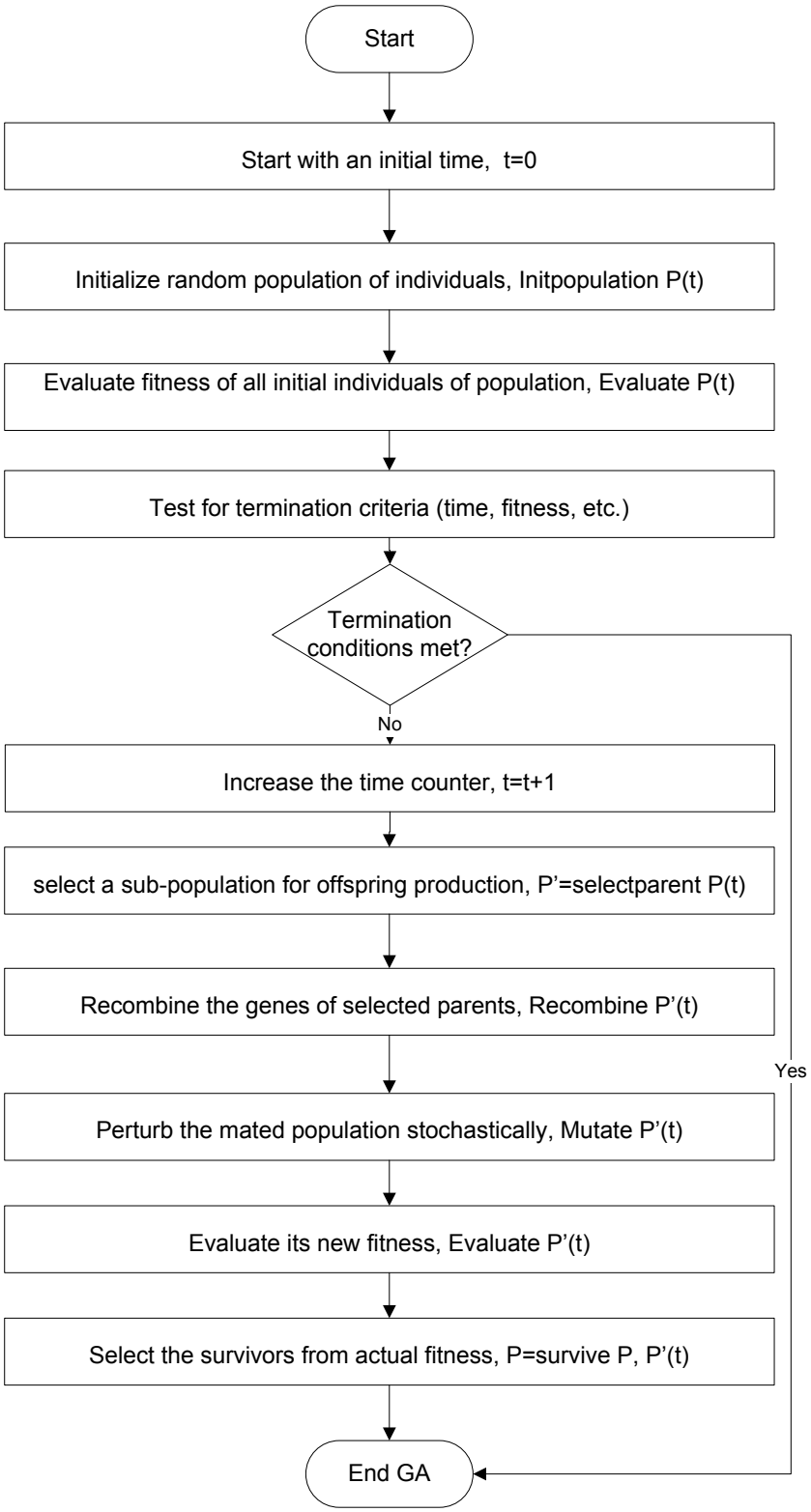

Figure 2.2 Algorithm of Genetic Algorithms

Genetic Algorithm then starts with initialization which is done by random generation, so it starts with large search space to make sure that it does not become stuck in a local sub-optimal point. Then, recombination typically involves crossover and mutation to yield the offspring. There are two types of operations in Genetic Algorithm which are:

- Genetic operations: involve crossover and mutation
- Evolution operation: involve selection

The genetic operations mimic the process of heredity of genes to create new offspring at each generation. The evolution operation imitates the process of Darwinian evolution to create populations from generation to generation.

Crossover is the main genetic operator. It operates on two chromosomes at a time and generates offspring by combining both chromosomes' features. A simple way to achieve crossover would be to choose a random cut point, and generate the offspring by combining the segment of one parent to the left of the cut point with the segment of the other parent to the right of the cut point. This method works well with bit string representation. The performance of Genetic Algorithm depends highly on the performance of crossover operator that is used.

The crossover rate is defined as the ratio of the number of offspring produced in each generation to the population size. This ratio controls the expected number of chromosomes to undergo crossover operation. A higher crossover rate allows wider exploration of solution space and reduces a chance of settling on local optimum. However, too high crossover rate can result in long computational time used to explore unpromising regions in the solution space.

Mutation is a background operator which produces spontaneous random changes in various chromosomes. Mutation represents new discovery in the new search space. It acts in the opposite way of crossover that is seen to move through the search space based on past information similar to memory in learning process. A simple way to achieve mutation would be to alter one or more of genes. In Genetic Algorithm, mutation serves crucial roles that are:

- Replace the genes that are lost from the population during the selection process so that they can be used again in a new combination

- Provide the genes that were not present in initial population

Mutation rate is defined as a percentage of the total number of genes that will be mutated in each generation. Mutation rate controls the rate at which new genes are introduced into the population for trial. If it is too low, the genes that would be useful are never been tried out. If it is too high, there will be too much perturbation, and offspring will lose characteristic that is inherited from parents. Therefore, the algorithm will lose the ability to learn from past information.

Selection is a process of choosing two parents from the population for crossing. In selection, the individuals are chosen to produce offspring. The purpose of selection is to emphasize fitter individuals in the population to produce offspring that have high probability to have higher fitness. The first step is fitness assignment. Each individual in the selection pool receives a reproduction probability depending on its own objective value and the objective of all other individuals in the selection pool. The higher fitness value, the more chance an individual has to be selected. This value is used for the actual selection step afterwards. There are two types of selection schemes, proportionate selection and ordinal-based selection. Proportionate-based selection picks out individuals based on their fitness values relative to the fitness of the other individuals in the population. On the other hand, ordinal-based selection selects individuals upon their rank, instead of fitness, in the population. (Sivanandam and Deepa 2008)

There are various selection methods including roulette wheel selection, random selection, rank selection, tournament selection, boltzmann selection and stochastic universal sampling, etc. A decision about which method of selection to be used for a specific application is one of the most important decision to be made. Selection is responsible for the speed of evolution. Inappropriate selection is often blamed where premature convergence stalls the success of genetic algorithm.

Fitness function in Genetic Algorithm is the value of the objective function for its phenotype. A phenotype is a decoded solution from a complete set of chromosomes called a

genotype. It should be designed to give graded and continuous feedback about how well an individual performs on the training set.

2.5.2 Differences between Genetic Algorithms and conventional optimization techniques

Genetic Algorithm differs from conventional optimization and search techniques in the following ways:

- Genetic Algorithm works with coding of solution set instead of the solutions themselves.
- Genetic Algorithm searches from population of solutions rather than a single solution found in conventional methods
- Genetic Algorithm uses fitness function for evaluation rather than derivatives or other auxiliary knowledge
- Genetic Algorithm uses probabilistic transition rules while conventional methods use deterministic transition rules

2.5.3 Advantages of Genetic Algorithms

There are main three main advantages when applying Genetic Algorithm to optimization problems which are:

- Genetic Algorithm does not have many mathematical requirements related to the optimization problems. Because of its evolutionary nature, Genetic Algorithm searches for solutions without any regard to the specific internal structure of the problem. Genetic Algorithm can handle any kind of objective function and any kind of constraint (e.g. linear vs nonlinear) defined on discrete, continuous or mixed search space.
- The ergodicity of evolution operators makes Genetic Algorithm very effective at performing a global search (in probability). The traditional approaches perform a local search by a convergent stepwise procedure, which compares the values of nearby points, and moves to the relative optimal points. Global optima can be found only if the problem possesses certain convexity properties which essentially guarantee that any local optimum is a global optimum.

- Genetic Algorithm provides us with a great flexibility to hybridize with domain dependent heuristics to make an efficient implementation for a specific problem (Gen 1997)

There are also additional advantages to the three main ones mentioned above as follows:

- Genetic Algorithm can scan thru solution sets quickly, and is not affected by bad proposals. Bad proposals are simply discarded by the algorithm.
- Genetic Algorithm is self inductive in nature, so it does not need to know any prior rules or data (domain knowledge). Genetic Algorithm works by its own internal rules. Therefore, Genetic Algorithm is good for complex or loosely defined problems.
- Genetic Algorithm searches problem space efficiently, so it is more likely to converge toward global optima.
- Genetic Algorithm can handle linear as well as non-linear problems
- Genetic Algorithm does not need to compute partial derivatives, so it saves some computational time
- Genetic Algorithm handles noisy search space better than stochastic hill climbing that sometimes get stuck in a local optimum.

2.6 Existing research in reverse logistic network design using Genetic Algorithm

Because reverse logistic design problems are combinatorial optimization problems, they usually be employed by non-conventional optimization algorithm such as Genetic Algorithm to obtain solutions. There are several publications that apply Genetic Algorithm to reverse logistic network design problem.

Genetic Algorithm has been applied to numerous supply chain management problems as the optimization approach in many different configurations. (Srin 2000)(Zhou 2002). Conway et al. (1994) demonstrated the use of Genetic Algorithm to solve a dynamic facility location problem. Neubauer (1995) applied Genetic Algorithm to production scheduling problems. Jeong et al. (2002) presented a Genetic Algorithm based system for implementing the forecasting activities required in supply chain management. For this research, a multiple objective Genetic

Algorithm with real number strings is used to derive solutions to a reverse logistics network problem.

Silva, Sousa et al (Silva, Sousa et al. 2005) discussed methodologies that can be used to optimize a logistic process like scheduling problem. They compared Genetic Algorithm, Ant Colony Optimization and classical dispatching rule in real world example. The results showed that Genetic Algorithm and Ant Colony Optimization outperformed classical dispatching policy. They also found that Genetic Algorithm and Ant Colony Optimization performed equally well in both solution quality and algorithm performance. However, Genetic Algorithm is slightly faster than Ant Colony Optimization in computational time.

Ant Colony Optimization and Genetic Algorithm are compared in logistic process optimization as follow:

- Genetic Algorithm and Ant Colony Optimization are robust algorithms in the sense that they always be able to provide good solutions.
- Genetic Algorithm and Ant Colony Optimization achieve comparable optimization results
- Genetic Algorithm is computational faster than Ant Colony Optimization

2.7 Multi-objective Optimization and Genetic Algorithms

The multi-objective Optimization Problem (MOP) is also called in other names such as multicriteria optimization, multiperformance or vector optimization problem. It can be defined as the problem of finding a vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Therefore, the term, “optimize,” means finding such a solution which would give the values of all objective functions acceptable to the decision maker (Osyczka 1985).

When dealing with real-life problems, especially in engineering design field, the optimal design cannot usually be expressed in terms of a single objective. In general, there is more than one objective to be satisfied in the design. Also, the objectives are usually in conflict in a multi-

objective model. For example, a decision to minimize fixed cost of opening facilities sometimes is in conflict with a decision to minimize cost of transportation, average service level, inventory holding cost, and so on. Therefore, there is no one solution exists that is optimal for all objectives. In this kind of problem, the notion of optimality is replaced by that of non-dominance or non-inferiority. A non-inferior solution is one in which an improvement in any one objective results in degradation of at least one of the other objective's values. Hence, a multi-objective model is used to generate various non-inferior solutions to the problem rather than to identify a single optimal solution (Jayaraman 1999).

To deal with a multi-objective optimization problem, it is common practice to combine multiple objectives to one objective. Then, a single objective optimization algorithm can be used to obtain the solution. This method is called aggregation method. It is done by translating multiple objectives into a single objective that is a convex combination of the original objective functions. This convex combination is determined by assigning relative weights to the original objectives and combining them. As mentioned before, the "good" weights are difficult to obtain without prior knowledge to the solutions. The other method is the constraint method. The constraints method identifies non-inferior solutions by optimizing one of the original objectives subjected to constraints on the value for the other objectives. Various non-inferior solutions are generated by varying the bounds on the other objectives. These approaches are less than ideal without prior knowledge how objectives interact with one another.

The last method to obtain a set of solutions for MOP is through the use of Pareto Optimal Theory. The multi-objective problems require a decision maker to make a choice of preferred solutions. The selection is essentially a tradeoff of one complete solution over another in multi-objective space. The definition of Pareto optimal in a minimization problem is that "x" is Pareto optimal if there exists no feasible vector x which would decrease some criterion without causing a simultaneous increase in at least one other criterion. The concept of Pareto Optimality is integral to the theory and the solving of MOPs (Coello Coello, Lamont et al. 2007).

In the other words, a solution can be considered Pareto optimal if there is no other solution that performs at least as well on every criteria and strictly better on at least one criteria. Thus, a Pareto-optimal solution cannot be improved upon without hurting at least one of the criteria. Solutions that are Pareto-optimal are also known in various literatures as nondominated, noninferior or Pareto-efficient. A solution is not Pareto-optimal if one criterion can be improved without degrading any others. This solution is known as a dominated or inferior solution.

Multi-objective optimization algorithms find these solutions by approximating the true Pareto optimal front that involves three objectives.

