A MIXED INTEGER LOCATION, INVENTORY AND PRICING MODEL FOR CLOSED LOOP SUPPLY CHAINS

by

Buşra Ürek

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

Koç University

August, 2012

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Buşra Ürek

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Asst. Prof. Onur KAYA

_____

Prof. Fikri KARAESMEN

_____

Asst. Prof. Şükrü Ekin KOCABAŞ

Date: _____

...to my family in love and gratitude...

ABSTRACT

Designing an appropriate closed loop supply chain is crucial not only due to the economic value that can be gained from used products but also due to the environmental concerns raising in popularity in last couple decades. In this paper, we address an inventory-location and pricing problem for a closed-loop supply chain with the purpose of collection of the used products and distribution of the newly produced ones simultaneously with the decision of price and incentive. We present a mixed integer nonlinear facility location allocation model with inventory considerations to decide both optimal location of these collection and distribution centers (CDC) and optimal values of price and incentive offered for new and used products, respectively, in order to maximize the profit. To solve this NP-hard problem, three different hybridized heuristic approaches are proposed throughout the paper. The heuristics correspond to applications of Simulated Annealing, Tabu Search and Genetic Algorithms which all are hybridized with Variable Neighborhood Search. Different neighborhood structures are embedded in these heuristics to obtain better results. The performances of the proposed algorithms are analyzed on extensive set of test instances up to 200 customers and 200 possible locations, and we compared it with the upper bound found by a linearization technique. Finally, we compare the three proposed metaheuristics with each other. In all comparisons, we report optimality gaps calculated with respect to the upper bound. The results of our computational study show that the Tabu Search algorithm hybridized with VNS(TSVNS) gives better results compared to solutions given by the other two metaheuristics. The success of the TSVNS algorithm highly depends on its memory structure while providing an effective diversification mechanism.

ÖZETÇE

Etkili bir kapalı devre tedarik zinciri tasarlamak, kullanılmış ürünlerin toplanmasıyla elde edilecek ekonomik değerlerin yanında son yıllarda artarak populerite kazanan çevresel bilinç kavramı içinde oldukça kritiktir. Bu tezde, yeni üretilen ürünlerin dağıtımı ve aynı zamanda kullanılmış ürünlerinde geri toplanması için gereken fiyat ve teşvik kararlarını envanter ve lokasyon kararlarıyla perçinleyen bir problem ele alınmıştır. Kârı en yüksek değere çıkarmak için açılacak toplama ve dağıtım merkezlerinin en uygun lokasyonları, envanter, fiyat ve teşvik değerleri karışık tamsayılı doğrusal programlama modeli ile incelenmiştir. Bu modelde ürünlerin tüm lojistik işlemlerinin müşteriler ve kullanılmış ürün tedarikçileri tarafından yapıldığı varsayıldığı için modelde nakliye ile alakalı bir gider bulunmamaktadır, fakat aradaki mesafenin talep ve geridönüş oranını olumsuz olarak etkilediği gerçeği dikkate alınmıştır. Bu NP-zor problemi çözebilmek için üç farklı melez sezgisel algoritma önerilmiştir. Melezleme yaparken Benzetimli Tavlama, Tabu ve Genetik algoritmaları, Değişken Komşuluk algoritmasının içine yerleştirilmiştir. En iyi sonuçlara ulaşmak için farklı komşuluk oluşturabilecek operatörler sezgisel metodların içine gömülmüştür. Önerilen algoritmaların performansları 200 müşteri ve 200 lokasyon seçenekli problem büyüklüklerine kadar olan kapsamlı kümeler için incelenip doğrusallaştırma tekniği yardımıyla bulunan üst sınırlar ile kıyaslanmıştır. Son olarak, en iyi performansı bulmak amacıyla, önerilen üç farklı melez sezgisel algoritma birbirleriyle kıyaslanmıştır. Tüm kıyaslamalar sonucunda, üst sınırla olan sapmalar rapor edilmiştir. Yaptığımız sayısal deneylerin sonucuna göre, TSVNS algoritması, amaç fonksiyonu açısından test edılen örneklerde kıyasladığımız diğer iki sezgisel yöntemden çok daha iyi sonuçlar vermiştir. Önerilen sezgisel algoritmanın başarısı, büyük ölçüde, etkili çeşitlendirme mekanizmaları önerirken kullandığı hafıza yapısına bağlıdır.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my supervisor Assoc. Prof. Onur Kaya for his continuous support and invaluable advices during my thesis. He has been a great source of inspiration and provided the right balance of suggestions, criticism, and freedom.

Besides my advisor, I would like to thank to the rest of my thesis committee members, for the critical reading, valuable suggestions and insightful comments.

I would like to acknowledge financial support from TÜBİTAK (The Scientific and Technological Research Council of Turkey) during my master thesis.

I am greatful to Müge Güçlü and Umut Arıtürk not only for their support and unforgettable friendship but also being a sister and brother to me during the master study. I would like to express my appreciation to Belin Beyoğlu, Nihal Bahtiyar, Görkem Sarıyer, Bora Kabatepe, Fatma Virdil, Utku Boz and all my friends at Koç University for their valuable friendship ,the stimulating discussions and all the fun we have had in the last two years.

And special thanks to Abdullah Bilgin for his endless support and worthy friendship during my master study.

Last but not the least, I owe my loving thanks to my family, Emine Ürek, Edip Ürek, Zeynep Ürek and my fiance Burak Çöme. Without their encouragement, patient, support and understanding it would have been impossible for me to finish this work. Their morale support and encouragements were the driving forces behind all my success. To them, I dedicate this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

GrSCM	Green Supply Chain Management
RL	Reverse Logistics
CC	Collection Center
DC	Distribution Center
EOQ	Economic Order Quantity
SA	Simulated Annealing Algorithm
TA	Tabu Search Algorithm
GA	Genetic Algorithm
MINLP	Mixed Integer Non-Linear Programming
CDC	Collection and Distribution Center
VNS	Variable Neighborhood Search
UFLP	Uncapacitated Facility Location Problem
NP	Non-polynomial
SAVNS	Simulated Annealing Algorithm hybridized with VNS
TSVNS	Tabu Search Algorithm hybridized with VNS
GAVNS	Genetic Algorithm hybridized with VNS
$\text{SAVNS}_{\text{EXP}}$	SANVS with exponential cooling schedule
$\text{SAVNS}_{\text{HYP}}$	SAVNS with hyperbolic cooling schedule
$\text{SAVNS}_{\text{LIN}}$	SAVNS with linear cooling schedule
$\text{TSVNS}_{\text{2NL}}$	TSNVS with $2Neighborhood_{Large}$
$\text{TSVNS}_{\text{2NS}}$	TSVNS with $2Neighborhood_{Small}$
TSVNS_{3N}	TSVNS with $3Neighborhood$
GAVNS_{20}	GANVS with $PopulationSize = 20$
GAVNS_{40}	GAVNS with $PopulationSize = 40$
GAVNS_{60}	GAVNS with $PopulationSize = 60$

Chapter 1

INTRODUCTION

Nowadays, the growing concern for environment necessitates integrating some environmentally sound solutions into existing supply chain management in practice. Hence, green supply chain management (GrSCM) does not only gain increasing attention among society, government, practitioners of operations but also gets researchers' attention. Adding a green component into supply chain management refers certain environmental consideration over full product life cycle (1). In other words, GrSCM not only involves all logistics activities from supplier to customer but also related to reverse logistics activities as remanufacturing and refurbishing processes. Reverse logistics (RL) can be defined as the flow of goods from their final destination in order to proper disposal or capturing the residual value. An appropriate reverse logistics network results in more profit and draws a socially responsible picture for companies.

Remanufacturing and recycling defines higher form than material recovery by emphasizing on value-added recovery in the forms of materials, energy and labor. Therefore, remanufacturing has undeniable effect on overall profit because it reduces consumption of the raw materials and provides cost reduction on waste treatment. Besides the economical incentives, most countries' recent legislations force the companies to accomplish take-back responsibilities to reduce waste. The governmental concern creates strong legislative incentives on why and how take back activities are organized for different classes of products. Additionally, social and environmental interests that companies engaged in also direct the managers to new techniques to design supply chains that are both economically and ecologically feasible. In other words, RL has been driven by high profitability, increasing number of legislative incentives, and growing awareness for environment.

Logistics network design is one of the most strategic decisions because whether opening or closing a facility is very expensive, time consuming and infeasible to change in a short time. Many tactical and operational decisions are limited by this strategic decision because logistic network configuration is directly related to facility locations. Take-back activities generate reverse flow of goods originated by product holders, also referred to as end users, from the customer zones. Collected used product should be consolidated before shipping to disassembly or remanufacturing facilities. To manage the reverse flow, RL is gaining more interest in supply chain management.

The form of collecting end-of-life product is mainly related to green network design of the company. After realizing the hidden value remaining in the used product, more profitable collection strategy should be developed by the company. In many cases, take back activities are carried out without any government pressure. If the value that will be recovered from the used product can compensate the cost resulted from take-back activities, companies are inclined to manage their RL activities to increase the overall profit. To achieve its profitability goal, some companies offer incentives to product holders. Amount of offered incentive plays the major role in this type of collection strategy so it must be critically analyzed.

Developing whether customer or company driven collection strategy is another important decision that the company must have. In that respect, two main scenarios should be taken into consideration. In the first scenario, collection can be done by establishing collection centers (CC) and used products can be introduced to the CC by end users. On the other hand, collection activities can also be done by company itself visiting each of end user and taking products separately from their final destinations. To decide on the most feasible return policy, the trade-off between transportation and fixed facility cost should be figured out carefully by the company. Moreover, in the second scenario, customers should return the used product. In this scenario, accessibility of the CC is the most important issue when choosing the location of them. Therefore, many companies usually prefer to use their existing structures of retail channels, also referred to as distribution centers (DC), to carry their taking back activities not only to reduce transportation cost between manufacturing and collection site, but also to address their customers to a known place.

Another challenge of companies when managing their logistic network is to handle inventory cost. In today's competitive business world, the main focus of companies is to reduce their overall operational cost while improving the customer service level. Some components of the returned product can be directly used as a raw material in production. Therefore, the RL inventory system should be organized and controlled appropriately to provide integrity between producer's material resource plans and the return flow amount of used products. The interaction between the location of CC and inventory levels in such a RL network design mainly affects timing and amount of the products that are sent to remanufacturing site. When the demand requirements in a given time period is known and lead times does not vary, economic order quantity (EOQ) formulation can be applied for single item, fixed order cost models. Modified EOQ models have been widely used in literature for RL network design with deterministic return assumption.

In reality, customers' response on price is very sensitive. To neutralize the effect of pricing on demand, in many location based studies in literature, demand rate is assumed to be constant. On the other hand, apart from location focused studies, many researchers have reported large number of studies about price sensitivity of demand. The term price sensitivity refers to the variation of customers' purchases of a product as its price changes. Interpreting the behavior of customers across price changing has become very interesting area in the fields of marketing and psychology and also hard to estimate. According to the Law of Demand mentioned by Clark (2) in 1917, customers tend to buy more of a good when its price decreases. As a rule of thumb, exponential nature is assumed for the relationship between demand and price in literature, generally. However, in the literature, there is not much intention to study all these inventory location and pricing decisions together within one model.

In this respect, the contribution of our study is two-folds. First, we develop a new model where location, inventory and pricing decisions affect the design of reverse supply chain network. To the best of our knowledge, this approach has not yet been suggested in the literature. To establish a more feasible and profitable reverse network design, all

these decisions should be taken into consideration together, therefore our model optimize them simultaneously and provides more realistic approach than previous models. Secondly, we develop three hybrid metaheuristic algorithm to solve our model. Simulated Annealing (SA), Tabu Search (TS) and Genetic Algorithms (GA) are studied as metaheuristics and their hybrid algorithms with Variable Neighborhood Search (VNS) are compared in terms of both solution quality and computational time. The novel characteristic of the proposed solution approaches can be summarized as follows: When developing the hybrid algorithms, the nested heuristic solution methodology is developed. The outer loop, which is related to find the best locations of collection and distribution centers (CDCs) to collect the returns and satisfy the orders among a set of candidate sites, is solved with metaheuristics. However, feasible financial values, price and incentive amounts, are seeking by local search to gain maximum profit with the present location of CDCs in the inner loop.

The organization pattern of this thesis is as follows. In Chapter 2, we review studies related to the reverse logistic network design location problems. In this chapter, we also review the related studies which implemented SA, TS, GA as a solution method for reverse logistic models. Then in Chapter 3, we define our problem in detail and give some insights on computational complexity. We also illustrate the mixed integer nonlinear programming (MINLP) formulation of our problem. Next in Chapter 4, we describe the solution method that can be used to find optimum values to reach the maximum profit and proposed metaheuristic algorithms are explained in detail. In Chapter 5, we give the details about data generation and results of computational studies are interpreted. Finally in Chapter 6, we give a general summary of the thesis and provide some future research directions.

Chapter 2

RELEVANT LITERATURE

In the literature, there are a lot of studies related to facility location models because of the fact that facility location is one of the most important element of strategic planning for many private and public firms and gives advantage for competition. Many operational and logistic decisions are associated with the facility location, therefore large capital outlays are involved when decision maker selects the site which are executed in the long term. For this reason, some of the models studied in literature are designed for optimizing the real supply chain cases and they have received significant attention. Location decision is not a crucial elements of a successful forward network design but also effects reverse network design. That's why model with location decisions are examined in reverse logistic design problems. Pricing decision is as important as location decisions to develop a profitable supply chain design. However, in the literature, not much has been written about models for competitive location including price decisions.

Apart from the modelling the network design, various exact and heuristic approaches have been developed as solution methods for the location problems. The most common solution techniques used in the literature are metaheuristic algorithms due to computational efficiency.

In this chapter, some of the related literature is discussed within three main categories. In Section 2.1, we review the location model which is mostly related to our model. Then, in Section 2.2, many of the reverse logistic network design problems are reviewed chronologically. Finally, in Section 2.3 the metaheuristic approaches used in our study and reverse logistic problems found in the literature are discussed.

2.1 Facility Location and Allocation Model

Location allocation models have been studied for more than a century and started from Weber's location problem in 1909(3). In this simplest problem, he assumes that a single server is to be located to meet a set of discrete demands while minimizing the transportation cost.

Discrete facility location problem deals with the determination of the sites where new facilities are to be established within a finite set of candidate locations. In general, there are two common and widely used location allocation problems. The simplest and one of the most studied location problem is p-median problem which is introduced by Hakimi(4). This problem analyzes which p facility are to be selected among the potential locations to minimize objective function while satisfying all customer demands. P-median problem is mostly applied for public type of facilities like hospital, ambulance, firefighting, etc.In p-median problem, facility setup costs are same through all potential location sites. Mladenovic et al.(5) give a complete survey about metaheuristic approaches applied to p-median problem which is classified as NP-hard. They survey classical heuristics and implementation of interchange local search as well as metaheuristics. Metaheuristic methods are briefly described and analyzed in the study as Tabu search , Variable neighborhood search (VNS), Genetic Algorithm, Scatter search, Simulated Annealing, Heuristic concentration, Ant colony optimization, Neural Networks, Decomposition heuristics, Hybrid heuristics. As a result of the survey, according to empirical results, metaheuristic approaches have been more successful than the earlier used methods i.e constructive heuristics and local searches according to the solution quality obtained in large instances. Moreover, the efficiency on computational time when solving the p-median model with metaheuristics is stated in the study.

The other type of the location allocation model considers the fixed facility location costs, therefore number of facilities to be established becomes an important decision. This problem is known as uncapacitated facility location problem (UFLP) and extensively studied in literature. Facility location problem is similar to the p-median problem except that the number of facilities to be located is originated within the problem. Revelle et al.(6) survey a number of the important problems in facility location including UFLP. Moreover, an ex-

tensive study can be reached in Mirchandani and Francis's book(7).

Both type of the models are aimed to minimize the assignment costs by allocating each customer to the nearest candidate sites and have common characteristics which are single product, single-period planning horizon, same deterministic parameters like demand of each customer, cost of the product and transportation costs, facility and location allocation decisions. However, the both models are not sufficient to fulfill the needs of realistic location allocation decisions. For this reason, recently many extensions to basic models have been proposed. Therefore, the location allocation problem has grown with a lot of type and classification.

2.2 Reverse Logistic Network Design

Increasing environmental, economic and legislative concerns lead to a wide recognition of the remanufacturing of used products into new ones in literature and in practice. Reverse logistics network design establishes the relationship between two markets: the market releasing used product and the market producing and distributing new product. According to market type, two main classes of papers have been studied about the network design with product recovery. The first class only deals with the former market and seeks the most efficient solution of the problem how to retrieve products from end users. This strategy is known as *reverse logistics.* On the other hand, the second class manages the both streams of products as flow through and back from the customers. This strategy is also known as *closed loop logistics* and examines the relations and group of activities between these two markets as a total system.