- Minimize the distance between solutions and the Pareto front
- Maximize the diversity of the non-dominated solutions to represent as much of the Pareto front as possible
- Maintain already found non-dominated solutions

Pareto optimality is named after an Italian economist, Vilfredo Pareto (1906). It is a measure of efficiency in multi-criteria situations. The concept has wide applicability in economics, game theory, multicriteria optimization, multicriteria decision-making, and the social sciences generally. Multi-objective problems are those in which there are two or more criteria measured in different units, and no agreed-upon conversion factor exists to convert all criteria into a single metric. In fig…, the first figure shows the mapping of decision parameter space into objective function space. The second figure shows the set of noninferior solutions lying on the curve between point C and D. Any point on the curve between C and is a noninferior solution point because an improvement in one objective requires a degradation in the other objective. Multi-objective optimization is concerned only with the generation and selection of noninferior solution points, i.e., Pareto optima. (Genetic Algorithm and Direct Search Toolbox 2, user's guide)

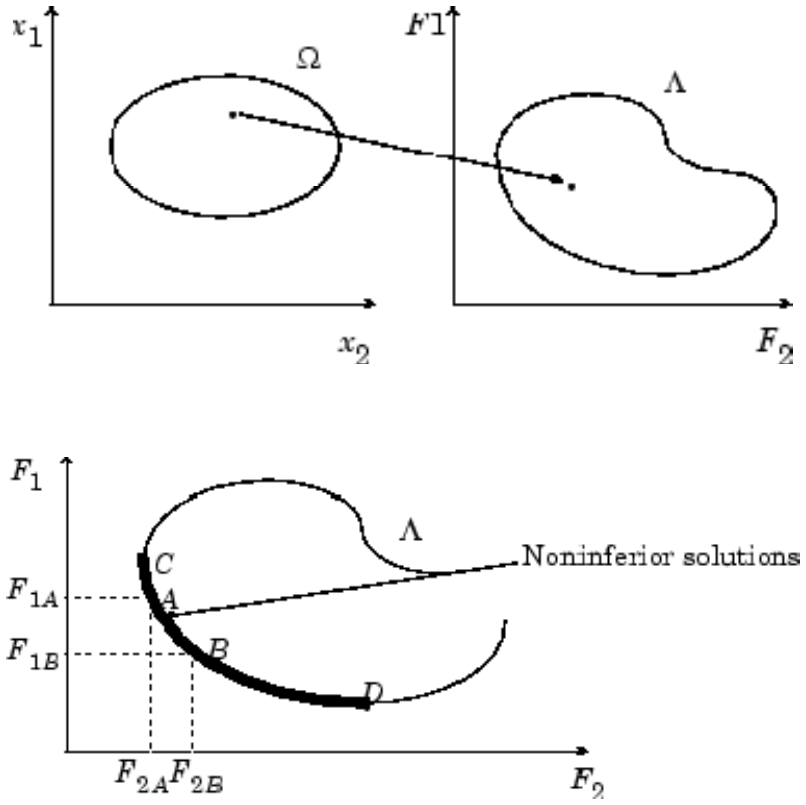

Figure 2.3 Pareto front plot showing noninferior solutions or Pareto optima
(Genetic Algorithm and Direct Search Toolbox 2)

Genetic Algorithms have recently become more widely used for their performance with large-scale, multi-objective problems. Genetic Algorithm is recognized as well suited to multi-objective optimization since their early development. Multiple individuals can search for multiple solutions in the same time, eventually taking advantage of any similarities available in the family of possible solutions to the problem. The ability to handle complex problems that involes features such as discontinuity, multimodality, disjoint feasible spaces and noisy function evaluation strengthen the potential effectiveness of Genetic Algorithm in multi-objective optimization. This is where Genetic Algorithm, including evolutionary computation, distinguishes itself from the competition (Sumathi, Hamsapriya et al. 2008).

CHAPTER 3

RESEARCH APPROACH AND METHODOLOGY

3.1 Research Objective

This research aims to develop a reverse logistics network design model that satisfies not only cost objective but also environmental requirements. Because of the hard combinatorial nature of reverse logistic network design problems together with multi-objective characteristics of real world optimization problems, the author decided to apply a nonconventional optimization method to obtain solutions. Ant Colony Optimization (ACO) was first explored. A few Ant Colony Optimization approaches exist that try to approximate the set of Pareto-optimal solutions (Dorigo and Stutzle 2004). Therefore, the other nonconventional techniques including Genetic Algorithm (GA) were surveyed. Finally, Genetic algorithm was selected instead of Ant Colony Optimization previously studied due its inherent multiple objective performance and available published researches at the time. Genetic Algorithm technique is used to obtain the solutions from a multi-objective reverse logistics network design model (MORLND) to reduce computational requirement of a traditional mixed integer linear programming (MILP) solver for NP-hard problems.

3.2 Research Plan

From the objectives above, the research plan is created to satisfy the objectives. The research plan explains steps needed to complete the dissertation. The steps are explained as follows.

1. Survey Literatures on reverse logistics Industry practices and Network Design
2. Survey literatures on reverse logistics optimization techniques such as Ant Colony Optimization and Genetic Algorithm
    a. Ant Colony Optimization Theories and algorithms

b. Ant Colony Optimization applications for logistics network design and optimization for both reverse and forward / Facility Location/allocation Problems

3. Survey literatures on Life Cycle Analysis, Environmental issues/regulations and trends.
4. Survey literatures on Evolutionary Computation, Genetic Algorithm and their applications
5. Survey literatures on Multi-objective Optimization Problem (MOP) and relevant algorithms used to solve MOP
6. Develop a Model for reverse logistic network design that satisfies multiple objectives including both cost and environmental requirements (transportation distance by unit in this case.)
7. Apply Genetic Algorithm to obtain solutions to the multi-objective reverse logistics network design (MORLND) model.
8. Case study with available data set, the data will be generated based on previous works
9. Sensitivity analysis on case study problems
10. Conclusion and future works

3.2 Research Methodology

1. Examine a reverse logistic network that is in consideration
2. Interview stake holders to find out what are the goals of the network design. For this model, the objectives need to be quantifiable such as minimizing total costs, transportation distance, service response time, etc
3. Define decision variables from the objectives
4. Explore cost components of RL network and its requirements
5. Explore distance between facilities in each echelon
6. Explore constraints of RL such as capacity, demand, operating duration and other requirements
7. Construct objective functions. In this case, there are two objectives to be minimized. The first objective is minimizing total cost. The second one is minimizing total

transportation distance (unit x distance). The total cost model is based on a general recovery network design model of Fleischmann (2000)

8. Run data through the Multi-objective Genetic Algorithms model (MOGA)
9. Evaluate results

3.3 Multi-objective Genetic Algorithm model (MOGA)

The Multi-objective Genetic Algorithm model attempts to create a set of Pareto optima for a multi-objective minimization. First, bounds and constraints on decision variables have to be defined. MOGA uses the genetic algorithm for finding local Pareto optima. As in the generic Genetic Algorithm, an initial population is randomly generated according to creation function specified by the users. The fitness function returns a vector of real number, integer or binary as defined in genetic representation.

3.3.1 Genetic representation

Genetic or chromosome representation is an important step in the design of Genetic Algorithm. Appropriate representation of candidate solutions greatly affects the efficiency and complexity of the search algorithm. In this model, vectors of real numbers are used to represent chromosomes. Each gene in the chromosome represents a solution to each decision variable. The chromosome length should be equal to possible transportation paths between facilities in each echelon in the reverse logistics network combined plus the other relevant decision variables.

3.3.2 Define fitness function

In the Darwinian model of evolution, individuals with the best characteristics have the best chance to survive and to reproduce. A mathematical function, namely fitness function, is used to quantify how good the solution represented by a chromosome is in order to determine the ability of an individual to survive. The genetic operators such as cross-over, mutation and selection make use of the fitness evaluation of the chromosomes. For example, selection operators are more likely to choose the most fit parents for cross-over while mutation is inclined towards the least fit individuals.

A fitness function of a multi-objective minimization problem is created in the form of a vector of functions. The functions represent each objective in terms of a vector $Z = \begin{bmatrix} Z_1 \\ Z_2 \\ \dots \\ Z_n \end{bmatrix}$, where $Z_1, Z_2, \dots, Z_n$ is objective functions.

In the methodology, there are two objectives to be satisfied. The first one is minimizing total costs associated with all operations in a reverse logistics network. The second one is minimizing total transportation distances by unit flows.

Total costs objective is adapted from Fleischman's recovery network design model. The objective function can be described as follow.

Minimize:

$$Z_1 = \sum_{i=1}^{I}\sum_{j=1}^{J} C_{ij}\, X_{ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} W_{jk}\, Y_{jk} + \sum_{i=1}^{I} F_i\, P_i + \sum_{j=1}^{J} G_j\, Q_j + \sum_{k=1}^{K} H_k\, R_k$$

Where:

Index sets:

$i = \{1, \dots, I\}$ set of potential retailer/ wholesaler locations

$j = \{1, \dots, J\}$ set of potential collection center locations

$k = \{1, \dots, K\}$ set of potential refurbishing facility locations

Variables:

$X_{ij} =$ quantity to be shipped from wholesaler/retailer i to collection center j

$Y_{jk} =$ quantity to be shipped from collection center j to refurbishing plant k

$C_{ij} =$ total variable costs per unit for satisfying collection center j by opening retailer/wholesaler i

$W_{kj} =$ total variable costs for satisfying refurbishing plant k by opening collection center j

$F_i =$ total costs of opening openning wholesaler/ retailer i

$P_i =$ indicator of opening or closing wholesaler/retailer $i, i \in \{0, 1\}$

G_j = total costs of opening collection center j

Q_j = indicator of opening or closing collection center $j, j \in \{0, 1\}$

H_k = total cost of openning refurbishing plant k

R_k = indicator of opening or closing refurbishing plant $k, k \in \{0, 1\}$

SP_i = quantity that is recalled to wholesaler/retailer i

RB_j = quantity that is to be refurbished at facility k

The second objective is minimizing total transportation distances by unit flows. This objective can be described as follows.

Minimize:

$$Z_2 = \sum_{i=1}^{I}\sum_{j=1}^{J} D_{ij}X_{ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} E_{jk}Y_{jk}$$

Where:

D_{ij} = transportation distance from wholesaler/retailer i to collection center j

E_{jk} = transportation distnace from collection center j to refurbishing plant k

3.3.3 Define constraints

Constraints of a multi-objective reverse logistic network design model can be defined as:

Supply of wholesaler/retailer constraints

$$\sum_{i=1}^{I}\sum_{j=1}^{J} C_{ij}X_{ij} \leq SP_i \,, for\ all\ 'i'$$

Flow conservation constraints

$$\sum_{i=1}^{I}\sum_{j=1}^{J} X_{ij} - \sum_{j=1}^{J}\sum_{k=1}^{K} Y_{jk} = 0 \,, for\ all\ i, j, k$$

Demand satisfaction for refurbishing plant constraints

$$\sum_{j=1}^{J}\sum_{k=1}^{K} Y_{jk} \leq R_j \ , for\ all\ j,k$$

Non-negativity constraints

$$X_{ij}, Y_{jk} \geq 0 \ , for\ all\ i,j,k$$

3.3.4 Define MOGA parameters

1. Population type

Population type can be real number, binary or mixed of both depending on chromosome representation. A real number population type is defined as “double”, i.e., double precision type.

2. Population size

Population size specifies how many chromosomes in each generation. For multi-objective Genetic Algorithm, it is normally set at 15 times of the length of the chromosome

3. Creation function

Creation function specifies the function that creates the initial population. For this constraint multi-objective optimization problem, feasible population is chosen to create a random initial population that satisfies the bounds and linear constraints.

4. Selection

The selection function chooses parents for the next generation based on their scaled values from the fitness functions. For multi-objective optimization problem, the tournament selection is chosen. It selects each parent by choosing individuals at random, the number of which can be specified in Tournament size, and then choosing the best individual out of that set to be a parent.

5. Reproduction

Reproduction parameters determine how the genetic algorithm creates children at each new generation. Reproduction parameters can be set through crossover fraction. Crossover fraction specifies the fraction of the next generation that crossover operation produces. Mutation

produces the remaining individuals in the next generation. Crossover fraction can be set between 0 and 1.

6. Mutation

Mutation functions make small random changes in the individuals in the population, which provide genetic diversity and enable the genetic algorithm to search a broader space. The mutation functions can be specified as Gaussian, uniform or adaptive feasible. In this model, adaptive feasible is used. Adaptive feasible randomly generates directions that are adaptive with respect to the last successful or unsuccessful generation. A step length is chosen along each direction so that linear constraints and bounds are satisfied

7. Crossover

Crossover combines two individuals, or parents, to form a new individual, or child, for the next generation. The crossover method can be chosen from scattered, single point, two point, intermediate, heuristic or arithmetic methods.

- Scattered method creates a random binary vector. Then, it selects the genes where the vector is a 1 from the first parent, and the genes where the vector is a 0 from the second parent, and combines the genes to form the child.
- Single point method chooses a random integer n between 1 and Number of variables, and selects the vector entries numbered less than or equal to n from the first parent, selects genes numbered greater than n from the second parent, and concatenates these entries to form the child.
- Two point method selects two random integers m and n between 1 and Number of variables. The algorithm selects genes numbered less than or equal to m from the first parent, selects genes numbered from m+1 to n from the second parent, and selects genes numbered greater than n from the first parent. The algorithm then concatenates these genes to form a single gene.

- Intermediate method creates children by a random weighted average of the parents. Intermediate crossover is controlled by a single parameter Ratio: child1 = parent1+ rand*Ratio*(parent2 - parent1). If Ratio is in the range [0,1], the children produced are within the hypercube defined by the parents locations at opposite vertices. If Ratio is in a larger range such as 1.1, children can be generated outside the hypercube. Ratio can be a scalar or a vector of length Number of variables. If Ratio is a scalar, all the children lie on the line between the parents. If Ratio is a vector, children can be any point within the hypercube.
- Heuristic method creates children that randomly lie on the line containing the two parents, a small distance away from the parent with the better fitness value, in the direction away from the parent with the worse fitness value.
- Arithmetic method creates children that are a random arithmetic mean of two parents, uniformly on the line between the parents.

  For this multi-objective objective problem, an intermediate crossover is used.

8. Migration

Migration is the movement of individuals between subpopulations, which the algorithm creates. The best individuals from one subpopulation occasionally replace the worst individuals in another subpopulation. These three parameters can control how migration occurs.

- Direction parameter specifies the direction in which migration can take place. It can be forward or both. For forward direction migration, the migration takes place toward the last subpopulation. That is the nth subpopulation migrates into the (n+1)th subpopulation. For both direction migration, the nth subpopulation migrates into both the (n–1)th and the (n+1)th subpopulation. Migration wraps at the ends of the subpopulations. That is, the last subpopulation migrates into the first, and the first may migrate into the last. To prevent wrapping, specify a subpopulation of size 0.