The well-known definition of reverse logistic is done by Roger (8)as:

"...the process of moving goods with cost effective flow of raw materials, from the point of consumption to the point of origin for the purpose of capturing value otherwise unavailable or for the purpose of proper disposal."

In the past years, many case studies have been analyzed and modelled comprehensively

for reverse logistics network design including recycling problems. We review a few more recent studies whose objectives and settings are related to ours chronologically as follows.

Recycling networks of iron and steel by-products in German industry have been analyzed by Spengler et al (9) .They complete one of the earlier study addressing the reverse logistic and propose a modified mixed integer linear programming (MILP) warehouse location model in order to maximize the total marginal income by decision of location and capacity level of selected recycling operations .

Sand recycling from construction waste in Netherlands is considered as an important problem and has been studied by Barros et al.(10). The authors propose two-level capacitated facility location model analyzing multiple scenarios in order to minimize total fixed and transportation costs. This problem is revisited under demand and supply uncertanity by Listes and Dekker (11).They connect the idea of the uncertainty characteristics of product recovery with reverse supply chain network design. Then, they establish a generic multi-stage stochastic programming model for these two level network design to find the optimum location of the storage and cleaning task and analyze the expected performance of the scenarios with given probabilities in Netherlands. They use the branch and cut algorithm to find the solution using the commercial software, GAMS. By this study, they find the most suitable solution among alternative scenarios which are described by field expert.

Louwers et al. (12) study a facility location allocation model for reusing of carpet waste as a case study and analyze its two applications, one in Europe and the other in the United States of America.

Krikke et al.(13) concern the remanufacturing processes of copiers. Two possible locations, one in the Czech Republic and the other in Venlo, are discussed throughout the study and the strategically most appropriate one is discussed for the processes preparation and re-assembly processes. To optimize total operational cost of the reverse logistic network design, they study three scenarios with a MILP model and compare them to re-design the reverse logistic network.

Product recovery is also important in electronics industry. Therefore, there are many papers that analyzed the logistic network design for original equipment manufacturer (OEM). Jayaraman et al.(14) have examined the electronic equipment recovery closed loop network design with multi-product capacitated warehouse location MILP to seek the optimal number and location of remanufacturing facilities while minimizing the total costs.

Fleischmann (15) emphasizes a continuous network design model using a MILP model for product recovery. To solve the generic model, he develops an heuristic algorithm and gets satisfactory results for larger size problems.

Shih(16) develops a MILP model to design the reverse network flow of computers and home appliances in Taiwan.

Jayaramann et al. (17) study a MILP reverse distribution model for type of products which can be end-of-life, commercial returns or other reverse functions i.e. recycling, remanufacturing, reuse, refurbishing to minimize the total transportation and fixed costs of opening facilities. To solve this NP-hard problem, they develop a heuristic procedure and optimal solutions are found for a significant proportion of generated test problems.

Schultmann et al.(18) examine reverse supply network of spent batteries in Germany. They develop a conventional two level MILP facility location model for the closed loop supply chain and develop a flow-sheeting system with a mathematical tool to simulate the complex chemical reactions of spent batteries used in the steelmaking industry.

Kusumastuti et al.(19) study a reverse logistic network design. To reflect the real life conditions, uncertainties are taken into account in the study and a multi-period, multi-objective MIP model is proposed. To solve the complex problem, spanning-tree based genetic algorithm is developed to find non-dominated solutions. Different scenarios are examined in the study to determine the best-preferred reverse logistic network design.

Shue et al.(20)consider the integrated logistics operational problems of green-supply chain management. The model adds two important features to existing literature: Firstly the model formulated a linear multi objective programming model to optimize the operations. In other words, it is not case specific. Secondly, the model contains the enforcement of corresponding governmental regulations. They proposed a composite multi objective programming model to maximize both manufacturing chain- and reverse chain- based profit considering the their weighted average in a closed loop manner. The well known Taiwanese notebook computer manufacturer's data are used in numerical example part of the study. Therefore applicability of the study is approved.

Nagurney et al. (21) develop a multitiered e-cycling network equilibrium model for reverse supply chain network of electronic waste and recycling. They analyze the optimizing behavior and describe main equilibrium conditions. Moreover, the authors study on qualitative equilibrium pattern properties to indicate the convergence of algorithmic scheme and study on numerical examples to support their algorithm.

Schultmann et al.(22) apply a symmetric capacitated vehicle routing problem (VRP) formulation with one depot in order to design the reverse material flow of end-of-life vehicle treatment in Germany. They propose a problem tailored model and analyze the real case data with different scenarios due to uncertainties in the closed loop supply chain. The results show the applicability of the model while comparing different scenarios in terms of distance and cost.

Biehl et al.(23) focus on reverse logistics of the US carpet industry. They examine the main factors that affect the design of carpet reverse logistic supply chain and exhibit the impact of these factors as well as environmental factors. Designing a simulation model, they analyze various scenarios' performance and discuss the critcal factors for design of a carpet reverse logistic supply chain.

Lu et al.(24) design a closed loop logistic system as two-level location problem with three types of facilities to be located. They propose a MIP model for the closed loop prob-

lem as an uncapacitated facility location model and develop a Lagrangian heuristic based algorithm to minimize the total cost of the problem. Improving the lower bounds with proposed model, they obtain quite good lower bounds for their model. The applicability of the model is tested by conducting the numerical examples on data adapted from classical test problems taken from OR-Library.

Ko et al.(25)propose a multi-period, two-echelon, multi commodity network design model for designing an integrated logistic network for $3PLs$. Their model considers forward and reverse flow simultaneously and this creates the main difference as compared to the existing location models. Because the model is MINLP and NP-hard, they develop a genetic algorithm based heuristic to find the minimum cost. In the proposed heuristic, simplex transshipment algorithm is added to Genetic Algorithm procedure to allocate the customers into open facilities with considering the capacity constraints. To illustrate the efficiency of the proposed heuristic, an example problem is analyzed and numerical results are presented at the end of the study.

Lieckens et al. (26) investigate on the extended version of the reverse logistic model currently found in literature. They proposed a MINLP model concentrating on high degree of uncertainties and some dynamic aspects. An improved version of the GA known as Differential Evolution Technique is applyied to solve the problem using the queueing relationships for interarrival and process time distribution.

Listes (27) designs a generic stochastic closed loop reverse logistic model and studies on a decomposition approach to solve this model, which is based on the branch-and-cut procedure and known as the integer L-shaped method. Computational studies are performed on alternative scenarios and also results show a consistent performance efficiency of the proposed method.

Salema et al. (28) analyze the design of reverse logistic network under uncertainty on demands and returns as a closed loop manner. They design the multi-product, capacitated, forward and reverse recovery networks modelling as a MILP model and solve it using

standard branch and bound techniques. The proposed model can be thought as an extend version the model proposed by Fleischmann (15). The mentioned problem is uncertainty-driven, therefore, scenario based approach is used. In the study, three different scenarios are examined depending on the demand and return rate. Finally, applicability and performance of the model is tested on an illustrative example.

Kusumastuti et al.(29) present a case study to redesign the service network for a computer company in Singapore. They establish a multi-horizon facility location allocation model to minimize total closed loop logistics cost. The result of the model clarifies the locations of new distribution centers which are also used to collect and consolidate the parts before sending out for repair.

Sheu (30) proposes a linear multi-objective programming (LMOP) model to optimizes both flow and reverse supply chain operations of nuclear power generation. In addition to literature, the model optimize both costs corresponding to nuclear power generation and induced nuclear waste due to governmental regulation. The case of Taiwan nuclear power generation is presented as the numerical illustration in this paper to evaluate the applicability of the model. Moreover, the closed loop model also emphasize on the operational risks induced in both forward and reverse supply chain processes.

The most relevant study with our problem is done by Aras et al. (31). They develop a discrete facility location-allocation model to find the predetermined number of collection centers and the optimal financial incentive values for different return types. For this reason, they formulate the collection center location MINLP problem as well known p-median problem including the incentive decision to be offered for each quality. Because of the NP-hard structure of their model, as similar to our solution methodology, they propose a heuristic procedure using two nested loop to find the maximum net profit. In the outer loop, Tabu Search algorithm is developed to find the optimal location of the collection centers in the solution space. For each location set, in the inner loop, Nelder Mead Simplex Search is called in the algorithm to obtain the best incentives resulting to net profit. Nelder-Mead Simplex Search (1965) is a derivative-free direct search method which provides shrinking

the simplex search towards desired point until reaching some desired bound. Finally, the performance of the proposed heuristic is analyzed in terms of solution quality and computational time on randomly generated instances.

Pati et al. (32) discuss multiple objectives (economic, environmental and quality) of waste management in reverse supply chain network. They propose a mixed integer goal programming (MIGP) model to figure out the relationship between these objectives with working on multi-item, multi-echelon and multi-facility environment. To assent the viability of the model, paper recycling system of India is analyzed as a case study through the paper.

Kim et al. (33) apply a well-known vehicle routing approach for designing the reverse logistic network of the end-of-life consumer electronic goods in South Korea. To find the minimum distance of transportation for each of four regional recycling centers, an integer programming model is formulated. Tabu Search algorithm is applied to find the optimum solution for the recycling problem. The computational results of the study show that the proposed method is more efficient and gives better results than the existing methods.

Min et al. (34)point out the problem about the location and allocation of repair facilities for 3PLs. They provide a MINLP model for reverse logistics network design to provide minimum total logistic costs solution. Similar to other studies, they develop a heuristic procedure to solve this NP-hard problem. They generate the data for experiment and the usefulness of the proposed GA is validated by its application to generated multi-period, multi-commodity problem.

Elsayyed et al. (35) study on a multi-period, multi-echelon, forward-reverse logistics network design under risk. To maximize the total expected profit, they develop a stochastic MILP model. They assume that in the first customer zones the demand is stochastic and second customer zones the demand is deterministic. The different model parameters are used to analyze the effect of mean demand and return ratio. Various scenarios of the single item model assures that mean demand and return ratio for a given capacity of the network have significant effect on expected profit.

Zhou et al. (36) design the reverse logistic network considering the repairing issue as well as the mostly studied remanufacturing options. They propose their MILP model which is the extended version of Fleishmann et al(15). To illustrate the model, they go on three cases taking into account the repairing, remanufacturing and both of them simultaneously using a standard MILP solver.

Du et al. (37) focus on the bi-objective optimization model for reverse logistic networks that deal with the returns requiring repair service. The aim of the proposed MILP model is minimization of both the overall costs and total tardiness of cycle time. The model optimizes the facility arrangement among potential locations and transportation flow between customer areas and service facilities. In the study a hybrid solution approach which is composed of scatter search, the dual simplex method and the constraint method is used. Three test data are generated for the computational study randomly. According to the computational results, they compare these two objective function served network structures, one of is centralized and the other is decentralized.

Mutha et al. (38)designs a nine echelon network system for reverse logistic and remanufacturing using new and old product modules. To illustrate the model, a single returned product with ten modules are considered and test data are generated. The proposed model is solved using GAMS software. The cost components of the model are carefully examined and the most effective ones are pointed in the study to make the model more beneficial for decision maker's strategic decisions.

Cruz-Rivera et al. (39) point out the closed loop supply chain management of end-of-vehicles in Mexico. For this purpose, they formulate an uncapacitated fixed charge facility location problem to minimize the total cost resulting from fixed facility cost and transportation cost. According to the percentage of coverage, three different scenarios are discussed through out the paper and Lagrangian relaxation is used to find the optimum number and location of end-of-vehicle collection facilities.

Xanthopoulos et al. (40) design a methodology named as "Multi-criteria Matrix" to help the manufacturers in terms of deciding the best end of life alternatives. This matrix identifies the product by ranking the criteria such as environmental, economical, quality and quantity. The use of methodology is demonstrated through the case of electronical and electronic products.

Grunow et al. (41) focus on the waste of electrical and electronic equipment network in Denmark. They present three MILP models to assist governmental agency to generate alternative solutions. The constructed basic static model is extended to basic dynamic and fairness models in order to satisfy further considerations.

G.Kannan et al. (42) study battery recycling because of governmental, environmental, economical and social reasons and formulate a multi echelon, multi period, multi product MILP model for closed loop supply chain of battery industry. With this model, they minimize the total supply cost from raw material purchase cost to disposal and recycling costs. They adapt the Genetic Algorithm to find the best solution of the larger sized problems. To discuss the efficiency of the proposed algorithm, for smaller problem size, they compare the results taken from GAMS and the proposed algorithm.

Kusumastuti et al.(43) formulate a multi-objective and multi-horizon reverse logistic network design problem for product recovery. To solve their model including uncertainties, spanning tree based Genetic Algorithm is proposed. Several scenarios are studied to reflect real-life conditions and using simulation the best network design model is obtained.

Pishvaee et al. (44)deal with an integrated forward and reverse logistic network design. They propose a bi-objective MILP model to minimize the total cost and maximize the responsiveness of a logistics network. Different from the literature, to solve the proposed multi objective MILP model, they use Memetic Algorithm which integrates the local search into GA to improve the intensification of the local search. They compare their model with the multi objective Genetic Algorithm models of Altiparmak et al. (45)who propose a new solution procedure based on Genetic Algorithm to find the set of Pareto-optimal solutions

for multi-objective SCN design problem. According to the results, the algorithm proposed by Pishvaee (44)outperforms the algorithm proposed by Altiparmak et al.(45) in terms of average ratio of Pareto-optimal solutions obtained.

In a very recent study, Salema et al. (46) formulate a multi-period, multi-product network MILP model to simultaneously design forward and reverse networks. In the model, strategic and tactical decisions are taken into account by modelling the time with managerial perspective. The performance, adequacy and applicability of the model is verified by the satisfactory results of the illustrative case.

Kara et al. (47) formulate a simulation model for reverse logistics network for collection of end of life products in Sydney Metropolitan Area. Simulation model assists to present the relationship between the activities in the network. Furthermore, the reverse supply chain studies, and closed loop supply chain models are also introduced throughout the study.

In addition to reverse logistic case studies and quantitative models, closed loop logistics design have been considerably studied in the literature.

Beamon and Fernandes (48) develop a closed loop supply chain model to determine the location and sorting capability of warehouses and collection centers as well as deciding the quantity of flow between the sites. For this purpose, they formulate a multi-period integer programming to minimize investment and operational costs.

Chouinard et al. (49) demonstrate a new approach and information support system to connect the reverse logistic activities with new systems. The new organization and information system performance are evaluated on the warehousing problem of Quebec Rehabilitation Center.

Inderfurth (50) studies a closed loop supply chain on product recovery system with existence of uncertainty. He proposes a multi-period stochastic programming logistic model for reverse supply chain and observe the effects of cost efficient dynamic decision making

on product recovery. Moreover, he supports his thesis by making sensitivity analysis on a numerical example.

Kumar and Malegeant (51) study the reuse-a-shoe program of Nike to inspect the value created by the partnership of manufacturer with an eco-non-profit community organization. They focus on the business benefits in terms of service/market oriented or environment/ safety oriented. Furthermore, they analyze the benefits of strategic alliances between manufactures and eco-non-profit community organization by creating a website as Throwplace.com which connects both businesses and donors.

Srivastava (52) proposes a multi-product, multi-echelon, profit maximizing reverse logistics and value recovery network model to represent the real life situations from collection to first stage of remanufacturing. The model focuses on the disposition decision for various products in the Indian context and optimize the location-allocation and capacity decisions for facilities using GAMS optimization tools.

Lee et al. (53) focus on the logistic network design for end-of-lease computer products recovery. They formulate a deterministic mixed integer programming model and propose a two-stage heuristic approach because of the computational complexity of the proposed NP-hard model while minimizing the total logistics cost. The model is broken down and analyzed into two subproblems as location allocation and revised network flow problem. Tabu Search algorithm with short term memory is considered as a metaheuristic approach to find the optimal shipping for the returned products. Finally, numerical studies are constructed to assent the capability of the model and efficiency of the algorithm in terms of both solution quality and computational complexity and experiments result in high-quality solutions.

Fuente et al. (54) deal with an integrated supply chain management which combines the new processes taken from reverse supply chain into existing processes of forward supply chain. The detailed analysis are done by working on a real case study of a metal-mechanic company. The metal-mechanic company already has forward and reverse logistic, therefore,

it is convenient to be analyzed under IMSCM. Through the study, the relationships between manufacturers, suppliers and clients are carefully examined to coordinate the operations in both chain.

2.3 Related Heuristic Algorithms

Heuristic algorithms has been widely used and successfully implemented for location problems due to NP-hardness of the problems. There are several references in literature that describe the use of heuristic and metaheuristic methods in the location problems' research field. Since we use hybrid algorithm of SA, TS and GA with VNS, it is beneficial to briefly present with the studies which are partly related to our problem and in which heuristic procedures have been used as a solution procedure. We review the literature under two subgroups: studies that metaheuristic have been implemented and studies that other type of heuristic procedures are implemented.