- Fraction parameter controls how many individuals move between subpopulations. Fraction is the fraction of the smaller of the two subpopulations that moves. If individuals migrate from a subpopulation of 50 individuals into a population of 100 individuals and Fraction is 0.1, 5 individuals (0.1 * 50) migrate. Individuals that migrate from one subpopulation to another are copied; they are not removed from the source subpopulation.
- Interval parameter controls how many generations pass between migrations. For example, if interval is set to 20, migration between subpopulations takes place every 20 generations.

9. Multi-objective problem settings

- Distance measure function is a measure of the concentration of the population. The distance crowding function is used to compute distance measure of individuals, computed in decision variable or design space (genotype) or in function space (phenotype)
- Pareto front population fraction keeps the most fit population down to the specified fraction in order to maintain a diverse population.

10. Termination criteria

Termination criteria determine what cause the algorithm to stop. They can be specified in terms of generation, time limit, fitness limit, stall generations, stall time limit, function tolerance or combination of them.

- Generations specify the maximum number of iterations the genetic algorithm performs.
- Time limit specifies the maximum time in seconds the genetic algorithm runs before stopping.
- Fitness limit; the algorithm stops if the best fitness value is less than or equal to the value of Fitness limit.

- Stall generations; the algorithm stops if the weighted average change in the fitness function value over Stall generations is less than Function tolerance.
- Stall time limit; the algorithm stops if there is no improvement in the best fitness value for an interval of time in seconds specified by Stall time limit.
- Function tolerance; the algorithm stops if the cumulative change in the fitness function value over Stall generations is less than Function tolerance.

3.3.5 Run multi objective Genetic Algorithm (MOGA) to obtain solutions

1. Run MOGA with parameters set as:
    - Population type: double vector
    - Population size: 15*number of genes
    - Creation function: feasible population creation function
    - Selection: tournament selection with tournament size = 2
    - Crossover fraction = 0.8, mutation fraction = 0.2
    - Mutation: adaptive feasible
    - Crossover: intermediate with crossover ratio of 1.0
    - Migration direction: forward with fraction of 0.2 and interval of 20
    - Distance measure function: distance crowding
    - Pareto front population fraction = 0.35
    - Termination criteria: 3000 generations, 100 stall generations or function tolerance of 10^{-4}
2. Perform population initialization

Initial population is generated by assigning a random value from the allowed domain to each of genes in chromosomes according to creation function defined in parameter setting. The generation is completed when a population size is reached. The population size remains constant throughout the algorithm.

3. Perform selection process

At the end of each generation, a new population of candidate solutions is selected to serve as the new population of the next generation. Tournament selection is used for this multi-objective problem. Tournament selection randomly picks out individuals from the population to form a sub group of population specified by tournament size. The scaled fitness of each individual in the subgroup is compared, and the best one is selected.

The new population is generated through cross-over, mutation and elitism operators. In crossover, the superior individuals have more opportunities to be chosen to reproduce to ensure that offspring contain genes from the best. In mutation, selection focuses on weak individuals in light that mutation will introduce better traits to increase the chance of survival. In elitism, the best individuals are selected and passed onto the next generation.

4. Perform reproduction process

    - Cross-over operation produces new offspring from two selected parents. Cross-over process creates a new individual by combining genetic material selected from parents. For this model, intermediate cross-over method is used as specified in parameter settings.
    - Mutation operation randomly changes the value of genes in a chromosome to increase genetic diversity. Adaptive feasible method is used as specified in parameter settings.

5. Evaluate fitness

Fitness of each individual in the new generation is calculated for the selection process for the next generation.

6. Terminate algorithm

Algorithm is repeated until one of termination conditions that are previously defined in parameter settings is met. They can be a combination of generations, time, stall generations, stall time and function tolerance.

3.3.6 Evaluate solutions

The non-dominated solutions are ranked by value of each objective function from low to high, so a decision maker can choose the solutions according to organization's goal.

3.3.7. Flowchart of MOGA

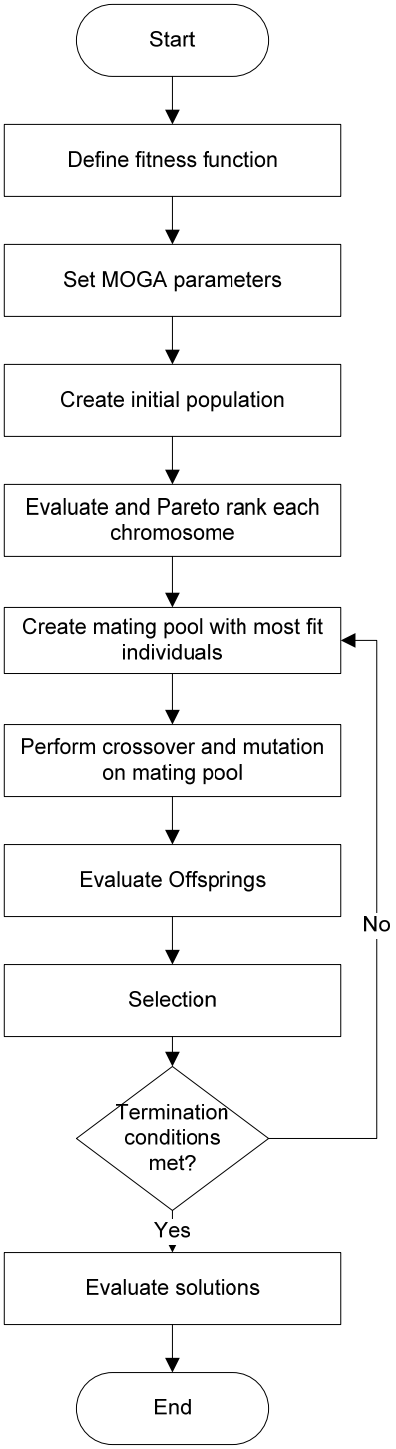

Figure 3.1 Flowchart of Pareto based Multi-objective Genetic Algorithms (MOGA)

CHAPTER 4

CASE STUDY AND DATA ANALYSIS

4.1 Case Study

The case study is based on the reverse distribution networks of previous works by Jayaraman, Patterson et al. and Kumanan, Venkatesan et al.

In Jayaraman, Patterson et al's model (Jayaraman, Patterson et al. 2003), the network is based on environmentally conscious manufacturing product recalls. Product recall is a reverse distribution activity that withdraws goods from consumers. The products are either hazardous, defective or have reached the end of their useful life. In particular, the paper examines product recall situations in which the customer returns the product to a retail store and the product is sent to a refurbishing site which will rework the product or dispose it properly. Costs of product recall through the reverse distribution channel are at least several times higher than costs incurred in forward distribution due to small quantities of shipments, demand uncertainty urgency and federal regulation involved in the recall process. The model objective focuses on total cost. Because such a model is NP-hard, the use of conventional Mixed Integer Programming (MIP) tools for solving problem is limited due to the complexity of the problem and the large number of variables and constraints. Therefore, a heuristic solution approach is employed to solve the problem.

Kumanan, Venkatesan et al. (Kumanan, Venkatesan et al. 2007) developed a supply chain logistics network model with the objective of minimizing the total cost of production and distribution. The model is comprised of multiple plants serving geographically dispersed customers and seeking to allocate demand for its products to the plants. The Genetic Algorithm (GA) and Particle Swarm (PS) search techniques are proposed for optimizing the supply chain logistics networks.

The reverse distribution network in this case study is consisted of three (3) retailers/wholesalers, three (3) collection facilities and two (2) refurbishing plants. A schematic of a reverse distribution network is shown in figure 4.1. It is a three echelon reverse logistic network with capacitated facilities.

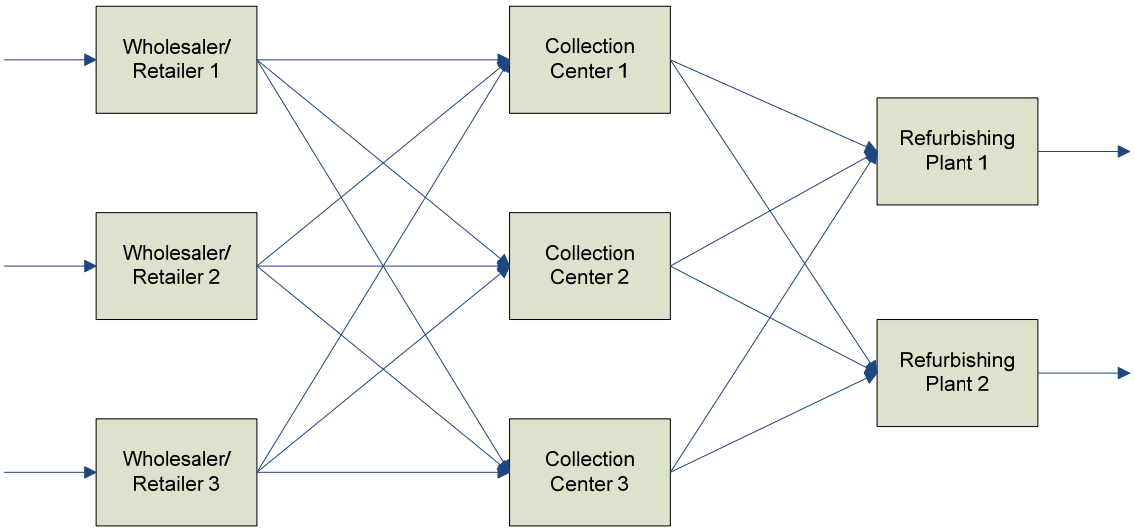

Figure 4.1 Capacitated three echelon reverse distribution network

The assumptions of the model are:

- Single product, multi-echelon network
- The product has been recalled and are to be recycled, disposed or hazardous
- Demand is deterministic and remain same throughout the period of study
- The capacity for collection centers is unlimited
- Transportation lead time is considered zero
- Total units for all collection centers must not exceed the units for retailers/wholesalers
- All units that enter collection centers must leave collection centers
- CO2 emission from transportation is dependent on the distance travelled

In this reverse distribution network, it is assumed that existing facilities are used, so there is no cost associated with opening new facilities. As a result, the cost components are total variable costs per unit from one facility to another facility. Total costs were randomly generated with uniform distribution, and the range is from $1 to $10 per unit. Total costs are shown in table 4.1 and 4.2. Distances between facilities are also shown in table 4.3 and 4.4 along with quantity recalled and demand of refurbishing plant in table 4.5 and 4.6

Table 4.1 Total costs per unit from retailers/wholesalers to collection centers

From/To	Collection Center		
Retailer/Wholesaler	1	2	3
1	8	4	6
2	1	10	2
3	6	8	7

Table 4.2 Total costs per unit from collection centers to refurbishing plants

From/To	Refurbishing Plant	
Collection Center	1	2
1	2	2
2	5	10
3	4	6

Table 4.3 Distance from retailer/wholesaler to collection center

From/To	Collection Center		
Retailer/Wholesaler	1	2	3
1	2	10	5
2	8	4	2
3	8	5	3

Table 4.4 Distance from collection center to refurbishing plant

From/To	Refurbishing Plant	
Collection Center	1	2
1	8	3
2	6	7
3	9	10

Table 4.5 quantity recalled for each wholesaler/ retailer

Wholesaler/retailer	Quantity recalled (units)
1	50000
2	100000
3	150000

Table 4.6 Demand of refurbishing plant

Refurbishing plant	Quantity (units)
1	200000
2	100000

The objectives of this network are to minimize total cost and minimize total transportation distance which eventually leads to minimizing carbon dioxide emission. The objective function can be formulated as:

Minimize:

$$Z_1 = \sum_{i=1}^{I}\sum_{j=1}^{J} C_{ij}X_{ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} W_{jk}Y_{jk}$$

$$Z_2 = \sum_{i=1}^{I}\sum_{j=1}^{J} D_{ij}X_{ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} E_{jk}Y_{jk}$$

Subjected to:

Supply of wholesaler/retailer constraints

$$\sum_{i=1}^{I}\sum_{j=1}^{J} C_{ij}X_{ij} \leq S_i \, , for\ all\ 'i'$$

Flow conservation constraints

$$\sum_{i=1}^{I}\sum_{j=1}^{J} X_{ij} - \sum_{j=1}^{J}\sum_{k=1}^{K} Y_{jk} = 0 \, , for\ all\ i, j, k$$

Demand satisfaction for refurbishing plant constraints

$$\sum_{j=1}^{J}\sum_{k=1}^{K} Y_{jk} \leq R_j \, , for\ all\ j, k$$

Non-negativity constraints

$$X_{ij}, Y_{jk} \geq 0 \, , for\ all\ i, j, k$$

Where:

$i =$ index number of wholesaler/ retailer, $i \in \{1, 2, 3, \dots I\}$

$j =$ index number of collection center, $j \in \{1, 2, 3, \dots J\}$

$k =$ index number of refurbishing plant, $k \in \{1, 2, 3, \dots K\}$

$X_{ij} =$ quantity to be shipped from wholesaler/retailer i to collection center j

$Y_{jk} =$ quantity to be shipped from collection center j to refurbishing plant k

$C_{ij} =$ total variable costs per unit for satisfying collection center j by opening retailer/wholesaler i

$W_{kj} =$ total variable costs for satisfying refurbishing plant k by opening collection center j

$D_{ij} =$ transportation distance from wholesaler/retailer i to collection center j

$E_{jk} =$ transportation distnace from collection center j to refurbishing plant k

$SP_i =$ quantity that is recalled to wholesaler/retailer i

$RB_j =$ quantity that is to be refurbished at facility k

Problem formulation and Genetic representation

This multi-objective optimization problem (MOP) is solved to obtain solutions by multi-objective genetic algorithm employed by matlab on a Pentium Core 2 Duo 2.4 GHz with 3 GB of ram. A multi-objective fitness function can be formulated in a vector function form as:

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}$$

The decision variables can be represented in chromosome genotype as:

$$\begin{bmatrix} X_{11} \\ X_{12} \\ X_{13} \\ X_{21} \\ X_{22} \\ X_{23} \\ X_{31} \\ X_{32} \\ X_{33} \\ Y_{11} \\ Y_{12} \\ Y_{21} \\ Y_{22} \\ Y_{31} \\ Y_{32} \end{bmatrix} \quad \leftrightarrow \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{bmatrix}$$

Parameters of the multi-objective genetic algorithm are set as follow:

- Population type: double vector
- Population size: 225 (15*number of genes)
- Creation function: feasible population creation function
- Selection: tournament selection with tournament size = 2
- Crossover fraction = 0.8, mutation fraction = 0.2
- Mutation: adaptive feasible
- Crossover: intermediate with crossover ratio of 1.0
- Migration direction: forward with fraction of 0.2 and interval of 20
- Distance measure function: distance crowding
- Pareto front population fraction = 0.35
- Termination criteria: 3000 generations, 100 stall generations or function tolerance of 10^{-4}

After running the algorithm, it was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 136 generations, 30826 function counts, average Pareto distance = 0.000414734 and average Pareto spread = 0.140132. The top 20 final populations are shown in the table below.