2.3.1 Metaheuristic Algorithms implemented Location Problems

Because many of the combinatorial optimization (CO) problems are NP-hard, we can not use the exact method to solve these problems completely in a polynomial time due to their complexity. Without guaranteing optimality, heuristic methods seeks a good solution at a reasonable computational time. Metaheuristic is a computational technique used to optimize combinatorial optimization problems by iteratively improving the candidate solution according to given measure of quality. Metaheuristic methods do not guarantee the optimality or the solution can not be proven as optimal however, they usually converge to a good solution. In our solution procedure, we focus on mainly three metaheuristics therefore, the relevant literature are studied under three subgroups.

Location Problems analyzed by Simulated Annealing Algorithm

In literature, there are an extensive number of studies using SA as a metaheuristic algorithm to solve NP-hard location problems. More relevant studies are analyzed as follows.

One of the earliest location problems solved by SA is studied by Selim et al. (55). They focus on a clustering approach for a non-convex p-median problem and discuss the solution

of the problem. SA approach, parameter settings, advantages and disadvantages of the method are discussed in detail and successful results are obtained through the study. They conclude the study by finding good results which converge to global optimal.

Another location problem solved by SA is studied by Murray et al. (56). They develop three heuristic approaches SA ,TS and Interchange procedures to solve the operational forest planning problem based on the work of Nelson and Brodie (57). It is the first study which discuss SA as a solution approach to a location problem. The results illustrate the efficiency of the algorithms by finding near optimal solutions in relatively short amounts of computer time. In the latter studies, Murray et al. (58) reanalyze SA approach with p-median and maximal covering location problems. They integrate interchange heuristic procedure to SA. Retrieving data from Beasleys OR-Library (59), the proposed algorithm concludes better results in terms of solution quality and encourages further studies.

Kincaid (60) studies a discrete version of two noxious facility location problem. In the study, TS and SA implementation for the proposed model is analyzed and as a result, it is conclude that TS outperformed SA.

Chiyoshi et al. (61)join basic features of the vertex substitution method of Teitz and Bart (62) with the general methodology of SA to solve the classical p-median problem. For computational studies,the standard test data are retrieved from the Beasley's OR-Library (59). The results of the study reveals that the algorithm works well with the worst gap to the optimum as 1.62%.

The classical NP-hard p-median problem is also analyzed by Levanova et al. (63). Ant system(AS) and SA algorithm are proposed for the solution method. The results reveal that the SA algorithm outperforms AS, however, both of the developed algorithms find good solutions with less than 2% error margin.

Arostegui et al. (64) focus on various Facility Location Problems (FLP)under time-limited, solution-limited, and unrestricted conditions. They search the dominant heuristic

for each type of facility location problems by comparing the relative performance of TS, SA and GA. According to the computational studies , performance of TS is better in most cases.

Location Problems by Tabu Search Algorithm

Tabu search(TS) was initially proposed by Glover (1977), evolved in Glover (1989, 1990) and Glover et al. (1993). TS is a meta-strategy iterative procedure that search the optimal solution by extending the neighborhood with a local search strategy and prevents getting trapped in local optima. To avoid being caught, it prevents the moves that result to visit previous solutions by naming them as "tabu" throughout predetermined number of iterations (tabu tenure). TS is a widely used metaheuristic method for location problems and results in very accurate solutions. There are also a lot of studies which use TS to solve location problems. Some of them can be analyzed as follows.

Rolland et al. (66)propose a new solution to p-median problem.They establish TS algorithm and compare it with well-known interchange and a recent hybrid heuristics. According to computational study, TS outperformes other heuristics.

Rosing et al. (67) compare the quality of two metaheuristic methods with a well-known p-median problem. TS and Heuristic Concentration are examined throughout the study and Heuristic Concentration discovers superior solutions.

Michel et al. (68)study on uncapacitated warehouse location problem and propose a simple and robust TS algorithm. In the study, computational experiments reveal that the TS algorithm outperforms the GA which is a heavily used method to solve this model.

Al-Sultan et al. (69)proposed TS to solve UFLP and tested their model on some standard problems in literature. For all test problems, optimal solutions are found more quickly than the existing models in literature.

Location Problems by Genetic Algorithm

Genetic Algorithm is a metaheuristic that has been widely used extensively to optimize location problems. The potential advantages of the GA over other heuristics are first studied by Hosage and Goodchild (70)in 1986 and since then it has been applied to many problems for three decades. The discrete space location problems,particularly well known p-median problem is solved by GA in the study. Due to being first, the heuristic algorithm structure and essential features are described, therefore the possibilities and applicability of GA to location problems are studied in detail.

Genetic Algorithms and Evolutionary Strategy are combined to find the locations of facilities and allocations of customers to facilities dealing with the service capacity of the facilities by Gong et al. (71). The problem is considered in two levels to minimize the total distance: Location part is optimized by proposed hybrid algorithm and for allocation part, Lagrange Relaxation is adopted. The computational experiment are run with randomly generated data and efficiency of the algorithm is demonstrated by comparing the result of the alternative location allocation heuristics suggested by Cooper. (72)

A comprehensive set of location problems is extensively studied by Jaramillo et al. (73). For the uncapacitated and capacitated fixed charge problems, the maximum covering problem, and competitive location models of medianoid and centroid problems are examined with GA and results are compared in terms of the optimal value, the average CPU solution time, the percentage of deviation from optimal for the best solution found by the Lagrangean heuristic.Computational experiments reveal that for the first three class of location problems, GA spends much more time but leads to better solutions than Lagrangean heuristic. However, for competitive models, GA outperforms the Lagrangean with regards to both computation time and solution quality. Moreover, they conclude the study that the GA should not be used for the capacitated location problems with fixed cost.

The well known capacitated p-median problem is also studied by Correa et al. (74). Unlike Hosage and Goodchild (70), they develop a new heuristic including hypermutation operator and study on a real p-median problem. Hypermutation operator mutate the in-

dividual gene resulting in the best fitness that it can possibly be by comparing all possible combinations. The two version of the GA, with and without hypermutation operator, are compared with the TS. The newly generated GA outperforms the TS algorithm in terms of solution quality.

Optimal location search under complex situations is examined by Li et al. (75). They propose GA to solve a spatial search problem to find optimal allocation of the n facilities (hospitals) according to the population data and transportation conditions which are retrieved from Geographical Information systems (GIS). A densely populated city, Hong Kong,data is used to make computational analysis. The efficiency of the proposed algorithm is tested by comparing the Neighborhood Search (Openshaw and Openshaw 1997) and SA algorithms. The GA shows much better performance than Neighborhood Search and SA to solve this multiple objective problem, where a site is searched to maximize the population coverage, minimize the total transportation costs and minimize the proximity to roads.

Neema et al. (76) consider p-median problem and develop two hybrid GA approaches with different replacement procedures. To compare the efficiency of the proposed algorithms, they compare the results with the traditional alternating location-allocation heuristics and simple GA by numerical simulation. The numerical results indicate that the hybrid algorithms always obtain the best solution than location allocation and simple GA for all problem sizes and less computational efforts.

The most recent study about the GA based location modelling is done by Sasaki et al. (77) to optimize current and future health planning. To reduce the response time and contribute to higher survival rates, they design the model to optimize the location of public health facilities in Niigata and solve the model using GA approach.

2.3.2 Location Inventory Models

In the traditional approach, the inventory studies are likely to be ignored the location decisions and its related costs. The cost of operational inventory, transportation and shortage are likely to ignore in studies when demand uncertainty exists. However ,in reality, inventory management is one of the most important component of reverse logistic network to minimize cost while guaranteing the desired service level. Quality, quantity and timing of the returned product can not be controlled by the producers and is the main difficulty of inventory management. In addition, when take back activities are carried without legislative forces, the location of the CD mainly affects the incentive of product holder to return their used product.

In literature, there are some studies that work on the relation between location decisions and inventory management. The major classification can be made according to return flow as deterministic versus stochastic model. In our study, we use the deterministic return rate approach and the deterministic EOQ type model is firstly offered Schrady (78) in 1967. U.S Naval Supply Systems Command stock holding problem with repairable items are examined with his model.He assumes constant demand and return rates and fixed lead times. Considering fixed setup cost for orders and remanufacturing process and linear holding costs, he offers the continuous supplement and substitution policies with fixed lot sizes serving demand from remanufactured products as possible. The substitution policy provide optimal procurement and repair service. Nahmias et al. (79) extend the Schrady's model for finite remanufacturing rate and conclude their work by determining optimal remanufacturing batch size depending on optimal ordering quantities. Another extension of the Schrady's model is studied by Mabini et al.(80). Multi-item system with stockout service level constraints are proposed in Mabini's model and numerical solution methods are mainly focused on.

Camm et al. (81) develop an uncapacitated facility location formulation for Procter and Gamble Company with integer programming framework to locate the DCs and assign the selected DC's to customer zones. The objective function is to minimize the total cost, composed of material handling costs, inventory costs, transportation costs, duties associ-

ated with border crossings, and so forth, with maintaining the maintain current levels of customer service.

Another related study introduced by Daskin et al. (82) proposes a location inventory model and its solution methodology. In this study, they consider DC location model which includes working inventory and safety stock costs as well as the economies of scale that exist in the transport costs from suppliers to DCs. An alternative solution procedure is developed for the model which was previously proposed by Shen (83)and Shen et al. (84) and solved by first recasting it as a set partitioning problem and then solving the resulting model using column generation. Daskin et al. (82) develop a Lagrangian solution algorithm and the efficiency of the algorithm is assented by numerical examples.

In the literature, the two most related studies to our research are done by Wojanowski et al. (85) and Aras et al. (31). They mainly focus on pricing issue while designing an optimal drop-off facility network. Wojanowski et al. (85) concentrate on the forward supply chain using continuous modelling approaches and they determine the sales price of the product by focusing the deposit-refund policy to maximize firm's profit. In this policy, the sales price is known by the customers including the deposit amount which will be paid back when customer takes back the used product to a collection facility. Therefore, customers' willingness with regards to purchasing and returning product are related to a stochastic utility choice model.

On the other hand, Aras et al. (31) concentrate on reverse supply chain and try to optimize only the return decision of product holders rather than purchasing decision under deposit-refund requirements. Another difference between these studies arises from their proposed pick-up policies. In Wojanowski et al. (85), customers have to bring their used products to the collection centers, however, Aras et al. (31) develop a pick-up policy with vehicles of limited capacity.

Differently from literature, we combine the pricing and inventory decisions with closed loop supply chain optimization. In other words, while Wojanowski et al. (85)and Aras et

al. (31) considers only forward or reverse supply chain, we study both of them as integrated forward and reverse supply chain. Therefore, beside the location and allocation decisions in logistic network, our model optimizes the sales price and incentive amounts as well as cycle time of orders, simultaneously. To the best of our knowledge, the proposed model have not been studied in literature before. Moreover, to solve our NP-hard problem, we develop three hybrid metaheuristics and compare their quality. Again, to the best of our knowledge, such a detailed comparison about the hybrid metaheuristics performances has not been studied in literature.

Chapter 3

LOCATION AND PRICING PROBLEM

In this study, we aim to determine the optimal price for new products and optimal incentive amounts to collect the right amount of used product while designing an integrated forward and reverse network. The details of the model is clarified in Section 3.1.

3.1 Problem Definition

In our problem, we have customer zones each with a certain population and various deterministic demand rates. We also have a production facility at a certain location. Products will to be sent to distribution centers (DCs) from the factory, and sold at a price P to the customers. The end-of-life products' owners (ELPO) return their used products to collection centers (CCs) and get incentive as R. In our problem, the forward and reverse supply chain flows coincide at CC and DC, therefore, the two centers are integrated and work as both collection and distribution centers (CDCs). Both used and new products are held at CDC and shipping to and from the factory at a certain time intervals. Our aim is to decide on where to open CDCs as well as the amount of P, R and collection cycle time.

In our model, we focus on a closed supply chain by receiving inspiration from UFLP. The classical UFLP is formulated as a set covering model and composed of the set of potential facilities V_1 and the set of customers V_2 and seeks the optimal location of the facilities to be established in order to satisfy all customer demands for the corresponding product.

In this study, the model is constructed in a closed loop supply chain setting, therefore both forward and reverse flows are taken into account. In literature, the first step of the forward supply chain begins with procuring the related raw materials from the suppliers to perform the production. Then, the finished goods are distributed by different channels in order to meet the customer demands. Afterwards, to capture the remaining value after the products lives end, the used products are collected from the final destination of the forward supply

chain or customers bring the used products to CCs. Collection of used products forms the first step of the reverse supply chain. Finally, they are sent to the centralized remanufacturing facility in order to gain their residual value and to use them in production process of the new goods.

In our model, the end-of-life products' owners (ELPO), who are the customers of their forward supply chain, transport the products from her coordinates to the nearest opened CC. Therefore, the tendency of customers leaving the used product to the the nearest opened CC is associated with the distance between the CC and the ELPO's location. In our model, this tendency is directly proportional with distance between the ELPO and the CC. Even though the popularity of using environmentally friendly products has been growing, many countries still lack any government legislation, social responsibility and environmental awareness about taking back activities of the used products. For this reason, in our problem setting, a financial incentive is offered to the ELPO to persuade them to return their products. The incentive amount is restricted by an upper bound which is the value will be gained after the remanufacturing process of a unit of used product. Despite the fact that the quality levels of returned products depend on many different parameters,i.e usage rate, condition etc., in the model, we do not differentiate the returned products by its quality, and we assume that quality levels of all returned products are the same. Therefore, we offer same incentive value to all customers for a unit of returned product and assume that we gain same value from a unit of return.

Like the relationship between the incentive value and distance in reverse flow, price and accessibility of a new product have also significant connection in forward flow. It is obvious that, in our competitive marketing world, the increase on price results the decrease on demand rate. Similarly, the increase on distance between distribution center(DC) and customer zone affects demand rate negatively.

Based on our problem settings, when a returned product is dropped to CDC, it can not be sent quickly to the centralized recycling facility due to economical deficiencies. Therefore, inventory holding cost of both new product and used product at CDCs take a part in the model. In the light of this, the fixed ordering cost and total inventory holding cost is balanced by using the well known EOQ framework. In the literature, EOQ is used to

determine the optimal ordering quantity to meet the demand while minimizing total cost associated with purchase, delivery and storage. Since we assume that the base demand and return rates are constant over time and these parameters are affected only remoteness, price and incentive values,i.e they are not stochastic, applying the EOQ model becomes suitable to manage inventory part of the model.

3.2 MINLP Formulation of the Location and Pricing Problem

We present the MINLP formulation to maximize the total gain i.e profit, by determining the optimal number and location of the CDCs as well as the amount of price and incentive offered. The model parameters and decision values are defined as follows:

Parameters:

D_j =expected total demand of customer j$\in V_2$ in a cycle time

B_j =expected total used product of customer j$\in V_2$ in a cycle time

S =amount of gained from unit used product

F_i =fixed cost of a CDC located at candidate site i$\in V_1$

k =the multiplicative constant between price and demand

b =the multiplicative constant between incentive and return

A_{io} =fixed ordering cost of a CDC located at candidate site i$\in V_1$ from the factory

hp_i =unit holding cost of new product in the CDC located at candidate site i$\in V_1$

hr_i =unit holding cost of used product in the CDC located at candidate site i$\in V_1$

t_{ij} =distance travelled from customer j$\in V_2$ to CDC located at candidate site i$\in V_1$

Decision variables:

In the model, we have financial, inventorial and location decisions, therefore, we can categorize the variables as follows:

Financial variables as:

P =optimum price value offered for unit product

R =optimum incentive value offered for unit used product

Location variables as:

$$y_i = \begin{cases} 1 & \textit{if the CDC is opened at candidate site } i \in V_1 \\ 0 & \textit{otherwise} \end{cases} \tag{3.1}$$

$$x_{ij} = \begin{cases} 1 & \textit{if customer } j \epsilon\ V_2 \textit{ is served by the CDC located at candidate site } i \in V_1 \\ 0 & \textit{otherwise} \end{cases} \tag{3.2}$$

$$z_{ij} = \begin{cases} 1 & \textit{if product holder in } j \in V_2 \textit{ is assigned to CDC located at candidate site } i \epsilon\ V_1 \\ 0 & \textit{otherwise} \end{cases} \tag{3.3}$$

T_i =returned product collecting cycle time from CDC located at candidate site i$\in V_1$

In our model, customers bring to products to CDC transportation costs. As a result of this, demand and return rates are affected negatively by the distance. The correlation between demand and the distance travelled by the customer is defined by a parameter, namely α_{ij}. In the same way, the return amount is correlated by distances travelled by ELPO with a parameter, β_{ij}. According to the modification explained, the problem is modelled as follows:

$$\begin{aligned} \Pi = max \sum_{i \epsilon V_1} \sum_{j \epsilon V_2} P(D_j e^{-kP}) x_{ij} \alpha_{ij} + \sum_{i \epsilon V_1} \sum_{j \epsilon V_2} (S-R)(B_j(1-e^{-bR}) z_{ij} \beta_{ij} - \sum_{i \epsilon V_1} F_i y_i \\ - \sum_{i \epsilon V_1} [\frac{A_i o}{T_i} + (\sum_{j \epsilon V_2} [(D_j e^{-kP}) x_{ij} \alpha_{ij} T_i \frac{h_i^p}{2} + B_j(1-e^{-bR}) z_{ij} \beta_{ij} T_i \frac{h_i^r}{2}])] y_i \end{aligned} \tag{3.4}$$

subject to:

$$\sum_{i \in V_1} x_{ij} = 1 \qquad \forall j \in V_2 \tag{3.5}$$

$$\sum_{i \in V_1} z_{ij} = 1 \qquad \forall j \in V_2 \tag{3.6}$$

$$x_{ij} \leq y_i \qquad i \in V_1 \quad j \epsilon V_2 \tag{3.7}$$

$$z_{ij} \leq y_i \qquad i \in V_1 \quad j \epsilon V_2 \tag{3.8}$$

The objective function (3.4), labelled as Π, consists of mainly two parts to maximize the total profit. The first part calculates the sum of the net profit resulting from selling new products and gaining the residual value of the collected end of life products. The net profit is found by multiplying the amount of total customers' demand with optimum price P and amount of total return with net unit gain. The amount of payment given by the company as incentive should be discounted from the value of the used product. As a consequence, net unit gain of company becomes $(S - R)$. In our problem, amount of demand decreases proportionally by price as $(D_j e^{-kP})$. Likewise, amount of return collected from ELPO increases by incentive amount as $(B_j(1 - e^{-bR}))$.