Table 4.7 Top 20 chromosomes ranked by objective function Z_1 values from lowest to highest

Chromosomes	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	Z_1	Z_2
	21144	13049	15806	62364	0	37636	50001	49999	50000	66666	33333	66666	33333	66668	33333	2837160	3825999
	21071	13109	15821	61885	550	37565	50001	49999	50000	66666	33333	66666	33333	66668	33333	2841771	3824745
	21023	13147	15830	61575	905	37520	50001	49999	50000	66666	33333	66666	33333	66668	33333	2844750	3823935
	20996	13169	15835	61400	1106	37494	50001	49999	50000	66666	33333	66666	33333	66668	33333	2846434	3823477
	20918	13232	15850	60898	1683	37419	50001	49999	50000	66666	33333	66666	33333	66668	33333	2851269	3822163
	20868	13273	15859	60571	2058	37371	50001	49999	50000	66666	33333	66666	33333	66668	33333	2854417	3821307
	20846	13290	15864	60430	2220	37350	50001	49999	50000	66666	33333	66666	33333	66668	33333	2855778	3820937
	20807	13322	15871	60174	2514	37312	50001	49999	50000	66666	33333	66666	33333	66668	33333	2858244	3820266
	20755	13364	15881	59839	2899	37262	50001	49999	50000	66666	33333	66666	33333	66668	33333	2861468	3819389
	20717	13394	15888	59595	3179	37226	50001	49999	50000	66666	33333	66666	33333	66668	33333	2863815	3818751
	20672	13431	15897	59298	3520	37182	50001	49999	50000	66666	33333	66666	33333	66668	33333	2866678	3817973
	20637	13459	15904	59073	3778	37149	50001	49999	50000	66666	33333	66666	33333	66668	33333	2868843	3817384
	20600	13489	15911	58833	4053	37113	50001	49999	50000	66666	33333	66666	33333	66668	33333	2871149	3816757
	20552	13527	15920	58526	4407	37068	50001	49999	50000	66666	33333	66666	33333	66668	33333	2874113	3815951
	20507	13565	15929	58228	4748	37023	50001	49999	50000	66666	33333	66666	33333	66668	33333	2876977	3815172
	20468	13596	15936	57978	5036	36986	50001	49999	50000	66666	33333	66666	33333	66668	33333	2879387	3814517
	20416	13638	15946	57641	5422	36937	50001	49999	50000	66666	33333	66666	33333	66668	33333	2882625	3813637
	20370	13675	15955	57345	5762	36893	50001	49999	50000	66666	33333	66666	33333	66668	33333	2885478	3812861
	20345	13695	15960	57179	5952	36868	50001	49999	50000	66666	33333	66666	33333	66668	33333	2887073	3812427
	20308	13725	15967	56940	6228	36833	50001	49999	50000	66666	33333	66666	33333	66668	33333	2889380	3811800

Table 4.8 Top 20 chromosomes ranked by objective function Z_2 values from lowest to highest

Chromosomes	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	Z_1	Z_2
	11525	20820	17655	0	71606	28394	50001	49999	50000	66666	33333	66666	33333	66668	33333	3437590	3662741
	11525	20820	17655	0	71606	28394	50001	49999	50000	66666	33333	66666	33333	66668	33333	3437590	3662741
	11574	20780	17645	318	71240	28441	50001	49999	50000	66666	33333	66666	33333	66668	33333	3434526	3663574
	11593	20765	17642	441	71099	28460	50001	49999	50000	66666	33333	66666	33333	66668	33333	3433341	3663897
	11642	20726	17633	756	70738	28506	50001	49999	50000	66666	33333	66666	33333	66668	33333	3430309	3664721
	11672	20702	17627	951	70514	28535	50001	49999	50000	66666	33333	66666	33333	66668	33333	3428436	3665230
	11689	20687	17623	1066	70382	28552	50001	49999	50000	66666	33333	66666	33333	66668	33333	3427326	3665532
	11718	20664	17618	1253	70168	28580	50001	49999	50000	66666	33333	66666	33333	66668	33333	3425530	3666021
	11775	20618	17607	1620	69746	28634	50001	49999	50000	66666	33333	66666	33333	66668	33333	3421992	3666983
	11798	20600	17603	1768	69576	28656	50001	49999	50000	66666	33333	66666	33333	66668	33333	3420572	3667368
	11831	20573	17596	1985	69326	28688	50001	49999	50000	66666	33333	66666	33333	66668	33333	3418476	3667938
	11889	20526	17585	2361	68895	28744	50001	49999	50000	66666	33333	66666	33333	66668	33333	3414860	3668922
	11918	20502	17579	2550	68679	28772	50001	49999	50000	66666	33333	66666	33333	66668	33333	3413044	3669416
	11977	20455	17568	2928	68244	28828	50001	49999	50000	66666	33333	66666	33333	66668	33333	3409402	3670406
	12020	20420	17560	3208	67922	28870	50001	49999	50000	66666	33333	66666	33333	66668	33333	3406703	3671140
	12047	20399	17555	3382	67723	28895	50001	49999	50000	66666	33333	66666	33333	66668	33333	3405031	3671594
	12092	20362	17546	3673	67389	28938	50001	49999	50000	66666	33333	66666	33333	66668	33333	3402227	3672357
	12175	20295	17530	4212	66770	29018	50001	49999	50000	66666	33333	66666	33333	66668	33333	3397039	3673767
	12197	20277	17526	4357	66603	29040	50001	49999	50000	66666	33333	66666	33333	66668	33333	3395639	3674148
	12244	20239	17517	4660	66255	29085	50001	49999	50000	66666	33333	66666	33333	66668	33333	3392721	3674941

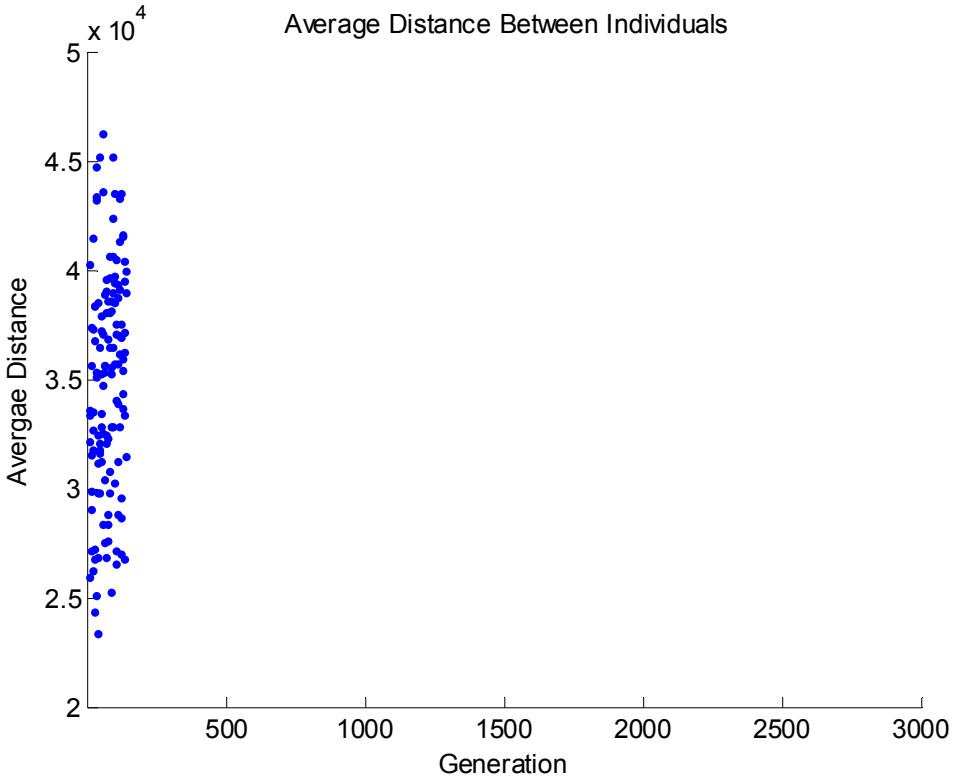

Figure 4.2 Average distance between individuals at each generation

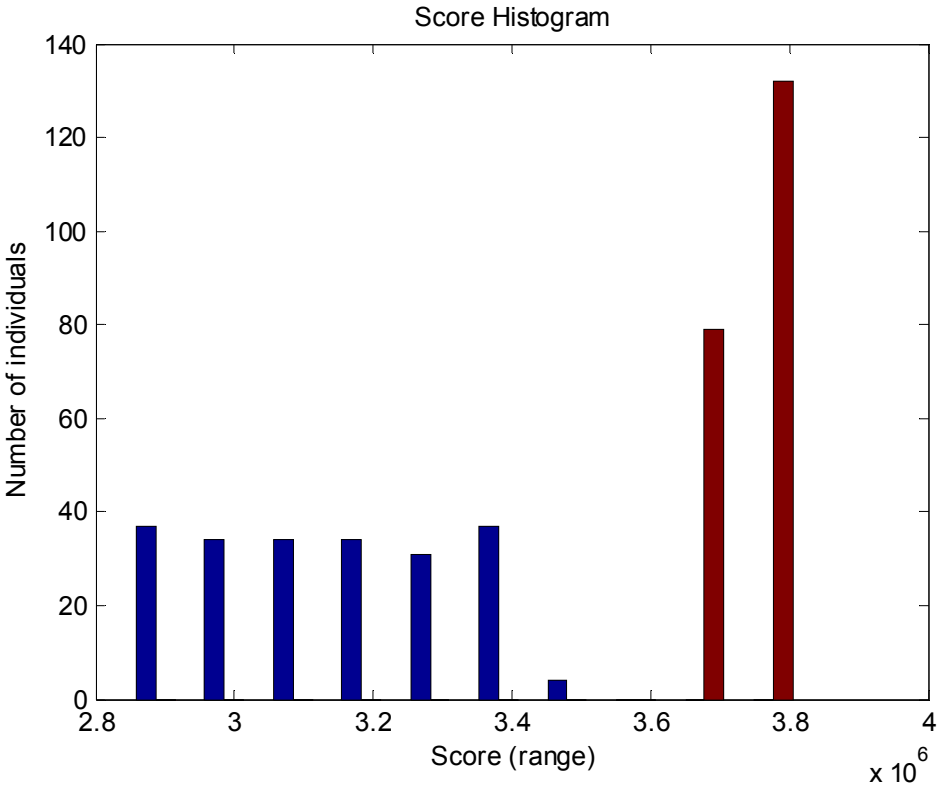

Figure 4.3 A histogram of the scores at each generation

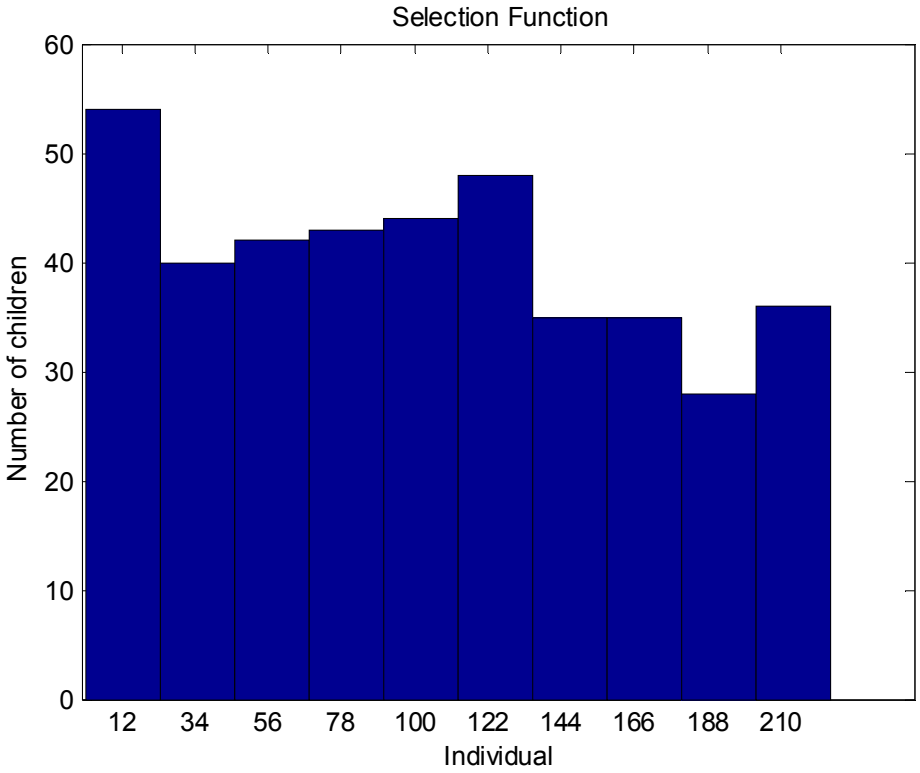

Figure 4.4 A histogram of the parents showing which parents are contributing to each generation

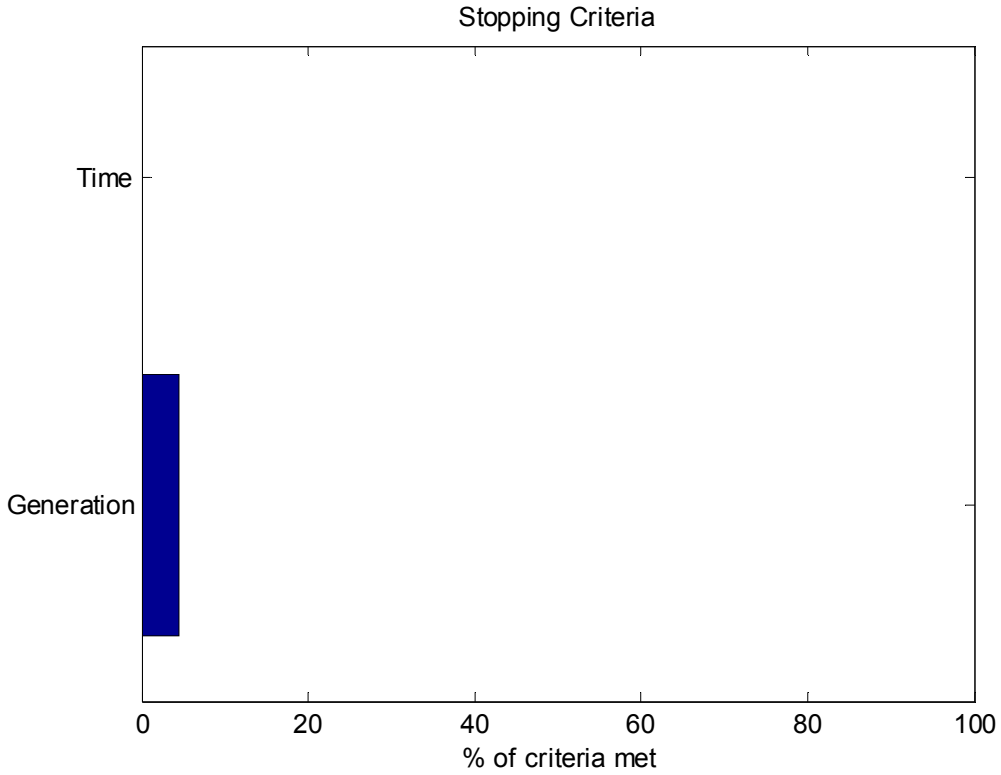

Figure 4.5 A plot shows stopping criteria levels (136 generations).

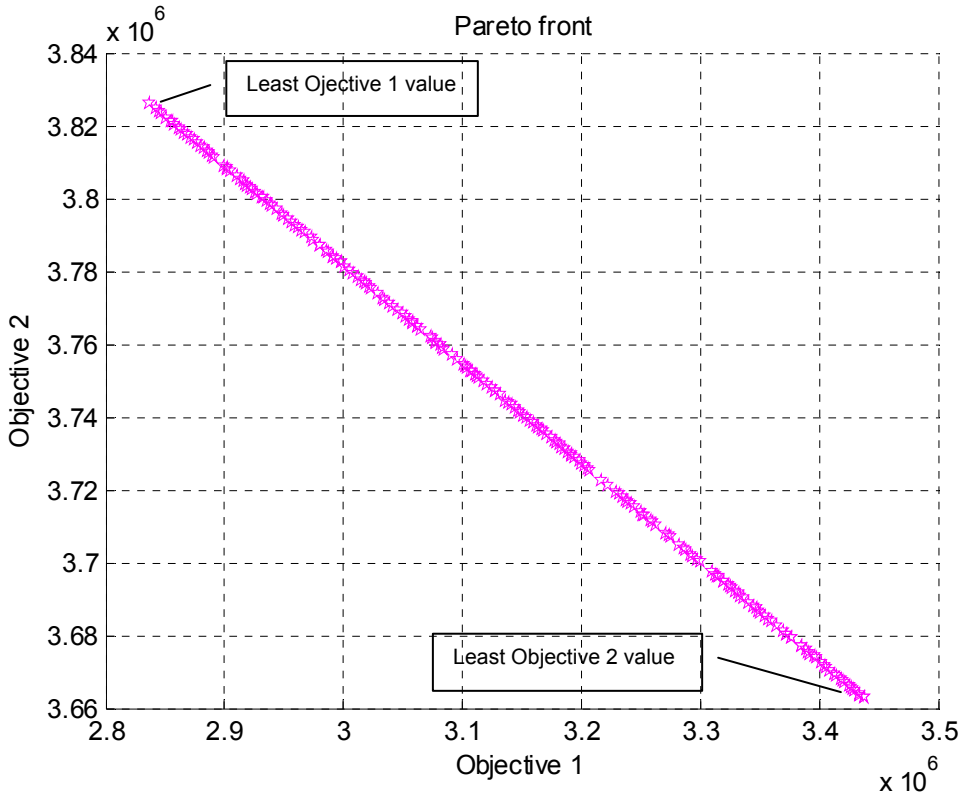

Figure 4.6 Pareto front plot showing objective function values for all noninferior solutions.

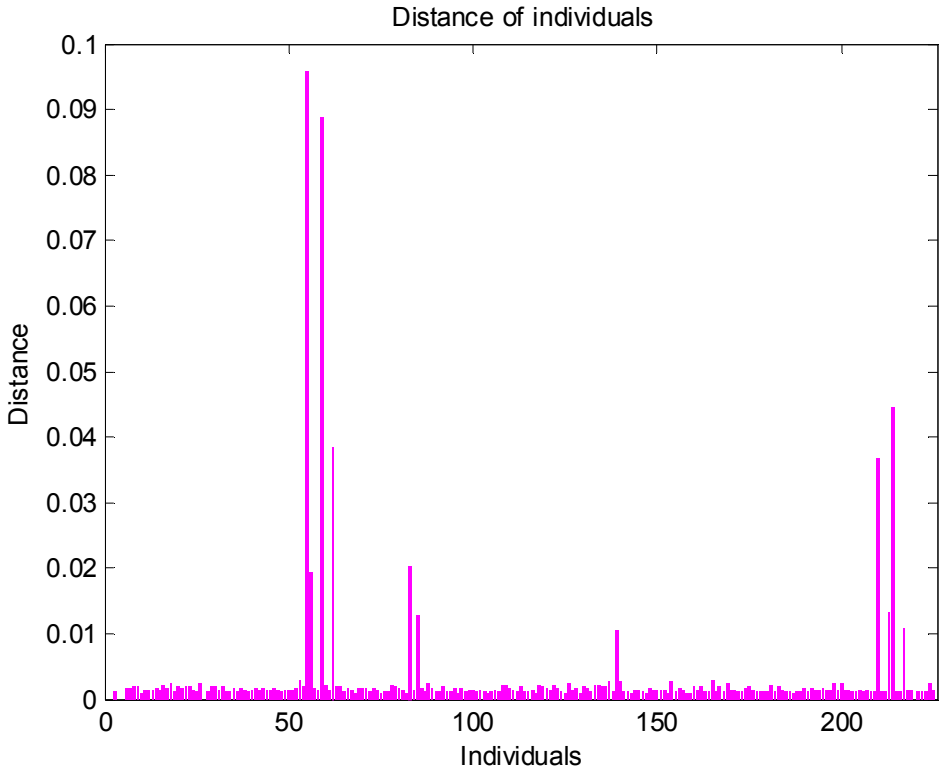

Figure 4.7 Average Pareto distance plot showing the average distance measure between individuals

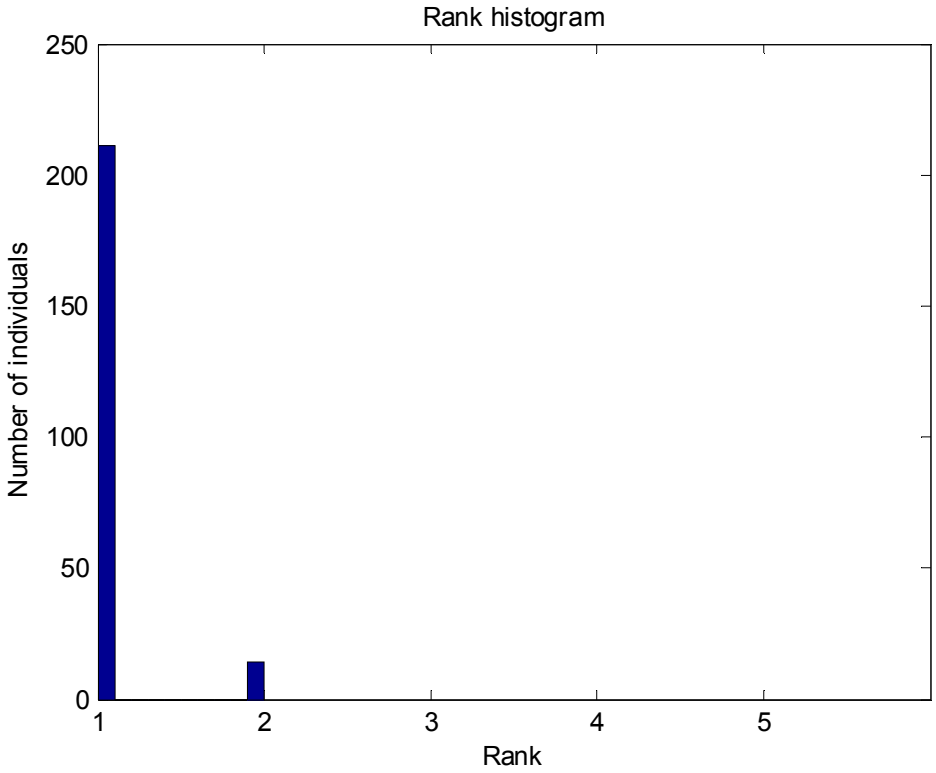

Figure 4.8 Rank histogram plot showing the fraction of individuals in each Pareto tier. Rank 1 individuals are best, rank 2 individuals are dominated only by rank 1 individuals, etc.

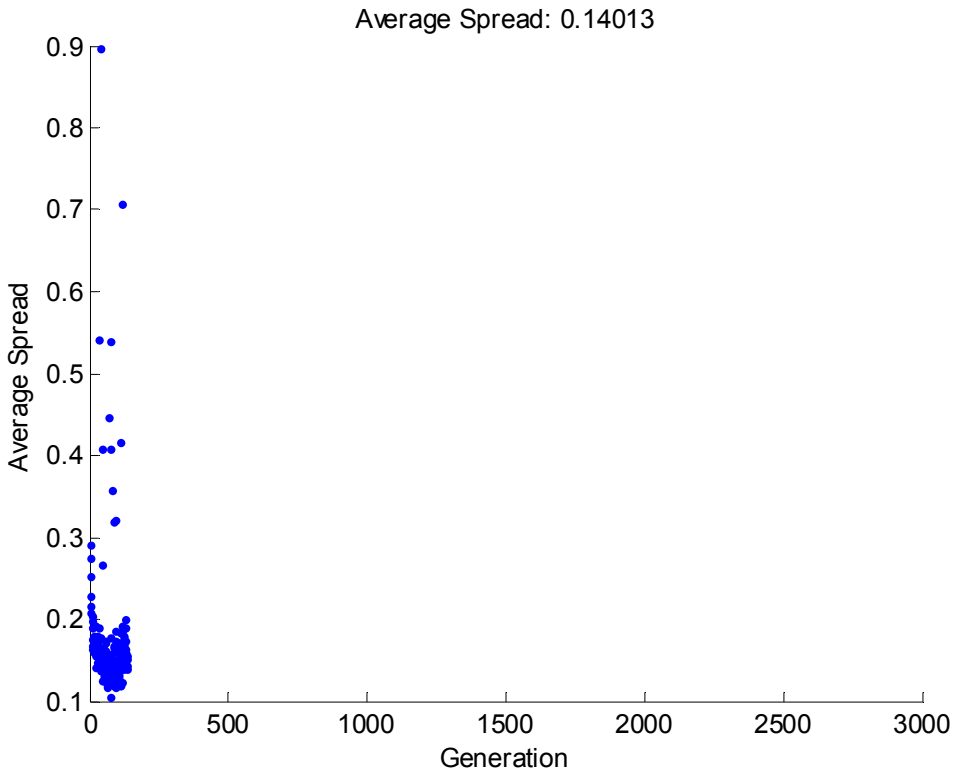

Figure 4.9 Average Pareto spread plot showing the change in distance measure of individuals with respect to the previous generation.

4.2 Data Analysis

Sensitivity analysis is performed for data analysis purpose. Sensitivity analysis investigates the change in the solutions resulting from making changes in parameters of the GA model. In this research, sensitivity analysis shows how sensitive of solutions and decision variables to changes in weights in objective functions. It shows that the solutions of an aggregation method are affected by weight adjustment. Thus, in case of aggregation method, if the weights are not appropriately assigned, the GA may not give out good solutions. On the other hand, for the proposed Pareto method, it is not sensitive to weigh, so incorrect weights do not affect the solution outcome of Pareto based MOGA.

4.2.1 Sensitivity analysis for multi-objective genetic algorithm model using aggregation method

A sensitivity analysis for weighted aggregation method is done by varying weight w_1 and w_2 in the equation below to determine changes in decision variables, $x_1 \, to \, x_{15}$ in GA solutions.

$$Z = w_1 * Z_1 + w_2 * Z_2$$

Case 1: $Z = 0.999 * Z_1 + 0.001 * Z_2$

Case 2: $Z = 0.99 * Z_1 + 0.01 * Z_2$

Case 3: $Z = 0.9 * Z_1 + 0.1 * Z_2$

Case 4: $Z = 0.5 * Z_1 + 0.5 * Z_2$

Case 5: $Z = 0.1 * Z_1 + 0.9 * Z_2$

Case 6: $Z = 0.01 * Z_1 + 0.99 * Z_2$

Case 7: $Z = 0.001 * Z_1 + 0.999 * Z_2$

The decision variables and objective function values are obtained as shown in table… Some of decision variables are highly sensitive to weigh variation. The pivoting point of these variables is where $w_1 \text{ value becomes less than } w_2$. A plot of all cases is shown below.