Besides the pricing outcome, due to transportation responsibilities, demand and return amounts are negatively affected by α_{ij} and β_{ij} parameters.α_{ij} and β_{ij} parameters are used as:

$$\alpha_{ij} = \frac{1}{1 + t_{ij}} \qquad i \epsilon V_1 \quad j \epsilon V_2 \tag{3.9}$$

same as:

$$\alpha_{ij} = \beta_{ij} = \frac{1}{1 + t_{ij}} \qquad i \epsilon V_1 \quad j \epsilon V_2 \tag{3.10}$$

However, they can also be different without effecting the model.

The second part of Π (3.4) is named as inventory part and composed of the total cost related to fixed ordering and inventory holding. EOQ model is used to determine optimum ordering and holding cost, therefore all EOQ assumptions are valid in our model. The only difference arises from the rate of demand and return which are not constant, instead pro-

portional to price and incentive. Moreover, UFLP allocation constraints are valid for our model. For allocation decisions, constraints (3.5) and (3.7) guarantee that each customer has to be served by only one of the opened CDCs. With the same logic, constraints(3.6) and (3.8) guarantee that each product holder has to be assigned to only one of the opened CDCs.

The formulation of the problem is a mixed integer nonlinear model ($MINLP$). Location of the CDCs and allocation decisions in the model is similar to the well-known uncapacitated facility location ($UFLP$)which makes our problem $NP-hard$. Additional incentive and pricing variables enlarge and complicate the model.

3.3 The computational Complexity of the Problem

The proposed model can be considered as combination of the three subproblems as: location, pricing and inventory. One of the subproblems of our model is well known UFLP problem and it is a widely studied location problem in literature. UFLP has all typical difficulties of mixed integer programming therefore it is NP-hard.(86). Since UFLP is known as NP-hard, with additional inventory and pricing decisions, our model is also NP-hard. Exact algorithms for this problem is analyzed in some papers (87). In the exact methods, it is needed to count all possible solutions to get optimal results. Therefore, solving UFLP even for small problem instances is a very difficult procedure to handle. As the size of the problem (N) increases the solution space including both feasible and infeasible solution increases as n!. If problem is a simple p-median problem, solution space becomes narrower and the total number of combinations to select p facility out of N candidate location reduces to:

$$\frac{N!}{p!(N-p)!} = \binom{N}{p} \tag{3.11}$$

Inventory management decisions in EOQ part result in a convex behavior and our model returns to a mixed integer nonlinear programming (MINLP) and gets additional complexity.

Additionally, our model consists price(P) and incentive(R) decisions. Assuming the price

of the new product and incentive of the used product offered to ELPO are bounded by a pre-specified interval, the solution space contains

$$\sum_{p} \sum_{r} N! \times p \times r \tag{3.12}$$

solutions. In such a big solution space, finding the optimal solution will be a very time consuming task.

Because of the computational complexity, in this study, we develop three hybrid meta-heuristics method to find near optimal solutions with small deviations in a reasonable time interval as described in the next chapter.

Chapter 4

SOLUTION METHODOLOGIES

In this section, all the solution methods are discussed to find the optimum values of decision variables to maximize the total profit. As stated in Section 3.3, our model is strongly NP-hard. The problem can not be solved with an exact algorithm in reasonable time limits. For this reason, three different heuristic methods are developed to produce high quality solutions in reasonable time limits.

Before discussing the proposed algorithms, to take an advantage, we reduce the number of decisions variables in Π 3.4 by substitution method. This simplification is explained using EOQ formulation as below

Observation:

Optimum cycle time of each CC is estimated by using EOQ formulation. To determine the optimum cycle time, T_i, the first derivative of the objective function is derived with respect to T_i and the cycle time of each CDCs is found in terms of other parameters as follows:

$$T_i = \sqrt{\frac{2A_{io}}{\sum_{j\epsilon V_2}[(D_j e^{-kP})x_{ij}\alpha_{ij}h_i^p + (B_j(1-e^{-bR}))z_{ij}\beta_{ij}h_i^r]}} \tag{4.1}$$

Because of computational complexity, embedding T_is found from (4.1) in the objective function (3.4)provides advantage by reducing the total number of variables in the model. After inserting the T_i's into the objective, number of variables in Π reduces to three independent decision variables P, R , y_i and two dependent decision variables x_{ij} and z_{ij}. Finally, objective function takes a form as:

$$\Pi = max \sum_{i \epsilon V_1} \sum_{j \epsilon V_2} P(D_j e^{-kP}) x_{ij} \alpha_{ij} + \sum_{i \epsilon V_1} \sum_{j \epsilon V_2} (S - R)(B_j(1 - e^{-bR}) z_{ij} \beta_{ij} - \sum_{i \epsilon V_1} F_i y_i$$
$$- \sum_{i \epsilon V_1} \sqrt{2A_{io} \left(\sum_{j \epsilon V_2} ((D_j e^{-kP}) x_{ij} \alpha_{ij} h_i^p + B_j(1 - e^{-bR}) z_{ij} \beta_{ij} h_i^r) \right)} \quad (4.2)$$

According to 4.2, the behavior of P and R in Π is observed as follows showing a unimodular property.

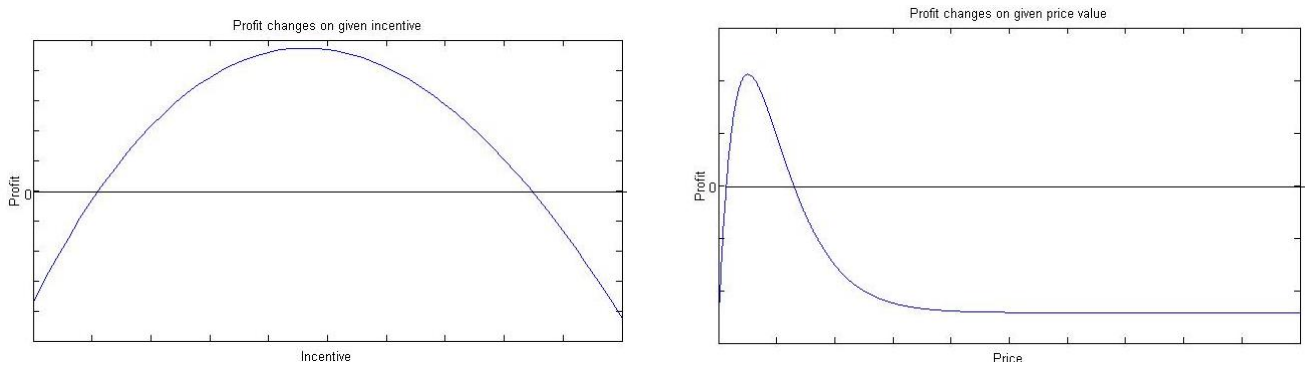


Figure 4.1: Profit function vs P and R

4.1 Exact Methods

4.1.1 Commercial Solver

In order to see how an exact method performs on our model, we try to solve all test instances with a commercial user. Therefore, we analyze our model with MINLP solver in ILOG CPLEX 11.2 and set a time limit of 3600 seconds (1h) for each run.

4.1.2 Enumeration Method

For small problem instances, enumeration method can provide the global optimum by visiting all feasible solutions. In the formulation of the model, x_{ij} and z_{ij} variables are dependent to y_i decision which means a customer can only be served or assigned by a CDC if the CDC is opened. For this reason, when the number and location of the opened CDCs are known,

it is optimal to assign each customer to the nearest CDC because of minimizing the transportation cost. In our model, we assume that the capacity of CDCs are enough to hold all products and returns collected from assigned customers. Therefore, we consider the allocation part of the model as in UFLP and it is feasible and also optimal to assign customers to nearest CDCs.

In the enumeration method, all possible combination of y_i arrays is examined and optimal assignment matrix are constructed as x_{ij} and z_{ij}. After determining of the location and allocation, all combination of price and incentive values, in a predetermined interval, is tried and optimal values are found which maximize the total profit. For our problem size, enumeration method can be applied only for 10x10 data sets. As number of customers and potential facility locations increase, the solution space of our problem increases rapidly with 2^n where n is the number of candidate CDC location sites.

Due to computational complexity, none of the exact methods solves our model in reasonable time limits. To overcome this computational complexity, we develop metaheuristic algorithms which is described in the next sections to find near optimal solutions. To claim the applicability and efficiency of the heuristics, an upper bound is found for each test problem. Upper bounds are calculated by using piecewise linearization method as a result of nonlinear behavior of the model.

4.2 Piecewise Linear Approximation

Optimum cycle time of each candidate CDC are restructured in terms of price and incentive values as seen in 4.1 and take a quadratic form. To deal with nonlinearity, which results in extra complexity in the objective, we suggest to use piecewise linear approximation for quadratic part of our model. By this way, we find a lower bound for inventory part and this results to an **upper bound** for objective value.

There are possible approximation schemes depending on the number of segments used in the piecewise linear functions. As the number of segments increases, the error resulting from the approximation decreases. However, as the number of segments increases, the computational complexity also increases. Therefore, we consider the trade off between complexity

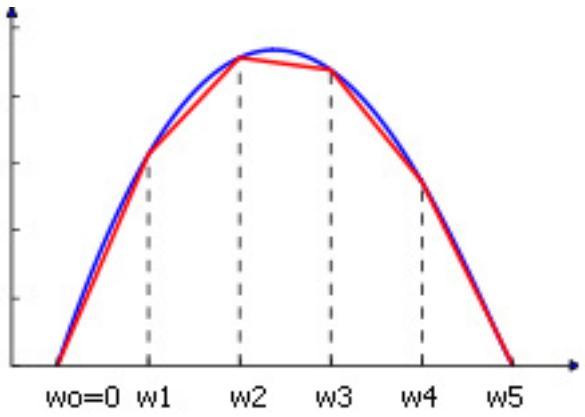


Figure 4.2: Piecewise linear approximation scheme

and allowable error range.

The formulation and additional constraints of the piecewise linearization are as follows:

$$\sum_{i \epsilon V_1} 2A_{io} \left(\sum_{j \epsilon V_2} ((D_j e^{-kP}) x_{ij} \alpha_{ij} h_i^p + B_j (1 - e^{-bR}) z_{ij} \beta_{ij} h_i^r) \right) = \sum_{k \epsilon N} w_k \times 2A_{io} \left(\sum_{j \epsilon V_2} ((D_j e^{-kP}) \alpha_{ij} h_i^p + B_j (1 - e^{-bR}) \beta_{ij} h_i^r) \right) \tag{4.3}$$

where;

$$w_k \leq v_k + v_{k-1} \qquad \forall k \in K \tag{4.4}$$

$$\sum_{k \in K} w_k = 1 \qquad \forall k \in K \tag{4.5}$$

$$\sum_{k \in K-1} v_k = 1 \qquad \forall k \in K - 1 \tag{4.6}$$

$$w_1 = v_0 = 1 \tag{4.7}$$

4.3 Metaheuristics Method

In literature Neighborhood Search, Tabu Search and Lagrangian Relaxation heuristics are successfully applied for solving UFLP. Besides the similarities between our model and UFLP, including the inventory part makes our model unique in literature. For this reason, we hybrid three well-known metaheuristic methods which are Simulated Annealing, Tabu Search and Genetic Algorithm with Variable Neighborhood Search as a solution procedure in this study and named as SAVNS, TSVNS and GAVNS.
All algorithms are composed of two nested loops. At the outer loop of our heuristic, we use metaheuristic methods to find the best locations of CDCs that will be opened to maximize the profit when decreasing the inventory costs. Location of the CDCs are also important for customers whose demand and return rate are negatively affected by distance and the impact is clearly observed on total profit. In the inner loop, the best price and incentive values are searched within a bounded set by using nonlinear optimization gradient search methods.

Before describing the algorithms in detail,we explained the solution representation used in all solution procedure below.

4.3.1 Solution representation

In the proposed algorithms, the optimum locations of opened CDCs (y_i) are determined.Therefore, a solution of y_i is represented as a binary vector in which the i^{th} entry shows the activity position of the i^{th} CDC. For example, the representation of a solution with possible CDC locations i= 1, 2, 3, ... ,10 as [1, 0, 0, 1, 0, 0, 1, 0, 1, 1] denotes that the CDCs located in 1, 4, 7, 9 and 10 are selected to established CDCs and all customers have to be satisfied with these 5 centers.

4.3.2 Proposed SAVNS Algorithm

Simulated annealing algorithm is one of the the most popular local search metaheuristic because of being easy to implement, having powerful convergence properties and allowing

downhill moves in order to escape from local optima. Using the basic principles of thermodynamics, it searches global optimum for discrete optimization problems based on a random local search technique.

In the proposed hybridization, Simulated Annealing algorithm is embedded in Variable Neighborhood Search algorithm. VNS causes explorative search and diversification by swapping between different neighborhood structures. In algorithmic framework, VNS changes a little bit due to the fundamental idea of SA in which the solution can be accepted even its objective function value is worse than the current solution to avoid being trapped in local maxima. Basically, SA is memoryless which means the neighborhood is created depending only on the current solution in given neighborhood structure. In each iteration, within a defined neighborhood structure, the best solution is sought. Possibility of accepting a nonimproved solution depends on a special probability function based on "Boltzmann" distribution. The optimization of the model is based on Physical Annealing analogy. Temperature of the system reduces in each step to reach thermal equilibrium. The cooling rule is appropriately selected with the aim of tuning the balance between diversification and intensification. When the algorithm stops, i.e termination conditions are met, the best solution found so far within the previous neighborhood is moved to the next neighborhood. The next neighborhood should also yield progressive diversification of the search. Therefore, good strategies should be used carefully to exploit different properties and characteristics of search space.

The proposed SAVNS algorithm steps are as follows:

Notation:

y_o: the initial solution of y_i array with dimension as $|V_1|$,

y: the current solution of y_i array ,

y^{iter}: the best solution of y_i array in the iteration

$y*$: the best known solution for y_i array,

f_y: total profit gained by location array y.

$nonimp_K$: number of consecutive nonimproved solution in neighborhood K.

$maxnonimp_K$:maximum allowable number for consecutive nonimproved solution in neigh-

borhood K.

$iter_K$:iteration number in neighborhood K.

$maxiter_K$:maximum allowable number iteration in neighborhood K.

Algorithm 1: SAVNS

Inputs:D_j, B_j, A_i, F_i, hp_i, hr_i, $\alpha_{i,j}$, k, b.

0 : Select a cooling schedule, initial temperature T_o, final temperature Tf,
the set of neighborhood structures N(K) for $K = 1, .., K_{max}$.
1 : Set $f(y^*) = 0$.
2 : Generate initial solution y_o. ▷ Initialization
3 : y← y_o.
4 : Construct $x_{i,j}$ and $z_{i,j}$
5 : Determine optimal P and R values by using exhaustive search.
6 : Calculate f(y).
7 : f(y*)← f(y) and y* ← y. ▷ Initialization ends
8 : Set k=1.
9 : **while** $(K < K_{max})$ **do**
10 : Set $iter_K = 0$ and $nonimp_K = 0$.
11 : **while** $(iter_K < maxiter_K)$ and
$(nonimp_K < maxnonimp_K)$ **do**
12 : Construct N(y).
13 : Find $y^{iter} \epsilon$ N(y) as with the optimal P and R values
using exhaustive search.
14 : $\delta = f(y^{iter}) - f(y*)$.
15 : **if**($\delta > 0$) **do**
16 : f(y*)← f(y) and y*← y.
17 : $nonimp_K = 0$
18 : $K = 1$.
19 : **else**
20 : Generate random n uniformly in the range (0,1).
21 : **if**$(n < exp(-\delta/T))$ **do**
22 : f(y*)← f(y) and y*←y.
23 : **end if**
24 : $nonimp++$
25 : **end if**
26 : Update T depend on cooling schedule.
27 : **end while**
28 : K++;
29 : **end while**

Output: Return the incumbent which serves maximum profit found by the algorithm for given instance as f(y*), y*,optimal P and R values.