Table 4.9 Decision variables and objective function values for each case

Case	*x1*	*x2*	*x3*	*x4*	*x5*	*x6*	*x7*	*x8*	*x9*	*x10*	*x11*	*x12*	*x13*	*x14*	*x15*	*Z*
Case 1	21102	13084	15814	62364	0	37636	50000	50000	50000	66667	33333	66667	33333	66667	33333	2837993
Case 2	19688	14226	16086	62364	0	37636	50000	50000	50000	66667	33333	66667	33333	66667	33333	2841936
Case 3	21143	13050	15807	62364	0	37636	50001	49999	50000	66666	33333	66666	33333	66668	33333	2936042
Case 4	21145	13049	15806	62364	0	37636	50000	50000	50000	66667	33333	66667	33333	66667	33333	3331579
Case 5	14973	18034	16992	0	71606	28394	50001	49999	50000	66666	33333	66666	33333	66668	33333	3619627
Case 6	11525	20820	17655	0	71607	28393	50000	50000	50000	66667	33333	66667	33333	66667	33333	3660488
Case 7	11525	20820	17655	0	71607	28393	50001	50000	50000	66666	33333	66666	33333	66668	33333	3662519

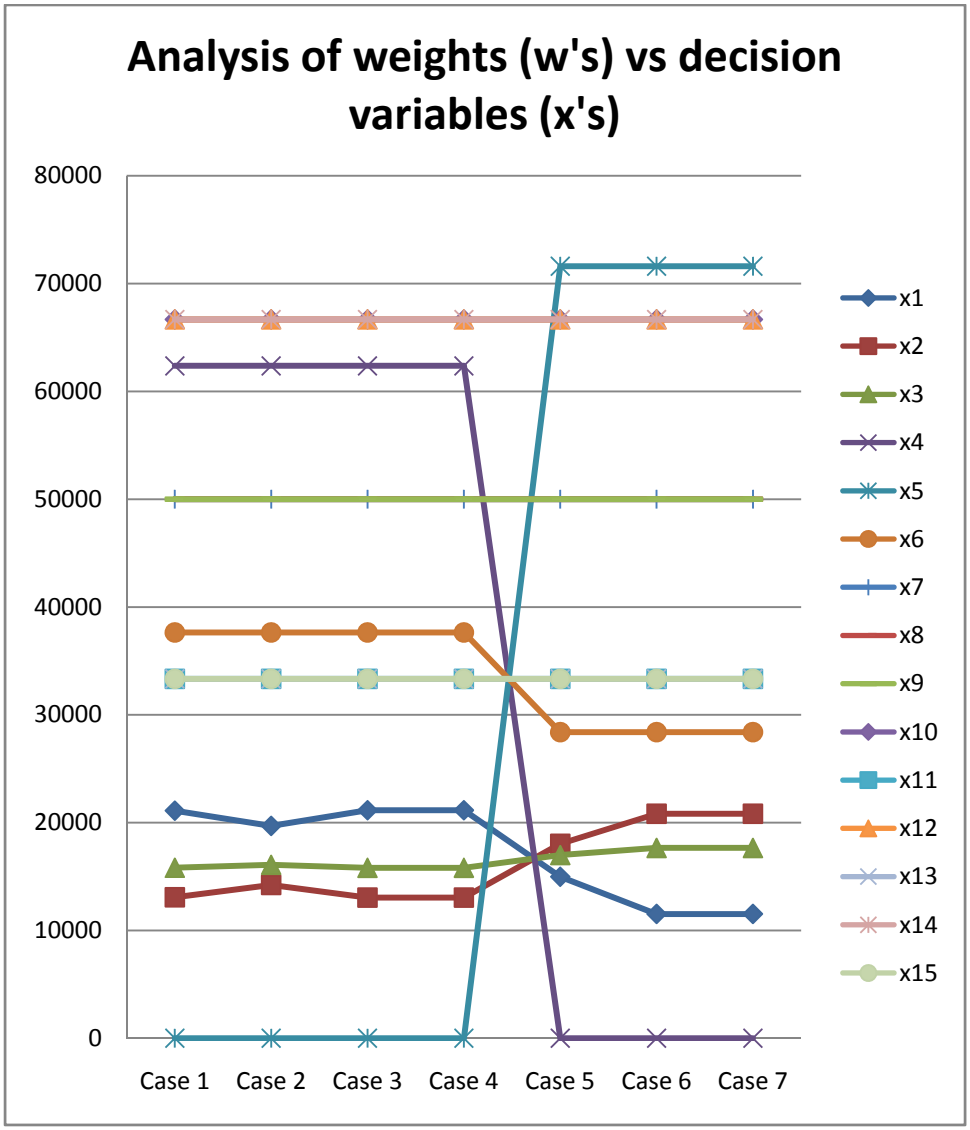

Figure 4.10 Analysis of weight w_1 and w_2 versus decision variables $x_1 - x_{15}$ in aggregation method

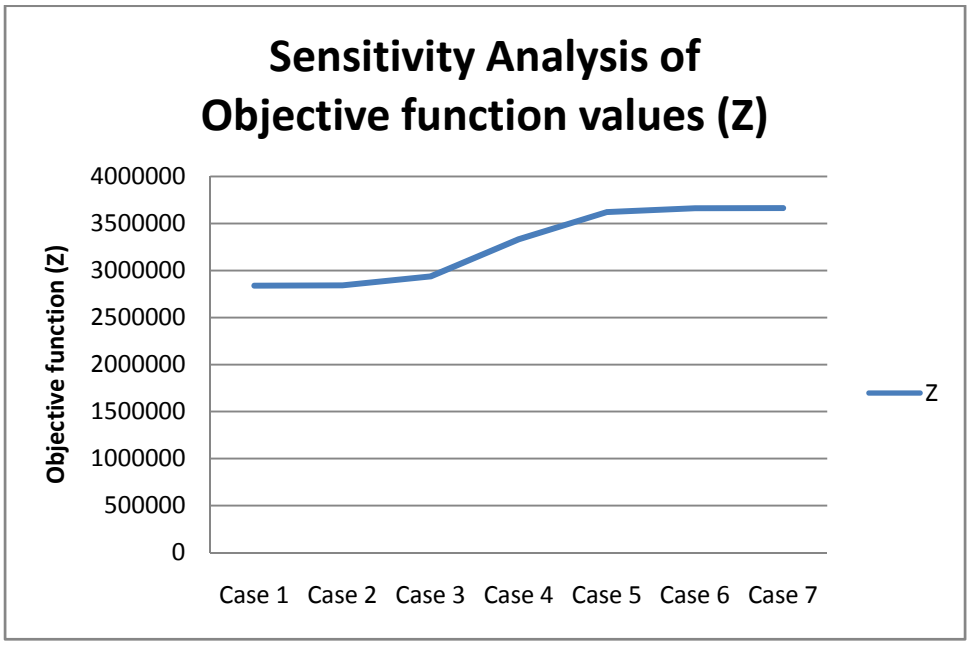

Figure 4.11 A sensitivity analysis of objective function value (Z) in aggregation method

4.2.2 Sensitivity Analysis for Pareto based MOGA

In the case of Pareto based MOGA, the definition of optimization is changed from finding optimal solutions to finding Pareto optimal solutions-compromise solution belong to the set of non-dominated solutions. Z is a vector of objective functions Z1 and Z2. Weights are assigned to z1 and z2 to generate 7 cases. Pareto front plots are compared between cases to see any changes in solutions. There is very little change in the pattern of Pareto front. The only difference is scale of Z_1 and Z_2 which are caused by weight variation. It is apparent that the Pareto based MOGA does not affected by weight changes as the aggregation method. Therefore, the Pareto based MOGA is more robust and does not need prior knowledge for defining a weight of each objective in multi-objective optimization problems.

Case 1: $Z = \begin{bmatrix} 0.999 * Z_1 \\ 0.001 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 102 generations with 23176 function counts, average Pareto distance of 0.000455465 and average Pareto spread of 0.160169

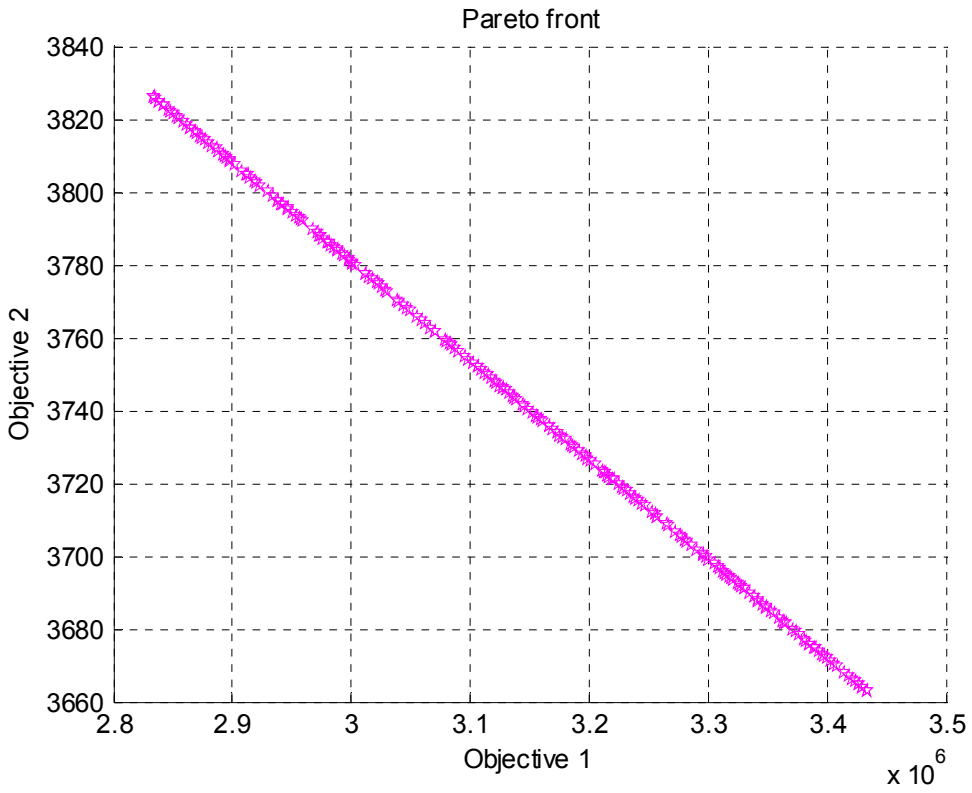

Figure 4.12 Pareto front plot showing results from case 1

Case 2: $Z = \begin{bmatrix} 0.99 * Z_1 \\ 0.01 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 102 generations with 23176 function counts, average Pareto distance of 0.00060696 and average Pareto spread of 0.121614.

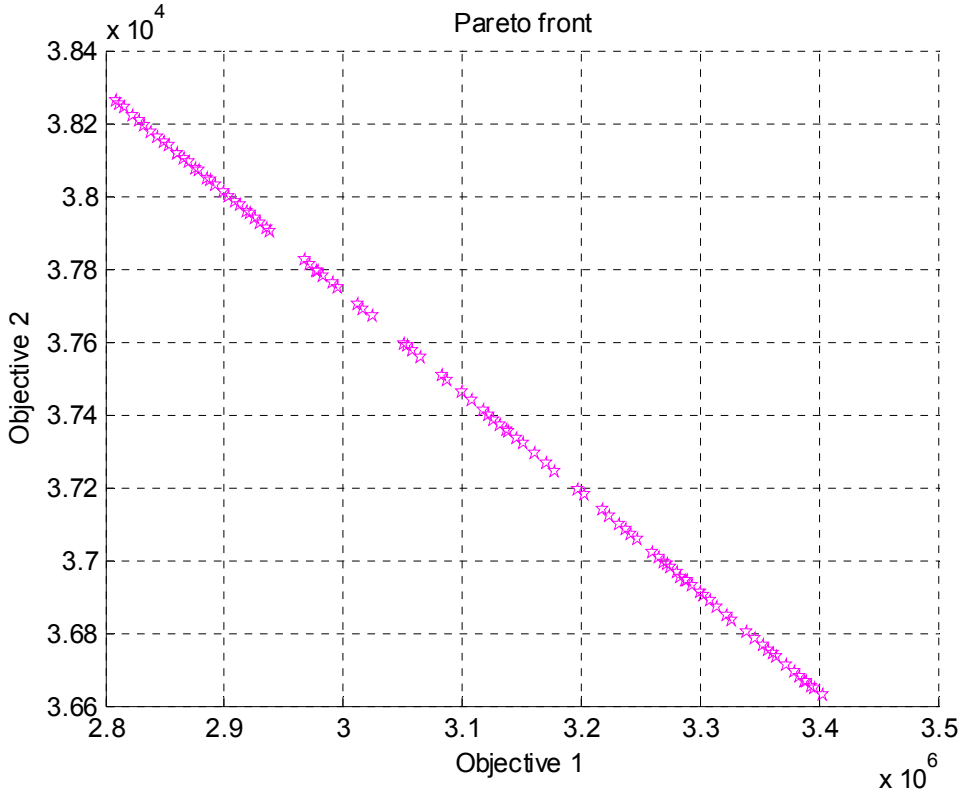

Figure 4.13 Pareto front plot showing results from case 2

Case 3: $Z = \begin{bmatrix} 0.9 * Z_1 \\ 0.1 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 102 generations with 23176 function counts, average Pareto distance of 0.000478192 and average Pareto spread of 0.168348.

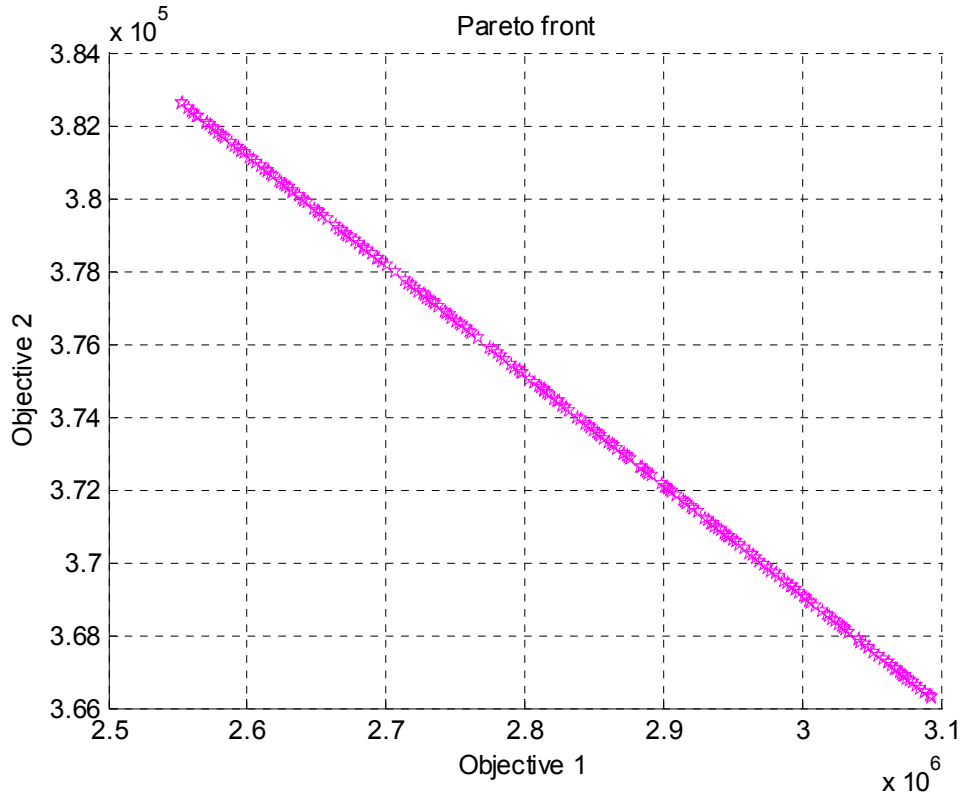

Figure 4.14 Pareto front plot showing results from case 3

Case 4: $Z = \begin{bmatrix} 0.5 * Z_1 \\ 0.5 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 119 generations with 27001 function counts, average Pareto distance of 0.000527609 and average Pareto spread of 0.117075.