Initial Solution

For the proposed model, initial solution is generated randomly within the candidate location sites. The upper bound of the number of opened nodes (CDC) is not restricted. However, to satisfy the demand of all customers, in other words not to violate the demand constraints, it is ensured that at least one CDC must be opened in initial solution and this

facility serves all customers because it can produce and ship unlimited quantities of the commodity.

Neighborhood Structure

The neighboring solutions are generated applying combinations of add, drop or swap moves to current solution. The proposed SAVNS has three neighborhoods to visit different solution spaces. In the first neighborhood, as a move operator,we apply 1-0 exchange to current solution. In other words, an **add move** which opens node i if the node is not opened or a **drop move** if the node is opened is assigned to the current solution. By changing the status of i^{th} coordinate, n different solutions are visited. The second neighborhood is composed of changing status of two randomly selected coordinates simultaneously,i.e applying twice 1-0 exchange in the same iteration. Neighborhood is generated by applying a predetermined number of 1-0 exchanges to two randomly selected nodes. Finally,in the third neighborhoods, status of randomly selected three nodes are changed by applying 1-0 exchange to them. All neighborhood in SAVNS provide different solutions and they are arranged adequately in terms of diversification manner.

Cooling Schedule

Cooling schedule designates the reduction in the rate of temperature and it is important for the success of SA. Keeping rate of nonimproved solution acceptance high or low at the beginning is one of the most critical decision for SA. Moreover, deciding the trend of temperature reduction is related to whether demanding diversification or not in search space. To compromise, various schedules are tried as linear, exponential and hyperbolic ones. The formulation of these schedules used in experiments are as follows.

Notation:

i:cycle that the algorithm runs within

t_i =temperature of cycle i

t_o =initial temperature

N:maximum cycle number

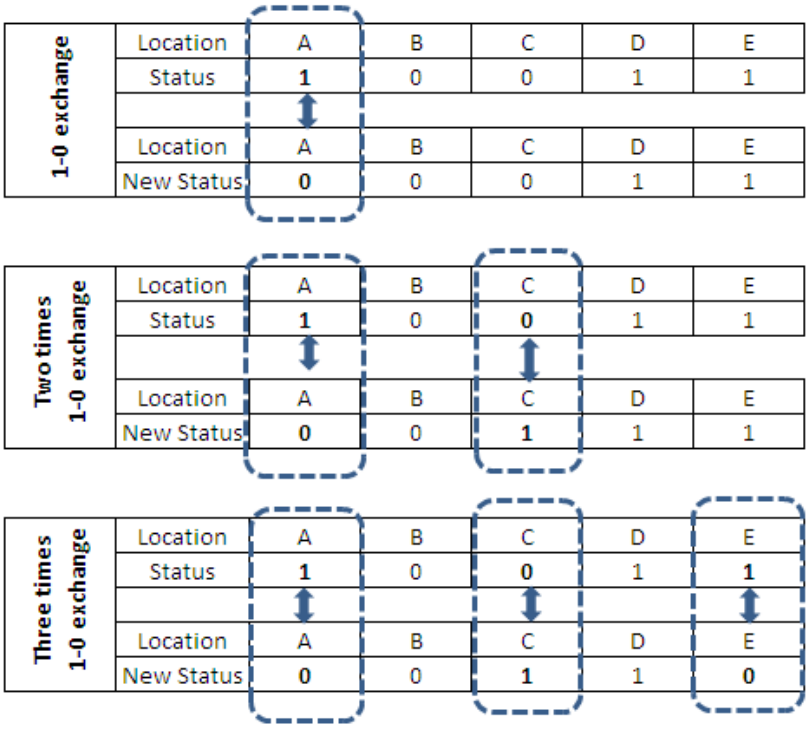


Figure 4.3: SAVNS neighborhood move operators

To observe the effect of linear reduction, the following cooling schedule is used:

$$t_i = t_o - i \times \frac{t_o - t_N}{N} \tag{4.8}$$

This cooling schedule reduces the temperature in each step by decreasing it with a fixed value $\Delta_t = t_o - t_N/N$. In other words, linear cooling schedule has same rate of cooling through the search.

To observe the effect of temperature reduction as exponentially, the following cooling schedule is used:

$$t_i = t_o \times exp^{-\frac{1}{N}\frac{t_o}{t_N})} \tag{4.9}$$

This cooling schedule spends less time at higher temperatures. It decreases at lower temperatures rapidly and spend more time in there.

Additionally, we use an hyperbolic cooling schedule using a function as follows:

$$t_i = \frac{t_o - t_N}{2} x(1 - tanh(\frac{10 * i}{N} - 5)) + t_N \tag{4.10}$$

If we compare the Eqn. 4.9 and 4.10, Eqn.4.10 spends more time at higher temperatures, decrease gradually and spends less time in lower temperatures.

All the schedules described above balance between diversification and intensification of the search space. For example, at the beginning of search, the reduction might be rapid and then show a behavior being nearly constant in order to make the algorithm converge to a local maximum at the end of search.

The initial and final temperatures are also adapted to particular problem instances.

Termination Criteria

In the proposed algorithm, we use two different stopping criteria. The first one is the maximum number of iterations allowed in a neighborhood. This number should be adjusted according to the size of the neighborhood. The second one is the maximum number of allowable iterations that incumbent solution does not improve. Both criteria are used at the same time and reaching one of them is enough to terminate the current neighborhood and to move to the next one. In each neighborhood, the allowable number of non-improved solutions determined again with respect to the size of the neighborhood. Non-improved solutions are accepted to allow diversification and escape from local maxima. However, new neighborhood begins with a solution that is the best solution in the previous neighborhood up to now. In other words, best found solution is also kept in each neighborhood when allowing to move toward a worse solution.

The flow chart of the SAVNS algorithm is given in Figure 4.8

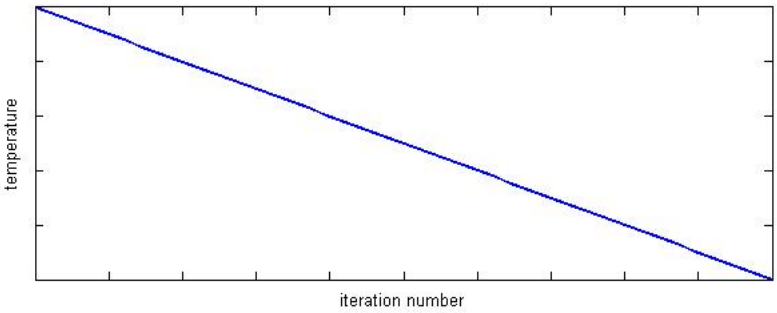


Figure 4.4: Linear Cooling Schedule

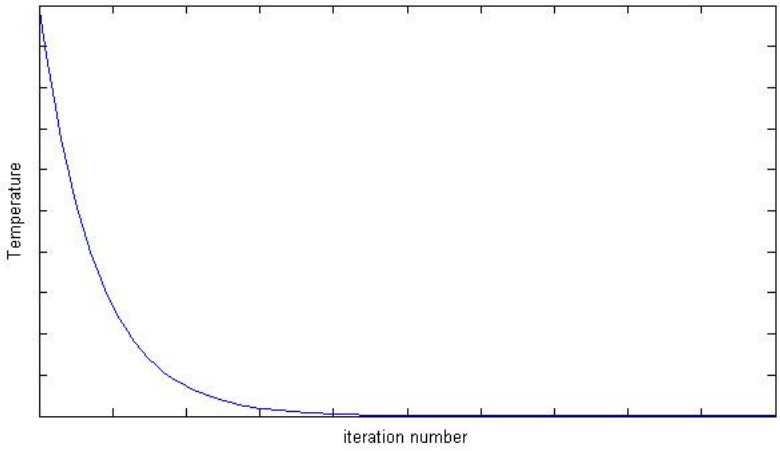


Figure 4.5: Exponential Cooling Schedule

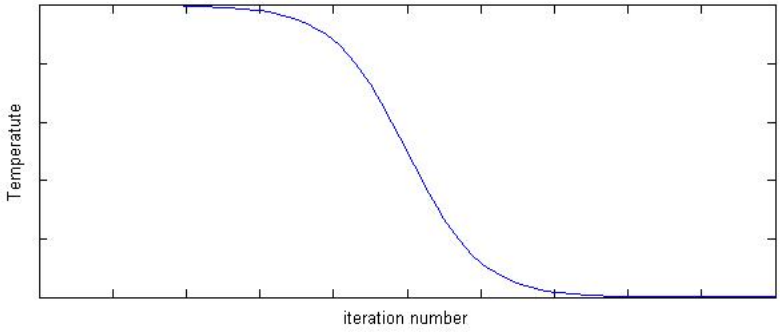


Figure 4.6: Hyperbolic Cooling Schedule

Figure 4.7: Various Cooling Schedules used in SAVNS Algorithm

4.3.3 Proposed TSVNS Algorithm

Tabu Search algorithm implements an explorative search by using the history to escape from local maxima. In the proposed algorithm, we embed TS into VNS algorithm. This hybridization helps the solutions trapped at a local optima and allow it to visit larger neighborhoods with systematical change. Deterministic search is used to expand neighborhoods by finding the best neighbor of incumbent solution in sequential order. Basically, VNS jumps from the incumbent solution to a new one if and only if a better solution has been found and does not forbid any moves. However, Tabu Search employs tabu restrictions to prevent visiting earlier selected solutions throughout the iterations. In tabu list, short term memory is implemented to keep track of most recently visited solutions. At each iteration, the best solution which has the maximum profit is sought by applying local search within allowed set in the current neighborhood. Then, the best solution is added to the tabu list and one of the solutions that were already in the tabu list is removed in a FIFO order. Thereafter, the search continues around the best solution. When the termination criteria are met, the algorithm stops and the best solution found so far within the previous neighborhood is moved to the next neighborhood the same way as SAVNS.

Notation:

y_o: the initial solution of y_i array with dimension as $|V_1|$,

y: the current solution of y_i array ,

y^{iter}: the best solution of y_i array in the iteration

$y*$: the best known solution for y_i array,

f_y: total profit gained by location array y.

$nonimp_K$: number of consecutive nonimproved solution in neighborhood K.

$maxnonimp_K$:maximum allowable number for consecutive nonimproved solution in neighborhood K.

$iter_K$:iteration number in neighborhood K.

$maxiter_K$:maximum allowable number iteration in neighborhood K.

Algorithm 1: TAVNS

Inputs:D_j,B_j,A_i,F_i,hp_i,hr_i,$\alpha_{i,j}$, k, b.
0 : Select $tenure_K$, the set of neighborhood structures N(K) for $K = 1, .., K_{max}$.
1 : Set $f(y^*) = 0$.
2 : Generate initial solution y_o. ▷ Initialization
3 : y← y_o.
4 : Construct $x_{i,j}$ and $z_{i,j}$.
5 : Determine optimal P and R values by using exhaustive search.
6 : Calculate f(y).
7 : f(y*)← f(y) and y*← y. ▷ Initialization ends
8 : Set $k = 1$.
9 : Set $iter_K = 0$, $nonimp_K = 0$, $tabulist_K = 0$. 10 : **while** $(K < K_{max})$ **do**
11 : **while**$(nonimp_K < maxnonimp_K)$ **do**
12 : Construct N(y).
13 : Find y^{iter} ϵ N(y) as with the optimal P and R values i using exhaustive search.
14 : Calculate f(y^{iter}).
15 : **if**$(f(y^{iter}) \leq f(y*)$ **do**
16 : $nonimp_K++$
17 : **if**$(nonimp_K \geq maxnonimp_K)$ **do**
18 : K++
19 : Set $nonimp_{K+1} = 0$, $iter_{K+1} = 0$, $tabulist_K$=0.
20 : **end if**
21 : Control $tabulist_K$.
22 : Select the best y that is not flagged as tabu.
23 : f(y*)← f(y) and y* ← y
24 : Update $tabulist_K$. 25 : **else**
26 : Control $tabulist_K$
27 : Select the best y that is not flagged as tabu
28 : f(y*)← $f(y)$ and $y*$ ← y.
29 : Update $tabulist_K$.
30 : **end if**
31 : **end while**
32 :**end while**
Output: Return the incumbent which serves maximum profit found by the algorithm for given instance as f(y*), y*,optimal P and R values.

Initial Solution

For TSVNS, initial solution is again constructed randomly as in SAVNS.

Neighborhood Structure

To find the best neighborhood structure, we compare three different neighborhood structures for TSVNS. The first structure has two neighborhoods. In the first neighborhood, as in SAVNS algorithm, 1-0 exchange is applied to current solution and all possible neighbors are visited. The number of opened facilities increases or decreases by one with the help of 1-0 exchange. The second neighborhood structure is composed of 1-swap moves. 1-swap move relocates a closed facility with an opened one in the current solution. All pairs of candidate locations that generate a different solution from current one is visited through

the local search to change the sequence of the facility positions. The possible neighbors that provide the same order i.e.if both of the candidate locations in a pair are opened or closed, are prevented to visit in the proposed algorithm. By this way, each solution in a neighborhood assures a different combination of the opened CDCs. This heuristic is named as $TSVNS_{2NS}$.

The second structure ,$TSVNS_{2NL}$, is like the $TSVNS_{2NS}$. It also has 2 neighborhoods and the first neighborhood establishment and move is same with $TSVNS_{2NS}$ as 1-0 exchange. However, in the second neighborhood, 3-swap moves relocates the 3 consecutive facilities. All 3 combinations of current solution are visited in this neighborhood, therefore second neighborhood size is much bigger than the second neighborhood of $TSVNS_{2NS}$.By the 3-swap moves, the number of opened facility number remains constant, however, locations of the opened facilities change.

Unlike $TSVNS_{2NS}$ and $TSVNS_{2NL}$, the third structure is composed of three neighborhood and named as $TSVNS_{3N}$. The first two neighborhood is same with $TSVNS_{2NS}$. However, in the third neighborhood, twice 1-0 exchange is applied to the current solution. By this way, unlike the others, the number of opened facilities can be increased or decreased by two.

Tabu definition

When a facility, named as facility i, is involved in obtaining a new solution which can either be added or removed, ith node position can not be changed again for a certain number of iterations, i.e.labeled as tabu moves and will not be allowed for a certain number of iterations such as the size of tabu tenure. Using dissimilar neighborhood structures result in keeping tabu restrictions in different ways. In the 1-0 exchange type moves, keeping the one node position, the exchanged, is sufficient as tabu move. In the second kind of neighborhood, a pair of different positioned coordinates are swapped. The coordinates resulting in the best solution in the whole neighborhood are labelled as tabu moves and in tabu list, the pair of

coordinates are kept. Therefore, the 1-swap move using the coordinates used previously can not be applied until they are non-active in tabu list. **Recency-based memory** is used in the TS heuristic which forbids moves towards the most recent visited solution. Size of tabu tenure is also selected according to the problem size.

Aspiration Criteria

Aspiration criteria are algorithmic devices that cancel the status of a move and allow to revisit the solution even though this move is restricted by the algorithm. We also use the classical aspiration criteria of allowing a move even if it is tabu, if it results in a solution which provides a profit value better than the incumbent one.

Termination Criteria

One of the most difficult decisions in the proposed algorithm is to determine the termination criteria. Like SAVNS, we use two termination criteria at same time as maximum number of iterations and the maximum number of allowable non-improved solution.

The flow chart of the TSVNS algorithm is given in Figure 4.10

4.3.4 Proposed GAVNS Algorithm

Unlike the trajectory methods discussed above as Simulated Annealing and Tabu Search, Genetic algorithm is a population based method which deals with a set (i.e. a population) in every iteration of algorithm rather than with a single solution. Studying different and a lot of solutions at the same time explores the search space and results in a parallel random search with centralized control. As the name of the algorithm suggests, the sexual reproduction inspire the algorithm which combines two parent strings into an offspring. Each parent, or actually named as chromosome, represents a solution of the problem. Combination of the genes form the chromosomes and N combinations of the chromosomes generate a population. Biological evolution operators, mainly crossover and mutation are used to create a new generation from current population. Selection of new population depends on their fitness values. **Fitness values** are calculated upon the objective values of the chromosomes. As we observe in the nature, the creatures, which adapt themselves to the changing environmental conditions, are more likely to survive compared to others. This also holds for the Genetic Algorithm with solutions of higher fitness values having greater probability to get selected in the next generation. Integrating the VNS into GA also causes diversification in search space of each neighborhood by changing the formation of new generations while affecting the crossover and mutation structures. As mentioned in previous algorithms, in each iteration within a defined neighborhood structure, we firstly generate the offsprings (i.e new generation) using genetic operators. Then, using fitness values of the chromosomes a new population is selected deterministically. When the termination criterion of one neighborhood is met, the algorithm jumps to next neighborhood to diversify and explore the search space. Neighborhoods are designed in nested strategy and enlarges as proceeded to the end of the algorithm.