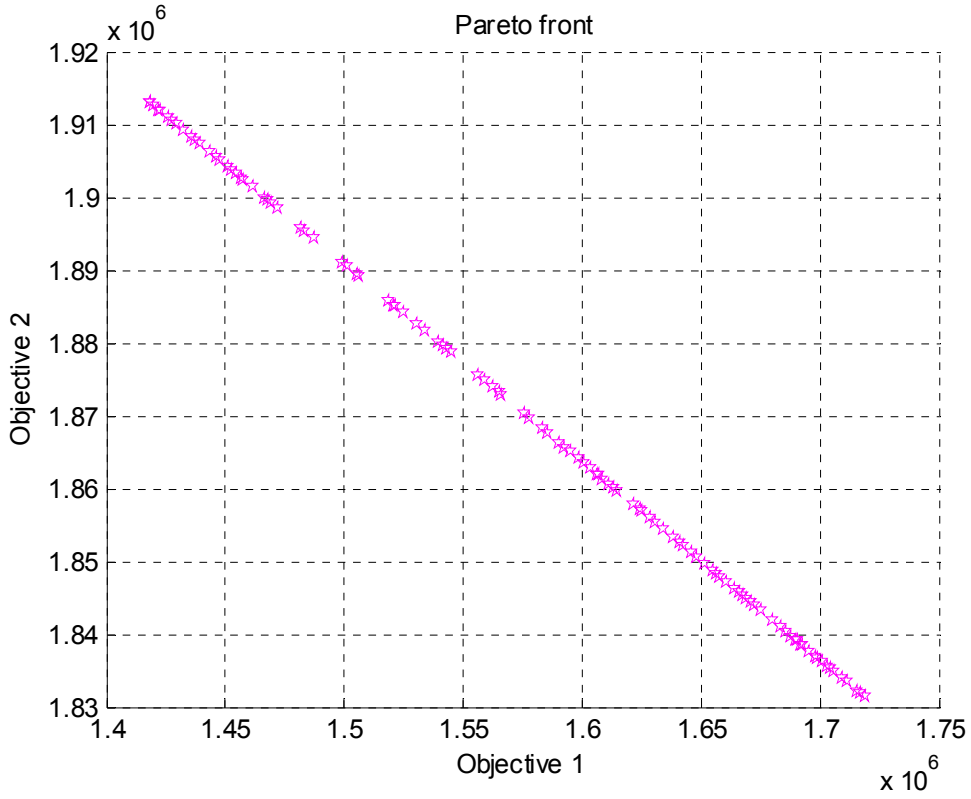

Figure 4.15 Pareto front plot showing results from case 4

Case 5: $Z = \begin{bmatrix} 0.1 * Z_1 \\ 0.9 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 120 generations with 27226 function counts, average Pareto distance of 0.000476068 and average Pareto spread of 0.14933.

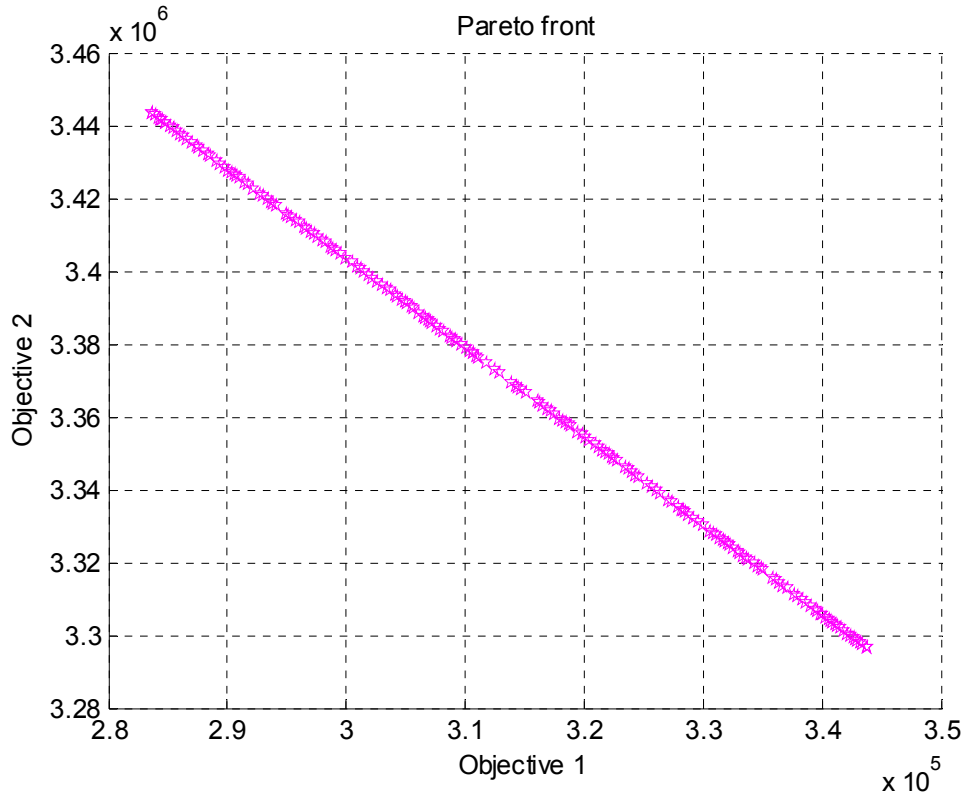

Figure 4.16 Pareto front plot showing results from case 5

Case 6: $Z = \begin{bmatrix} 0.01 * Z_1 \\ 0.99 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 120 generations with 27226 function counts, average Pareto distance of 0.000544765 and average Pareto spread of 0.15128.

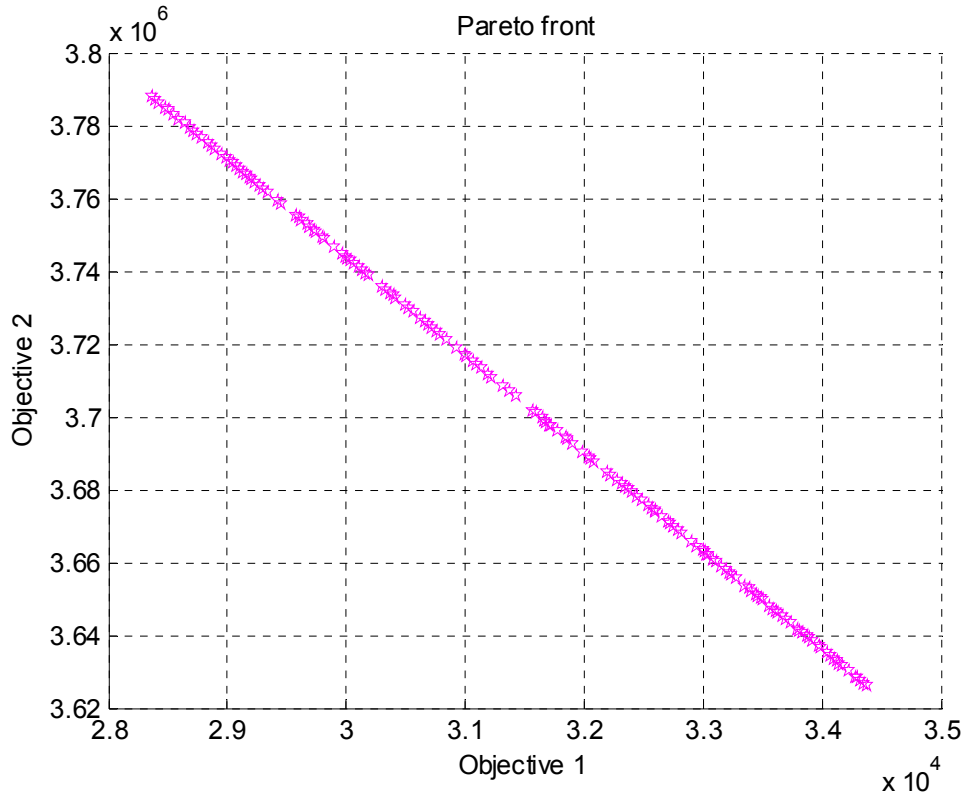

Figure 4.17 Pareto front plot showing results from case 6

Case 7: $Z = \begin{bmatrix} 0.001 * Z_1 \\ 0.999 * Z_2 \end{bmatrix}$

The algorithm was terminated by function tolerance criterion (average change in the spread of Pareto solutions less than function tolerance) at 104 generations with 23626 function counts, average Pareto distance of 0.000396804 and average Pareto spread of 0.140807.

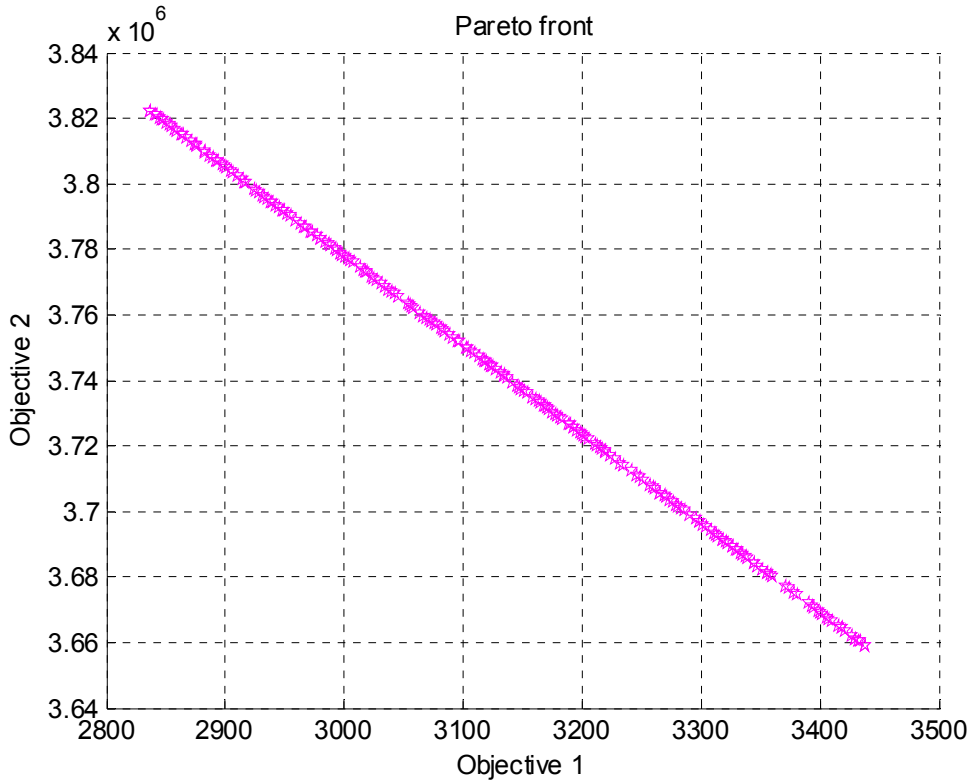

Figure 4.18 Pareto front plot showing results from case 7

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Most of the reverse logistics network designs in previous literature focused on minimizing total costs only. In real world problems, there are multiple objectives to be considered and they are usually in conflict. An aggregation method that is usually used to transform multiple objectives to single objective does not provide good solutions if the weights are not properly assigned. Prior domain knowledge is also required in order to obtain appropriate weights. Pareto based method, employing dominance ranking schemes, can be used to achieve nondominated solutions which optimally balance the trade-offs among objectives. In addition, Genetic Algorithms can obtain good quality solutions in short time and are suitable for the multi-objective environment due to its population based nature.

Literature in reverse logistics, reverse logistics network design, multiple objective optimization, Genetic Algorithms are discussed in this dissertation. The Pareto based multi-objective Genetic Algorithms model are constructed. A case study and sensitivity analysis are also performed.

The proposed model utilizes Pareto based Genetic Algorithms to solve multiple objective problems in reverse logistics network design. The Pareto based method is designed to obtain non-dominated solutions from multiple objective problems coupled with genetic algorithms which can obtain good quality solution efficiently.

In order to validate the proposed model, a case study was conducted with generated data based on product recall network on previous work. A sensitivity analysis was also conducted to compare robustness and stability between aggregation based multi-objective

genetic algorithms and Pareto based multi-objective genetic algorithms. The results show that Pareto based method is not susceptible to inappropriate weight assignment. Therefore, it is more robust for multi-objective environment. Furthermore, Genetic Algorithms can find solutions more efficiently than conventional optimization techniques.

5.2 Future work

There are ideas that came to the author's attention as future research that can further improve this proposed reverse logistics design model:

- Include transshipment in reverse logistic network design model to improve efficiency of merchandise return with high marginal value of time such as laptop, computers, etc.
- Add multimodal capability.
- Combine multiple period (dynamic) capability in the design model.
- Include capability for the user to choose how many and which facility to be opened or closed.
- Analytical Hierarchy Process (AHP) might be used to evaluate non-dominated/ non-inferior/ Pareto optima instead of the ranking procedure if more than two objectives are considered.
- Apply other evolution computation algorithms such as Differential Evolution (DE) to the multi-objective reverse logistics network design. DE is a population based search strategy similar to standard genetic algorithms. DE main difference from GA occurs in reproduction step where are created from three parents utilizing an arithmetic cross-over operator.
- Implement a multi-objective Genetic Algorithm in programming language such as C++ or Java to be able to run the algorithm more efficiently.