Notation:

$nonimp_K$: number of consecutive nonimproved solution in neighborhood K.

$maxnonimp_K$:maximum allowable number for consecutive nonimproved solution in neighborhood K.

$iter_K$:iteration number in neighborhood K.

$maxiter_K$:maximum allowable number iteration in neighborhood K.

cp:crossover probability.

mp:mutation probability.

popsize:population size.

Algorithm 1: GAVNS

Inputs:D_j, B_j, A_i, F_i, hp_i, hr_i, $\alpha_{i,j}$, k, ba.
0 : Select popsize, crossover cp_K and mutation mp_K probabilities
k and b values,arrange neighborhood structures N(k) as $k = 1, .., k'max4$ according to $num_n onimp_i ter_k$.
1 : Generate an initial population of size popsize. ▷ Initialization
2 : **while** $(num_i ter < maxiter)$ and
$(nonimp < maxnonimp)$ **do**
3 : Construct $x_{i,j}$ and $z_{i,j}$ matrix for each individual
4 : Determine optimal P and R values for each individual by using exhaustive search.
5 : Compute f(y) for each individual and find the best individual as y* and f(y*)
6 : **if** $(nonimp_K < maxnonimp_K)$ **do**
7 : **for**(popsize/2) **do**
8 : Select two parents randomly
9 : Generate cp randomly (cp=U[0,1])
10 : **if**($cp > cp_K$) **do**
11 : Recombine the two parents depends on crossover operator of $neighborhood_K$.
12 : **else**
13 : Duplicate the two parents as offsprings 14 : **end if**
15 : **end for**
16 : **for**(popsize) **do**
17 : Generate mp randomly for each offspring (mp=U[0,1])
18 : **if**($mp > mp_K$) **do**
19 : Mutate the offspring depends on mutation operator of $neighborhood_K$.
20 : **else**
21 : Keep the offspring as unchanged
22 : **end if**
23 : **end for**
24 : Construct matingpool from parents and offsprings
25 : **for**(popsize*2) **do**
26 : Construct $x_{i,j}$ and $z_{i,j}$ matrix for each individual in matingpool
27 : Determine optimal P and R values for each individual by using exhaustive search.
28 : Compute f(y) for each individual and find the best individual as y^{iter} and $f(y^{iter})$
29 : **if**($f(y^{iter} > f(y*))$ **do**
30 : Set $nonimp_k = 0$.
31 : **else**
32 : Set $nonimp_k = nonimp_k + 1$.
33 : **if**$(nonimp_k >= maxnonimp_k)$ **do**
34 : Set $k = k + 1$.
35 : **end if** 36 : Set $nonimp_k = nonimp_k$.
37 : **end if**
38 : **end for**
38 : Construct the new generation according to fitness values.
39 : $iter = iter + 1$.
40 : **end if**
41 :**end while**
Output: Return the incumbent which serves maximum profit found by the algorithm for given instance as f(y*), y*,optimal P and R values.

Initial Population Generation

A population of individuals are created randomly from candidate solutions. Generation of initial population is really important in order to find better solutions. Sometimes, ini-

tial population can be found by other heuristic methods to reach more quickly to global optimum rather than using a random start. However there is a possibility to get trapped in premature convergence, therefore we construct the initial population randomly. Another important decision for starting GA is adjusting the population size. Population size should be determined according to the size of the problem at hand. If population size is too small, the algorithm may not explore enough. On the other hand, a large population can cause a highrate of changein population and prevent to concentrate on better solutions.

Crossover Operators

In molecular biology and genetic, offsprings are generated by crossover which exchanges genetic information between parents to transfer the best gene that they inherently have. In molecular biology and genetics ,crossover is made in order to generate offsprings by exchanging genetic information between parents. Instead of move operators in SA and TS, GA uses crossover and mutation operators. The crossing over provides mechanism for transferring the characteristics of parent chromosomes to their offspring. It is hoped that some of the offsprings will be produced through properly selected crossover operations and will have the best characteristics of the parents, and that will most likely to be retained for further generations.

Similarly SAVNS and TSVNS, three different crossover operators are used to visit different solution spaces in GAVNS and diversify the search. In the first neighborhood, uniform crossover mask is generated randomly as the same length of the chromosomes. Parts of chromosome or namely parents, to be exchanged is determined by the mask. The offspring1 is produced by taking the bit from parent1, if the corresponding mask bit is equal to 1, or taken from parent2 if the corresponding mask is equal to 0. After the crossing over, two offsprings will be created from mated parents.

In the second neighborhood, 1-point crossover is applied to mated parents in which parents are cut from a randomly selected position and their chromosome strings are swapped from their tails. By this way, two offsprings are generated and partly similar to their parents.

In the third and last neighborhood 2-point crossover is used. The pair of parents' chro-

mosomes are cut from the randomly selected two point and the middle segments of the parents' strings are swapped. Different crossover probabilities are tried during the tuning. If crossover probability of the mated parents is less then the set value, in other words, if crossover is not applied, the selected parents are duplicated in order to generate the two offsprings.

The working manner of these crossover operators can be seen in Figure 4.11, 4.12 and 4.13.

Mutation Operators

Mutation operator changes one or more genes value in a corresponding chromosome and results in a new one. Like crossover operators, in GAVNS, we apply different mutation operators to different neighborhoods. After applying crossover, some genes of offsprings with small probability (i.e.mutation probability) are altered in order to diversify the search space. For each offspring, a mutation probability is created randomly and determined whether it undergoes a mutation or not. If randomly generated probability is less than the mutation probability of the neighborhood in which the algorithm runs, the offspring is exposed to mutation.

The solution representation , also chromosomes, are composed of binary genes. Therefore, flip-bit operator is suitable for GAVNS. In the first neighborhood, a random number is generated between 1 and n, and that much of randomly selected gene status are exchanged from "1" to "0" or vice versa. In the second neighborhood, inversion mutation technique is applied to the chromosomes which will be mutated. The sub-tour is selected randomly. In the third neighborhood, head and tail of the chromosome is changed from a randomly selected gene position. The mutation probability of the neighborhoods increases so as to diversify the search space by the end of the algorithm to escape from local maximum.

The working manner of these mutation operators can be seen in Figure 4.14

Formation of the next generation

After the generation of mating pool, the chromosomes that will form the next generation is selected according to their fitness value. As fitness value, we use the total profit that can

be generated by the chromosome. The survival of the chromosomes depends on the selection strategy used through the algorithm. In literature, three different selection strategies as fitness ranking, tournament selection and roulette wheel selection are described as follows:

Fitness Ranking (FR):

FR sorts the individuals according to their fitness values and selects the parents having best fitness until the pool is full. By this way, next generation is composed of only strong individuals and this may lead to premature convergence.

Tournament selection (TR):

Similar to fitness ranking, TR selects the individuals by generating tournament between randomly selected two individuals and copied the better (winner) one to the pool. The difference between TR and FR is that while TR compares only two individuals, FR compares all the individuals' fitness values. Randomizing in TR provides divergency in the algorithm.

Roulette wheel selection(RWS):

Unlike FR and TR, RWS creates next generation with a stochastically selection process. In this process, fittest individuals will tend to have a greater chance of survival than weaker ones and form the mating pool for the next generation. In other words, fittest individuals has larger share of the roulette wheel. Size of the population is equal to the number of times the roulette wheel spanned. The individuals are sorted according to their fitness values and based on the randomly selected point, an individual is selected to form the next generation. After each selection, the probabilities are calculated again in order to provide greatest chance to fittest individuals among remaining.

Through the GAVNS, the population of next generation is selected in enlarged sampling space using roulette wheel selection. Enlarged sampling space is composed of both parents and offspring and the basic idea of selection depends on the fitness values. Parents and offsprings are ranked according to their chance of competing for survival and this creates their fitness values in population. Using roulette wheel selection, we give a small chance to

weakest individual to be selected therefore prevent the risk of premature convergence. The flow chart of the GAVNS algorithm is given in Figure 4.15

Termination Criteria

Like SAVNS and TSVNS, we use two termination criteria at same time as maximum number of iterations and the maximum number of allowable non-improved solution.