REFERENCES

Ahn, C. W. (2006). Advances in evolutionary algorithms, theory design and practice, Springer.

Anstreicher, K. M., N. W. Brixius, et al. (2002). "Solving large quadratic assignment problems on computational grids." Mathematical Programming **91**(3): 563-588.

Anthony, R. N. (1965). Planning and control systems; a framework for analysis. Boston, Division of Research, Graduate School of Business Administration, Harvard University.

Applegate, D., R. Bixby, et al. (1995). "Finding Cuts in the TSP." Technical report **95-05**.

Azzone, G., G. Noci, et al. (1996). "DEFINING ENVIRONMENTAL PERFORMANCE INDICATORS: AN INTEGRATED FRAMEWORK." Business Strategy and the Environment **5**(2): 69-80.

Back, T., D. B. Forgel, et al. (2000). Evolutionary Computation 1, Basic Algorithms and operators. Bristol and Philadelphia, Institute of Physics Publishing.

Beamon, B. M. (1998). "Supply chain design and analysis: Models and Methods." International Journal of Production Economics **55**(3): 281-294.

Bostel, N., P. Dejax, et al. (2005). The Design, Planning, and Optimization of Reverse Logistics Networks. Logistics Systems: Design and Optimization**:** 171-212.

Chapman, S. J. (2008). MATLAB programming for engineers. Toronto, Ontario Thomson.

Coello Coello, C. A., G. B. Lamont, et al. (2007). Evolutionary algorithms for solving multi-objective problems. New York, Springer.

Cohon, J. L. and D. H. Marks (1975). "A Review and Evaluation of Multiobjective Programming Techniques." Water Resources Research **11(2)**: 208-220.

Davis, D. and K. Kostuik (2007). Genetic evolution as a foundation for supply chainoptimization. World Trade Magazine.

De Brito, M. P. (2004). Managing Reverse Logistics or Reversing Logistics Management?

De Brito, M. P. and R. Dekker (2002). "Reverse Logistics - A Framework." Econometric Institute Report **EI 2002-38**.

De Jong, K. A. (2006). Evolutionary computation : a unified approach, Bradford book.

Dekker, R. (2004). Reverse logistics : quantitative models for closed-loop supply chains. Berlin; New York, Springer.

Demirel, N. O. and H. Gokcen (2007). "A mixed integer programming model for remanufacturing in reverse logistics environment." Int. J. Advanced Manufacturing Technology.

Dorigo, M. and C. Blum (2005). "Ant colony optimization theory: A survey." Theoretical Computer Science **344**: 243-278.

Dorigo, M. and L. M. Gambardella (1997). "Ant Colony System: A cooperative learning approach to the traveling salesman problem." IEEE Trans. on Evolutionary Computation **1**(1): 53-65.

Dorigo, M., V. Maniezzo, et al. (1996). "Ant System: Optimization by a colony of cooperating agents." IEEE Trans. Syst., Man, and Cybern.

Dorigo, M. and K. Socha (2006). An introduction to Ant Colony Optimization. IRIDIA - Technical Report Series, IRIDIA.

Dorigo, M. and T. Stutzle (2004). Ant Colony Optimization. Cambridge, MA, MIT Press.

Eiben, A. E. and J. E. Smith (2003). Introduction to Evolutionary Computing. Berlin, Springer-Verlag.

Engelbrecht, A. P. (2005). Fundamentals of computational swarm intelligence. Hoboken, NJ, Wiley.

Fleischmann, M. (2000). Quantitative Models for Reverse Logistics Erasmus University Rotterdam. **PhD**.

Fleischmann, M. (2001). Reverse logistics network structures and design. ERIM Report Series Research in Management. Rotterdam, Eramus Research Institute of Management (ERIM).

Fleischmann, M., P. Beullens, et al. (2001). "The impact of product recovery on logistics network design." Production and Operations Management **10**(2): 156-172.

Fleischmann, M., J. M. Bloemhof-Ruwaard, et al. (1997). "Quantitative models for reverse logistics: a review." European Journal of Operational Research **103**(1): 1-17.

Fleischmann, M., H. R. Krikke, et al. (2000). "A characterisation of logistics networks for product recovery." Omega **28**(6): 653-666.

Geunes, J. and P. M. Pardalos (2005). Supply chain optimization New York Springer.

Harhalakis, G., R. Nagi, et al. (1993). "Hierarchical Modeling Approach for Production Planning." IFAC SYMPOSIA SERIES.

IDSIA. from http://www.idsia.ch.

Jayaraman, V. (1999). "A multi-objective logistics model for a capacitated service facility problem." International Journal of Physical Distribution & Logistics Management **29**(1): 65-81.

Jayaraman, V., V. D. R. Guide Jr, et al. (1999). "A Closed-Loop Logistics Model for Remanufacturing." The Journal of the Operational Research Society **50**(5): 497-508.

Jayaraman, V., R. A. Patterson, et al. (2003). "The design of reverse distribution networks: Models and solution procedures." European Journal of Operational Research **150**(1): 128-149.

Krikke, H., I. l. Blanc, et al. (2004). "Product modularity and the design of closed-loop supply chains." California Management Review **46**(2).

Krikke, H., J. Bloemhof-Ruwaard, et al. (2003). "Concurrent product and closed-loop supply chain design with an application to refrigerators." International Journal of Production Research **41**(16): 3689-3719.

Krikke, H. R., J. M. Bloemhof-Ruwaard, et al. (2001). Dataset of the Refrigerator Case: Design of Closed Loop Supply Chains. ERIM Report Series. Rotterdam, SSRN.

Krikke, H. R., J. M. Bloemhof-Ruwaard, et al. (2001). Design of Closed Loop Supply Chains. ERIM Report Series
SSRN.

Krikke, H. R., A. V. Harten, et al. (1999). "Business Case Oce: Reverse Logistic Network Re-Design for Copiers " OR Spektrum
21(3).

Krikke, H. R., C. P. Pappis, et al. (2001). Design Principles for Closed Loop Supply Chains. ERIM Report Series, SSRN.

Krikke, H. R., A. Van Harten, et al. Business Case Oce: Reverse Logistic Network Re-Design for Copiers, SSRN.

Kumanan, S., S. P. Venkatesan, et al. (2007). "Optimization of supply chain logistics network using random search techniques." International Journal of Logistics Systems and Management **3**(2): 252-266.

Kumanan, S., S. P. venkatesan, et al. (2006). "Perofrmance analysis of supply chain network using a genetic algorithm and simulation." Int. J. Agile Systems and Management **1**(4): 450-462.

Kusumastuti, R. D., R. Piplani, et al. (2008). "Redesigning closed-loop service network at a computer manufacturer: A case study." International Journal of Production Economics **111**(2): 244-260.

Kusumastuti, R. D., R. Piplani, et al. (2004). An approach to design reverse logistics networks for product recovery, Singapore, Institute of Electrical and Electronics Engineers Inc., Piscataway, NJ 08855-1331, United States.

Langevin, A. and D. Riopel (2005). Logistics systems : design and optimization. New York, Springer.

Lebreton, B. and A. Tuma (2006). "A quantitative approach to assessing the profitability of car and truck tire remanufacturing." International Journal of Production Economics **104**(2): 639-652.

Lee, J.-E., M. Gen, et al. (2008). "Network model and optimization of reverse logistics by hybrid genetic algorithm." Computer & Industrial Engineering **xxx**(xxx): xxx.

Lieckens, K. and N. Vandaele (2000). "Reverse logistics network design with stochastic lead times." Computer & Operations Research **xxx**(xxx): xxx.

Lim, G. H., R. D. Kusumastuti, et al. (2005). Designing a reverse supply chain network for product refurbishment. International Conference on Simluation and Modeling.

Listes, O. (2007). "A generic stochastic model for supply-and return network design." Computers & Operations Research **34**(2007): 417-442.

Listes, O. and R. Dekker (2005). "A stochastic approach to a case study for product recovery network design." European Journal of Operational Research **160**(2005): 268-287.

Lu, Z. and N. Bostel (2005). "A facility location model for logistics systems including reverse flows: The case of remanufacturing activities." Computers & Operations Research **34**(2007): 299-323.

Lu, Z., N. Bostel, et al. (2005). Simple Plant Location Problem with Reverse Flows. Supply Chain Optimisation**:** 151-166.

Matlab (2008). Genetic Algorithm and Direct Search Toolbox 2, Matlab Press.

Miller, P. (2007). "Swarm Behavior." National Geographic Magazine, online edition **07, 2007**.

Min, H., H. J. Ko, et al. (2006). "A genetic algorithm approach to developing the multi-echelon reverse logistics network for product returns." Omega **34**(1): 56-69.

Moncayo-Martinez and D. Zhang (2005). Multi-objective Max-Min Ant System for Desgining the Supply Chain. Exeter, University of Exeter.

Moncayo-Martinez and D. Zhang (2006). Supply chain design using multi-objective Ant Colony Optimization Metaheuristics. Exeter, University of Exeter.

Osyczka, A. (1985). "Multicriteria optimization for engineering design." Design Optimization: 193-227.

Parsopoulos, K. E. and M. N. Vrahatis (2002). "Partical Swarming Optimization Method in Multiobjective Problems." Proceedings of the ACM Symposium on Applied Computing.

Parsopoulos, K. E. and M. N. Vrahatis (2002). "Recent Approaches to Global Optimization Problems through Partical Swarming Optimization." Natural Computing **1(2-3)**

Poirier, C. C. (2003). Using Models to Improve the Supply Chain, Routledge**:** 263 - 266.

Pokharel, S. (2007). "A two objective model for decision making in a supply chain." International Journal of Production Economics **111**(2008): 378-388.

Realff, M. J., J. C. Ammons, et al. (2000). "Strategic design of reverse production systems." Computers & Chemical Engineering **24**: 991-996.

Realff, M. J., J. C. Ammons, et al. (2004). "Robust reverse production system design for carpet recycling." IIE Transactions **36**: 767-776.

ReturnBuy.com. (2000). "http://www.returnbuy.com/."

Revlog. "The European Working Group on Reverse Logistics." from http://www.fbk.eur.nl/OZ/REVLOG/welcome.html.

Rogers, D. S., R. S. Tibben-Lembke, et al. (1999). Going backwards : reverse logistics trends and practices. [Reno], University of Nevada, Reno, Center for Logistics Management.

S.N.Sivanandam and S.N.Deepa (2008). Introduction to Genetic Algorithms. Berlin, Springer-Verlag.

Schleiffer, R., J. Wollenweber, et al. (2004). Application of Genetic Algorithm for the design of large-scale reverse logistic networks in Europe's automotive industry. The 37th Hawaii International Conference on System Sciences.

Shear, H., T. Speh, et al. (2003). The warehousing link of reverse logistics. annual warehousing education and research council conference.

Shih, L.-H. (2001). "Reverse logistics system planning for recycling electrical appliances and computers in Taiwan." Resources, Conservation and Recycling **32**(1): 55-72.

Silva, C. A., J. M. C. Sousa, et al. (2005). "Soft computing optimization methods applied to logistic processes." International Journal of Approximate Reasoning **40 (2005)**: 280–301.

Sivanandam, S. N. and S. N. Deepa (2008). Introduction to genetic algorithms. Berlin ; New York, Springer.

Song, H., S. Liman, et al. (2006). A quantitative reverse logistics model and waste application for electronic products, Scottsdale, AZ, United States, Institute of Electrical and Electronics Engineers Inc., Piscataway, United States.

Stutzle , T. and H. H. Hoos (2000). "MAX-MIN Ant System." Future Generation Computing System.

Stutzle, T. and H. Hoos. (2005). Stochastic Local Search: Foundations and Applications, Morgan Kaufmann.

Sumathi, S., T. Hamsapriya, et al. (2008). Evolutionary intelligence : an introduction to theory and applications with Matlab. Berlin, Springer.

Takata, S. and Y. Umeda (2007). Advances in life cycle engineering for sustainable manufacturing businesses : proceedings of the 14th CIRP Conference on Life Cycle Engineering, Waseda University, Tokyo, Japan, June 11th-13th, 2007. London, Springer.

Tang, Q. and F. Xie (2007). A Genetic Algorithm for reverse logistics network design. International Conference on Natural Computation, IEEE.

Theerawatsathein, K. and T. Achalakul (2005). "THE PARALLEL DYNAMIC ROUTING SIMULATION BASED ON ANTNET." Proceedings of the 2005 International Conference on Simulation and Modeling.

Thierry, M. C. (1993). Strategic production and operations management issues in product recovery management. Rotterdam, Erasmus Universiteit/Rotterdam School of Management, Faculteit Bedrijfskunde.

Tibben-Lembke, R. S. and D. S. Rogers (2002). "Differences between forward and reverse logistics." Supply Chain Management: An International Journal **7**(5): 271-282.

Umeda, Y., A. Nonomura, et al. (2000). "Study on life-cycle design for the post mass production paradigm." AI EDAM **14**(02): 149-161.

Union, O. J. o. t. E. (2007). WEEE Directive 2002/96/EC. WEEE Directive

Wang, H. S. (2009). "A two-phase ant colony algorithm for multi-echelon defective supply chain network design." European Journal of Operational Research **192**(2009): 243-252.

"WEEE Returns." from http://www.sun.com/aboutsun/ehs/weee.html.

Yu, T., L. Davis, et al. (2008). Evolutionary Computation in Practice, Springer-Verlag.

BIOGRAPHICAL INFORMATION

Sanya Yimsiri has received his Bachelor of Engineering in Industrial Instrumentation Engineering from King Mongkut's Institute of Technology Ladkrabang in 1993, Master of Science in Engineering Management from the University of Missouri – Rolla in 1998 and Doctor of Philosophy in Industrial and Manufacturing Systems Engineering from the University of Texas at Arlington in 2009. Sanya has worked in both engineering and management fields at multi-national companies including Asea Brown Boveri, Otis Elevator and Delta Electronics. He has worked in academic field at Siam University after he finished his Master's. During his Ph.D. study he has worked as a Graduate Research Assistant at Small Business Development Center for Enterprise Excellence to assist many small businesses improve their processes in North Texas region.

Sanya involved in various projects including ISO 9000 quality management system, Lean Six Sigma and inventory management. His main research interests are in supply chain management and optimization, reverse logistics network design, multi-criteria optimization, facility layout, soft computing, artificial intelligence, productivity improvement and waste reduction.