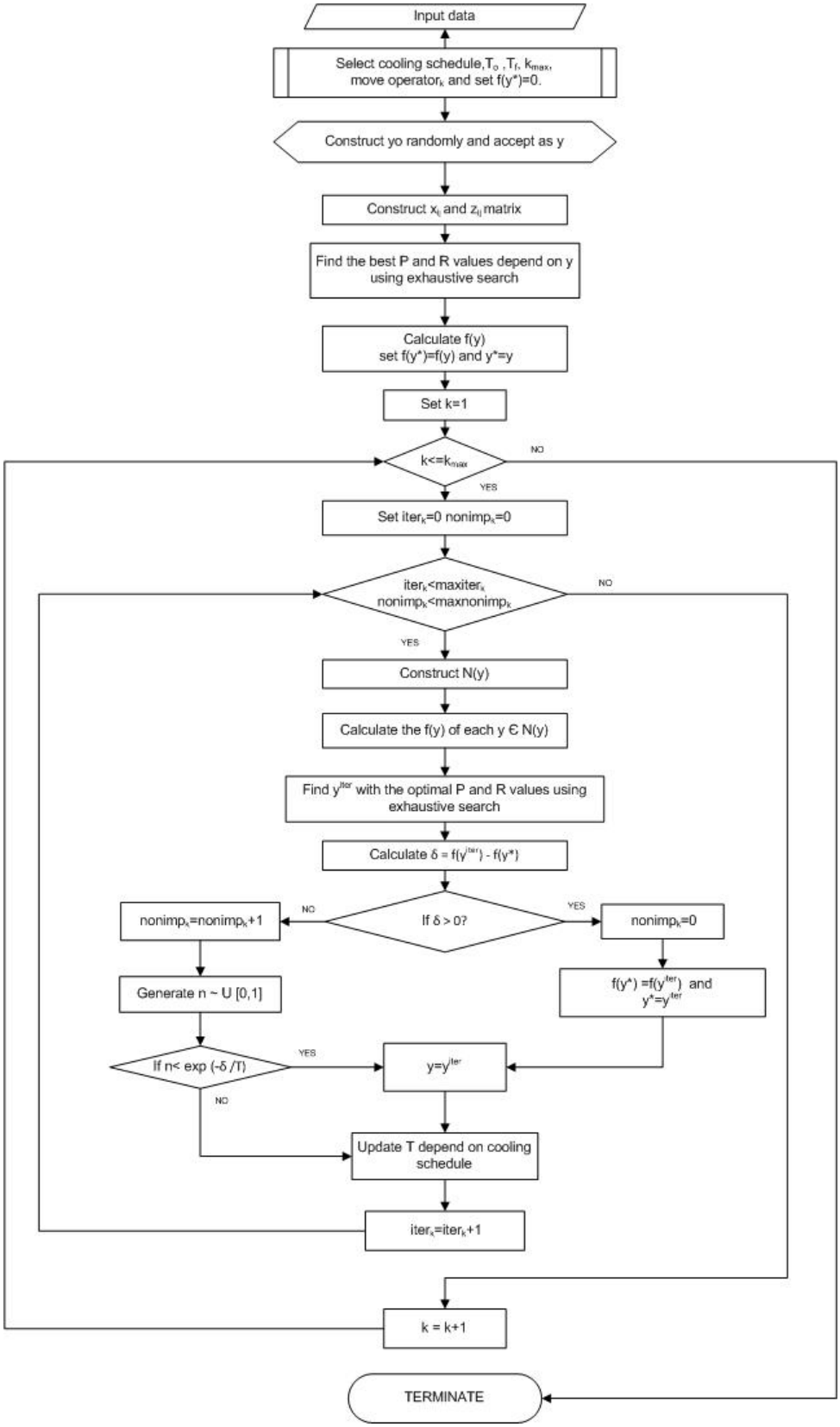


Figure 4.8: Flowchart of the SAVNS Algorithm

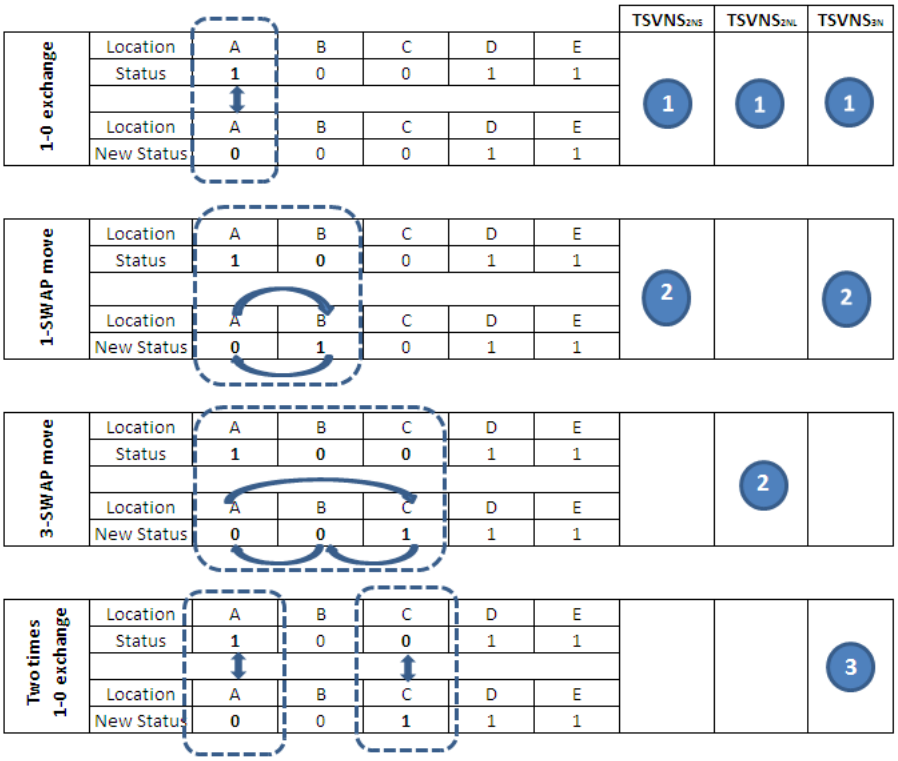


Figure 4.9: TSVNS neighborhood move operators

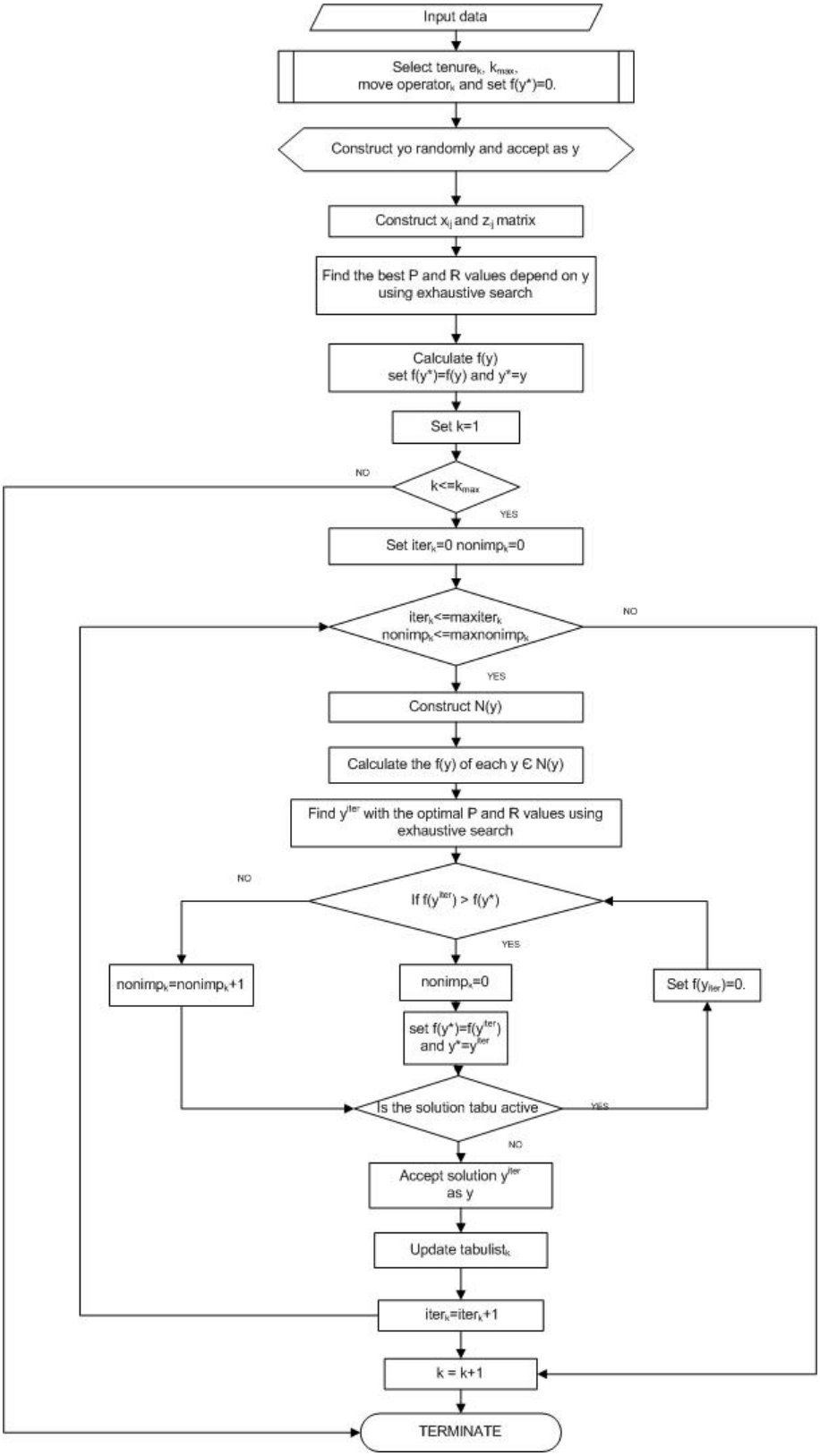


Figure 4.10: Flowchart of the TSVNS Algorithm

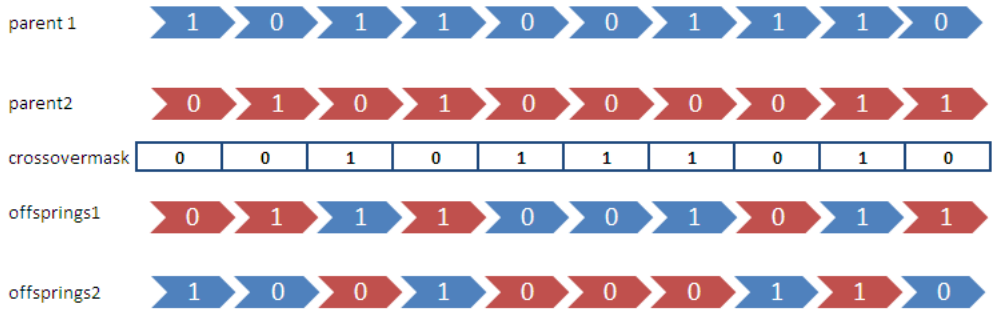


Figure 4.11: Crossover operation using crossover mask

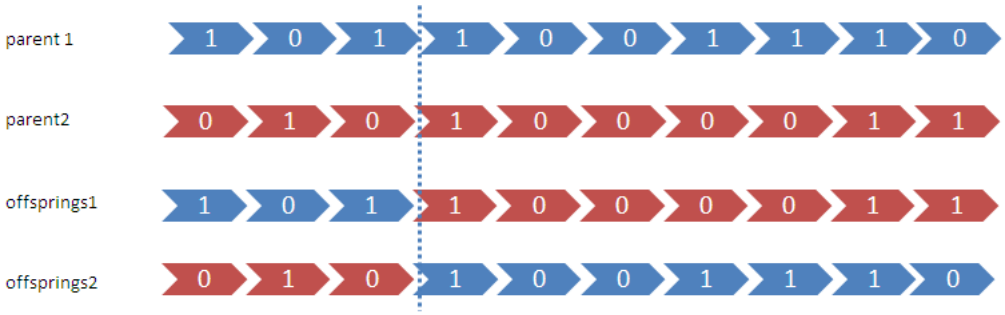


Figure 4.12: 1-point crossing over operation

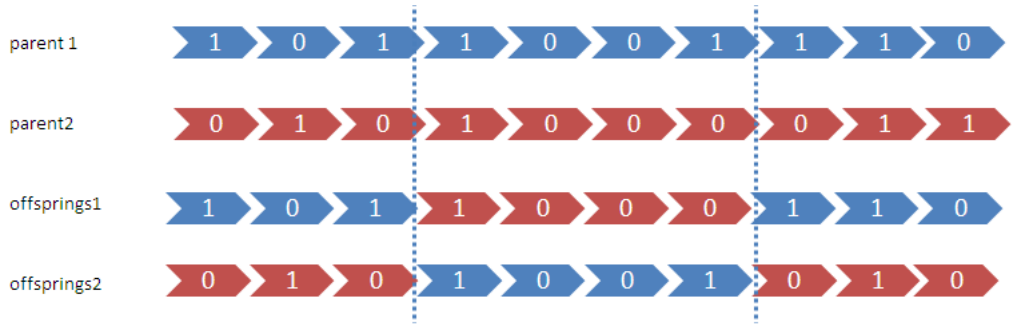


Figure 4.13: 2-point crossing over operation

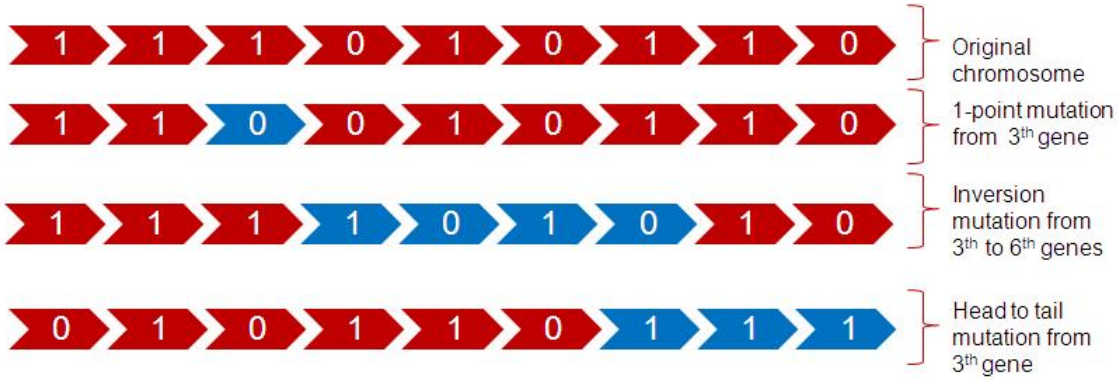


Figure 4.14: Mutation operators

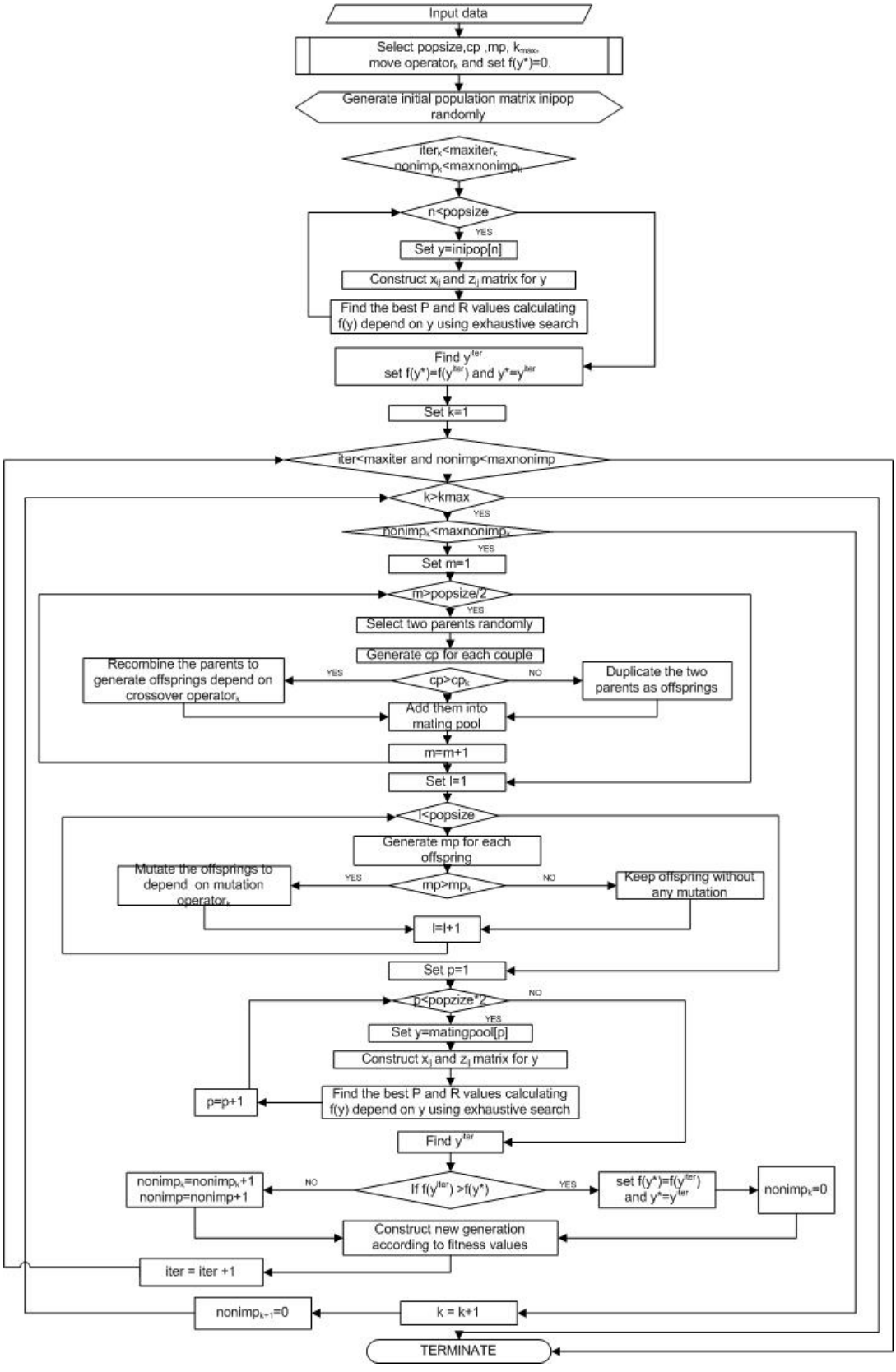


Figure 4.15: Flowchart of the GAVNS Algorithm

Chapter 5

COMPUTATIONAL STUDIES

In this chapter, computational experiments are examined to analyze the performance of proposed hybrid heuristics in terms of the solution quality and computation time. In Section (5.1) and (5.2.1), we describe how the test instances are generated and parameters are set. Then, in Section(5.3), results of the computational experiments are given and finally, in Section (5.4), results are discussed in detail.

5.1 Data Generation

Because our model has not been studied before, we generate new test instances in varying problem sizes and parameter values. To generate a test instance, we need the parameters mentioned in (3.2)as: demand and return rate of each customer location, respectively D_j and B_j, fixed ordering cost and fixed operation cost of each candidate CDC, A_i and F_i, transportation cost between potential CDC sites and customers, t_{ij}, unit holding cost of product and return $(hp_i)$and (hr_i), the multiplicative price and return constant k and b. In the test bed, there are 32 test instances of varying sizes as ixj=(10x10),(50x50),(100x100) and (200x200) where i is the number of candidate CDC sites and j is the number of customer zones. In all test instances, the customer zones are used as candidate location for potential CDCs. In original test data, the demand of each customer is generated using discrete uniform distribution in the interval [0,300]. Return amounts of each customer is correlated with demand of the customer with formula B=D*U [0,1]. Fixed ordering and operation costs for each candidate CDCs are generated using discrete uniform distribution between [500,1000] and [1000,10000] respectively. Distance between CDC and customer locations are generated using a continuous uniform distribution between [0,3]. (hp_i) and (hr_i), are set as 0.5 and 0.25, respectively, and assumed as constant for all candidate CDC locations.

Table 5.1: Parameter setting for original test data

Parameter	**Abbreviation**	**Value**
Demand rate	D_j	U [0;300]
Return rate	B_j	D*U [0;1]
Fixed Ordering Cost	A_i	U [500;1000]
Holding Cost of product at location i	hp_i	0.5
Holding Cost of return at location i	hr_i	0.25
Fixed Operating Cost	F_i	U [1000;10000]
Distance	t_{ij}	U [0;3]

In order to understand the effect of the parameters used in data generation, 7 different instances are created for all problem sizes. In this procedure, while creating an instance, we change only one parameter value and use same data for all remaining parameters as in theoriginal setting. In Table 5.2, the varying parameters and their generation rule are given.

Table 5.2: Parameter setting for different test instances

Instance	**Changed parameter**	**Abbreviation**	**Value**
a	Demand rate	D_j	U [0;600]
b	Return rate	B_j	D_a*U [0;1]
c	Fixed Ordering Cost	A_i	U [1000;2000]
d	Holding Cost of product at location i	hp_i	1
e	Holding Cost of return at location i	hr_i	0.25
f	Fixed Operating Cost	F_i	U [2000;20000]
g	Distance	t_{ij}	U [0;6]

The other algorithm specific parameters are explained in following sections.

5.2 Parameter Settings for the Algorithms

One of the main difficulties of applying an heuristic algorithm is setting the parameter values correctly. The parameter setting is critical, because it controls the balance between diversification and intensification in the search area. In this section, in order to observe the best performance of three hybrid metaheuristics, specific parameters and their final values are given with the methodology used through the parameter tuning. We select a special test instance for each problem size, named it as "original", and use it through the tuning.

Moreover, through experiments, discrete values of parameters are used when searching the best value of each parameter.

5.2.1 Parameter Settings for SAVNS

Basically, the parameters of the Simulated Annealing algorithm is initial and final temperature values, cooling schedule, maximum iteration and maximum non-improved iteration numbers. Throughout the tuning experiments, the parameters are initially set as follows:

Table 5.3: Initial value of parameters in SAVNS

T_o	100
T_f	0.01
cooling schedule	linear cooling schedule in Eqn4.8
maximum iteration number (N)	100
maximum number of consecutively nonimproved solution	25

To allow almost free exchange in neighboring solutions and to heat system rapidly, T_o value should be selected high enough. To decide the best, we examined the effects of four different temperature values for each population size. We tested the following temperatures as initial temperature shown in Table 5.4.

Table 5.4: Candidate values used as T_o for SA

Problem Size	Candidate Initial Temperature Values
10x10	100, 500, 1000 ,5000
50x50	100, 500, 1000 ,5000
100x100	100,500,1000,5000
200x200	100,500,1000,5000

The effects of different T_o's according to problem size can be seen in Table 5.5. Each value presented in Table 5.5 are averaged over 30 experiments with the selected case as T_f=T_o/4,T_o/2,T_o/10 and 0,01.

By observing the effects of initial temperatures on objective value, it can be easily seen that when T_o increases, the performance of the SAVNS increases. Hence, we set T_o as 5000

Table 5.5: Preliminary test results for T_o and T_f setting

		$T_f = T_o/2$		$T_f = T_o/4$		$T_f = T_o/10$		$T_f = 0.01$	
Problem Size	T_o	Result	Runtime	Result	Runtime	Result	Runtime	Result	Runtime
10x10	100	23451.80	0.90	23519.41	0.58	23451.80	0.59	23519.41	0.88
	500	23519.41	0.67	23519.41	1.10	23519.41	1.04	23519.41	0.63
	1000	23519.41	0.75	23519.41	1.16	23519.41	0.88	23519.41	0.77
	5000	23519.41	0.60	23519.41	0.61	23519.41	0.74	23519.41	0.69
50x50	100	181945.56	5.86	181033.94	6.12	182261.32	4.63	180738.95	4.11
	500	181608.23	3.26	181268.96	4.25	181119.88	4.24	181434.48	4.94
	1000	181429.45	5.60	182101.96	5.76	181127.65	5.07	182560.95	3.82
	5000	182703.67	3.54	182180.08	5.41	181616.70	4.33	182703.67	3.54
100x100	100	421863.48	18.17	422104.33	22.16	422110.47	22.19	420410.39	512.91
	500	421537.84	23.72	421849.36	16.51	421995.38	18.12	422130.04	555.93
	1000	422071.14	15.67	422090.46	15.88	421658.85	25.09	420467.98	588.51
	5000	422071.14	15.67	422090.46	15.88	421658.85	25.09	420467.98	588.51
200x200	100	1726776.58	85.58	1726539.84	88.04	1725638.32	93.99	1727086.04	88.12
	500	1726845.22	76.84	1726689.53	80.21	1727586.91	72.16	1726899.19	82.50
	1000	1726773.02	86.70	1726671.46	71.09	1726676.82	80.37	1727542.45	95.67
	5000	1727866.63	88.89	1727651.96	99.03	1726414.23	87.46	1728088.76	86.99

for all problem sizes.

On the other hand, the decrease in T_f does not effect significantly computational efficiency or CPU time. As seen from Table 5.5 there is not purely a dominant T_f value, therefore, when the number of iterations increases, to prevent the acceptance of non-improving moves, we suggest to set T_f as 0.01.

Termination Criterion: As termination criteria, we use two bounds, the total number of iterations performed and maximum number of consecutive iterations while the best solution (incumbent) does not improve. To select the best values, both parameters are analyzed based on the problem size. In Table 5.6, values used for tuning can be seen. Moreover, the values presented in Table 5.6 are the average run times and average objective function values of solutions obtained by the SAVNS algorithm over 30 experiments.

Table 5.6: Preliminary test results for termination criteria setting

	$Neighborhood_1$		$Neighborhood_2$		$Neighborhood_3$			
Problem Size	*maxiter*	max nonimp	*maxiter*	max nonimp	*maxiter*	max nonimp	Runtime	Result
10x10	25	10	25	10	25	10	23451.801	0.38
	50	10	50	10	50	10	23451.80	0.39
	50	20	50	20	50	25	23519.41	0.65
	100	10	100	10	100	10	23384.19	0.41
	100	20	100	20	100	25	23519.41	0.73
	100	50	100	50	100	50	23519.41	0.91
50x50	25	10	25	10	25	10	182203.36	3.54
	50	10	50	10	50	10	180237.95	4.50
	50	20	50	20	50	25	181684.58	8.32
	100	10	100	10	100	10	180354.92	4.48
	100	20	100	20	100	25	181477.15	8.30
	100	50	100	50	100	50	181313.20	16.51
100x100	25	10	25	10	25	10	422250.15	13.66
	50	10	50	10	50	10	420063.31	13.84
	50	20	50	20	50	20	422549.40	15.86
	100	10	100	10	100	10	422300.20	13.77
	100	20	100	20	100	20	422712.02	23.27
	100	50	100	50	100	50	422115.85	44.64
	200	10	100	10	100	10	422661.21	12.15
	200	20	100	20	100	20	421937.55	27.50
	200	50	100	50	100	50	421519.84	90.50
	200	100	200	100	200	100	422619.71	109.12
200x200	25	10	25	10	25	10	1727392.69	50.34
	50	10	50	10	50	10	1727382.50	54.67
	50	20	50	20	50	20	1727413.55	100.25
	100	10	100	10	100	10	1726626.79	58.45
	100	20	100	20	100	20	1728567.38	86.95
	100	50	100	50	100	50	1727742.79	214.53
	200	10	100	10	100	10	1727293.97	113.51
	200	20	100	20	100	25	1728001.04	123.20
	200	50	100	50	100	50	1726202.81	240.74
	200	100	200	100	200	100	1725901.88	575.13

From Table5.6 it can be easily observed that the increase in maximum iteration number results in an increase on CPU time. However, increase in maximum iteration number does not always provide better objective function values. When we focus on the results, the most efficient and effective results are gained with max iteration number as 100 and max nonimproved iteration number is 20.

5.2.2 Parameter Settings for TSVNS

As mentioned before, the parameters are initially set as follows in the experiments:

Table 5.7: Initial values of parameters in TSVNS

tenure	7
maximum iteration number (N)	100
maximum number of consecutively nonimproved solution	20

Determination of the tabu tenure is one of the most critical decision in TS because it controls the memory of the search. While larger tabu tenure explores the search and avoid to revisit high number of solutions, smaller ones cause algorithm to cycle. Therefore to decide the best value as tabu tenure, different tabu sizes are examined for each problem size as stated in Table 5.8

Table 5.8: Candidate Tabu Tenure values

	Tenure size	
problem size	$Neighborhood_1$	$Neighborhood_2$
10x10	5, 10, 20	5, 10, 20
50x50	5, 10, 20	5, 10, 20
100x100	7, 15, 30, 50	7, 15, 30, 50
200x200	7, 15, 30, 50	7, 15, 30, 50

The effect of the different tabu tenures on solution quality and run time can be seen in Table 5.9. The run time and objective values seen in Table 5.9are the output of the TSVNS algorithm, averaged over 30 experiments for each problem size.

Table 5.9: Preliminary test results for Tabu Tenure Setting

Problem Size	$tenure_1$	$tenure_2$	Runtime	Result
10x10	3	3	23519.41	0.04
	3	5	23519.41	0.10
	3	7	23519.41	0.07
	5	3	23519.41	0.20
	5	5	23519.41	0.08
	7	3	23519.41	0.24
	7	5	23519.41	0.32
	7	7	23519.41	0.27
50x50	3	5	183031.57	3.06
	3	7	183236.58	3.21
	7	3	182683.49	4.51
	7	5	182139.10	6.34
	7	7	181140.46	5.25
	7	15	181780.07	6.40
	7	30	181229.68	7.29
	15	7	182046.32	6.37
	15	15	181728.42	7.21
	15	30	180685.69	5.29
100x100	7	7	423692.31	102.83
	7	15	423692.31	92.83
	7	30	423714.74	82.52
	7	50	423562.78	81.11
	15	7	423638.98	94.39
	15	15	423724.71	101.22
	15	30	423871.39	100.57
	15	50	423443.09	124.00
	30	7	423726.09	100.48
	30	15	423545.26	124.65
	30	30	423797.55	114.56
	30	50	423473.87	124.60
200x200	7	7	1730412.23	227.41
	7	15	1729907.94	706.12
	7	30	1729858.70	783.91
	7	50	1729858.70	798.91
	15	7	1730148.09	789.55
	15	15	1730761.70	898.74
	15	30	1731127.82	801.89
	15	50	1730048.78	872.62
	30	7	1729434.79	772.24
	30	15	1730204.37	835.42
	30	30	1731112.88	960.53
	30	50	1730431.94	1021.72

According to the observed results in Table 5.9,the best performances of tenure size are achieved in terms of solution quality and CPU times for 50x50 problem size as 3 and 7 and 100x100 and 200x200 problem size 15, 30. For the problem size 10x10, neither smaller or larger tenure effects the solution quality so we set the tenure as 3 and 7 for latter runs. We observed that increasing tabu tenure size after a value does not produce better results and increase the CPU time redundantly.

Two termination criteria are also used in TS like SA as the total number of iterations performed and maximum number of iterations during which the incumbent does not improve. Both values are examined according to the problem size as follows:

Table 5.10: Candidate values for Termination criteria

	Neighborhood1		Neighborhood2	
problem size	max iteration	max nonimp	max iteration	max nonimp
10x10	10,20,50,100	5,10,25	10,20,50,100	5,10,25
50x50	10,20,50,100	5,10,25	10,20,50,100	5,10,25
100x100	20,50,100	10,20,50	10,20,50,100	10,20,50
200x200	20,50,100	10,20,50	10,20,50,100	10,20,50

The effect of different termination criteria on solution quality and run time can be seen in Table 5.11. Through the experiments, tabu tenures are selected as 7, 10, 20, 25 for 10x10, 50x50, 100x100 and 200x200 respectively and the maximum iteration number set in $Neighborhood_1$ and $Neighborhood_2$ are 50 and 100 respectively.

By observing the effects of these different values, for smallest problem size(10x10) solution quality is same for all termination value so we choose the value that results in the smallest CPU time. Therefore, maximum iteration number is set as 10 for problem size 10x10 and we eliminate the second termination criteria. For 50x50, 100x100 and 200x200 problem sizes, maximum iteration and maximum nonimproved solution numbers are set as 50,5 ;50,10 and 50,10 respectively. Final parameter values of TSVNS are summarized as shown in Table 5.13.

Table 5.11: Preliminary test results for Termination Criteria

problem size	max iteration	max nonimp	Runtime	result
10x10	10	5	23519.41	0.138
	10	10	23519.41	0.05
	20	5	23519.41	0.088
	20	10	23519.41	0.139
	50	5	23519.41	0.067
	50	10	23519.41	0.08
	50	25	23519.41	0.092
	100	10	23519.41	0.128
	100	25	21861.35	0.155
	100	50	23519.41	0.198
50x50	10	5	183107.26	2.12
	10	10	182168.25	1.727
	20	5	183200.401	2.32
	20	10	182652.64	2.68
	50	5	183237.64	1.97
	50	10	183076.97	3.87
	50	25	182020.54	4.47
	50	50	181356.45	5.18
	100	5	182124.58	1.65
	100	10	181107.81	2.21
	100	25	181764.14	4.491
	100	50	181924.67	6.849
100x100	20	10	423418.83	32.451
	20	20	422661.94	35.784
	50	10	423684.24	56.509
	50	20	422878.72	60.981
	50	50	422399.50	66.714
	100	10	422177.53	59.88
	100	20	422969.19	67.38
	100	50	422924.94	75.51
200x200	20	10	1729680.55	367.46
	20	20	1727753.97	402.93
	50	10	1730287.33	515.89
	50	20	1729242.14	554.44
	50	50	1729237.84	603.91
	100	10	1727694.44	495.01
	100	20	1728014.44	510.76
	100	50	1727271.85	604.59

Table 5.12: Final values used as termination criteria for TAVNS

	$Neighborhood_1$		$Neighborhood_2$	
problem size	max iteration	max nonimp	max iteration	max nonimp
10x10	10	-	10	-
50x50	50	5	50	5
100x100	50	10	50	10
200x200	50	10	50	10

5.2.3 Parameter Settings for GAVNS

For parameter tuning, the parameters are initially set as follows:

Table 5.13: Initial value of parameters in GAVNS

population size	20
p_c	0.9
p_m	0.1
max iteration	20
max nonimp	10

For all problem sizes, 3 different population sizes are applied with 20, 40 and 80 chromosomes. In other words, like many other parameters, we do not specify the population size according to problem size.

Performance of the Genetic Algorithm also depends on the crossover probability (p_c) and mutation probability (p_m), therefore these parameters have to be chosen carefully. To decide on the best p_c, 0,50, 0.65, 0.8, 0.95 and 1.0 are tried for all problem sizes. The effect of different p_c and p_m on solution quality and run time can be seen in the Table 5.14 and Table 5.15.

According to the observed results in Table5.14,the best p_c performances are achieved in terms of solution quality and CPU times with 0.95 for all problem sizes.

To decide on the best p_m, values of 0.001, 0.01, 0.1, 0.2 and 0.5 are analyzed for all neighborhoods. The effect of different p_m on solution quality and run time can be seen in Table 5.15.

According to the observed results in Table 5.15, the best p_m performance are achieved in terms of solution quality and CPU times with 0.2 for all problem sizes.

Termination Criterion Setting:

Like SA and TA, two termination criteria are also used in GA as in the total number of

Table 5.14: Preliminary test results for p_c

Problem Size	p_c	Result	Runtime
10x10	0.5	23316.58	2.19
	0.65	23451.80	2.83
	0.85	23451.80	2.04
	0.95	23451.80	2.98
	1.00	23326.85	2.34
50x50	0.5	1183130.37	42.94
	0.65	183160.87	45.21
	0.85	183168.59	39.83
	0.95	183205.05	46.61
	1.00	183109.68	47.52
100x100	0.5	423588.12	307.73
	0.65	423632.07	322.37
	0.85	423672.54	305.74
	0.95	423787.81	347.08
	1.00	423597.71	309.72
200x200	0.05	1730597.90	1373.60
	0.65	1730487.71	1155.15
	0.85	1730844.51	1296.53
	0.95	1731026.99	1068.11
	1.00	1730277.76	1177.57

Table 5.15: Preliminary test results for p_m

Problem Size	p_m	Runtime	Result
10x10	0.001	22556.66	2.34
	0.01	22710.22	2.88
	0.1	23316.58	1.72
	0.2	23519.41	1.84
	0.5	23519.41	1.63
50x50	0.001	183191.74	42.98
	0.01	183087.22	49.46
	0.1	183095.46	42.11
	0.2	183087.22	49.46
	0.5	183191.74	42.98
100x100	0.001	423588.75	325.50
	0.01	423530.85	370.24
	0.1	423519.45	348.05
	0.2	423631.21	355.80
	0.5	423643.64	376.52
200x200	0.001	1730230.59	1091.88
	0.01	1730000.99	1086.94
	0.1	1730449.62	1061.66
	0.2	1730581.21	1073.66
	0.5	1730255.47	1056.72

iterations performed and maximum number of iterations during which the incumbent does not improve. For each problem size, effects of different values can be seen in Table 5.16.

Table 5.16: Preliminary test results for termination criteria setting for GA

	$Neighborhood_1$		$Neighborhood_2$		$Neighborhood_3$			
Problem Size	*maxiter*	max nonimp	*maxiter*	max nonimp	*maxiter*	max nonimp	Runtime	Result
10x10	10	5	10	5	10	5	23384.12	1.83
	10	10	10	10	10	10	23220.30	2.42
	20	10	20	10	20	10	23451.81	3.02
	50	10	50	10	50	10	23451.81	2.81
	50	20	50	20	50	20	23519.41	4.67
	50	50	50	50	50	50	23519.41	9.09
	100	10	100	10	100	10	23519.41	2.84
	100	20	100	20	100	20	23519.41	4.961
	100	50	100	50	100	50	23519.41	11.54
50x50	10	5	10	5	10	5	182778.02	21.03
	10	10	10	10	10	10	183087.18	28.91
	20	10	20	10	20	10	182971.89	20.46
	50	10	50	10	50	10	182871.80	33.76
	50	20	50	20	50	20	183094.88	52.99
	50	50	50	50	50	50	183100.91	68.38
	100	10	100	10	100	10	183109.67	52.33
	100	20	100	20	100	20	183132.05	61.56
	100	50	100	50	100	50	183124.40	113.45
100x100	10	5	10	5	10	5	423415.12	233.44
	10	10	10	10	10	10	423209.2	240.71
	20	10	20	10	20	10	423295.83	282.14
	50	10	50	10	50	10	423294.68	318.88
	50	20	50	20	50	20	423350.79	340.71
	50	50	50	50	50	50	423379.76	403.4
	100	10	100	10	100	10	423370.77	342.357
	100	20	100	20	100	20	423632.26	359.83
	100	50	100	50	100	50	423492.18	639.57
200x200	10	5	10	5	10	5	1729215.80	678.19
	10	10	10	10	10	10	1729508.952	691.61
	20	10	20	10	20	10	1729198.21	697.64
	50	10	50	10	50	10	1729380.68	733.82
	50	20	50	20	50	20	1729369.41	795.69
	50	50	50	50	50	50	1730398.82	883.99
	100	10	100	10	100	10	1730121.60	985.38
	100	20	100	20	100	20	1730493.43	1014.42
	100	50	100	50	100	50	1730328.32	2039.47

According to the observed results in Table 5.16, the best performance are achieved in terms of solution quality and CPU times with maximum iteration number as 20 and maximum non-improved solution number as 10 for 10x10 problem sizes. For other sizes, maximum iteration number is set as 100 and maximum non-improved solution number is set as 20.

5.3 Results of Computational Experiments

In this section, we explain the results of our numerical experiments with hybrid metaheuristics. We compare the results of proposed algorithms with upper bounds in terms of solution quality and tabulate run times.

5.3.1 Computational Platform

All of the runs throughout the computational experiments are performed on a workstation with an Intel(R) Core(TM)2 Duo processor, 2.53 GHz speed, and 2GB of RAM. All of the metaheuristic algorithms are coded in C. Furthermore, the relaxed model used to find upper bound is solved in ILOG CPLEX 11.2.

For smallest problem size, 10x10, using MINLP with CPLEX, optimal solutions are found as follows:

Table 5.17: For 10x10 problem size, the optimal results found by CPLEX

Size	Test instance	Result	Runtime
10x10	original	23519.40	308.2
	a	87284.17	323.5
	b	27641.52	317.9
	c	23256.74	347.1
	d	23259.04	326.0
	e	23500.86	340.2
	f	16995.80	321.7
	g	18380.83	330.8

5.3.2 Upper Bounds

Due to the failure of exact computational methods when solving large sized instances, we use an upper bound UB, to compare the quality of the solutions obtained by proposed heuristic algorithms mentioned in Section 4.3.2, Section 4.3.3 and Section 4.3.4. The upper bound is found by the methodology explained in Section 4.2.
In the specified time limit (3600sec.), the upper bounds found by the CPLEX are given in Table 5.18.

Table 5.18: Upper bound of the test instances found by GAMS

	Probleminstances							
problem size	original	a	b	c	d	e	f	g
10x10	24683.85	90963.20	28591.24	23894.67	23981.70	24002.84	17369.71	18729.88
50x50	188510.69	520122.24	216820.17	189383.67	185394.06	189413.56	177989.11	189329.55
100x100	433186.42	981370.48	473590.26	427155.79	436440.43	438301.26	420677.67	302250.39
200x200	1830315.01	2263992.50	2040801.80	1886922.29	1882225.57	1918459.91	1834869.32	1852555.39

5.3.3 Performance Indicator Parameters

We use two different parameters to measure the performances of the proposed heuristic algorithms as deviation from upper bound and computational time.

Deviation from Upper Bound

For each problem instance, the percentage deviation of objective function values from upper bounds is used as one of the performance indicator parameter. The average magnitude of deviations are calculated as follows:

$$\%averageUBdeviation = \frac{\sum_{i=0}^{n} \frac{UB - Objectivevalue}{UB}}{n} \tag{5.1}$$

where n represents the number of replication done for each problem size.

Computation Times

In literature, another performance indicator parameter is the average computation time, therefore for making comparison and discussion, average computation times are also considered.

In the following, we conduct the results and performance indicator parameters of all test problems in Table 5.19, Table 5.20 and Table 5.21.

Table 5.19: Performance of Simulated Annealing Algorithm hybridized with VNS

		Hyperbolic cooling schedule		Exponential cooling schedule		Linear cooling schedule	
Size	test instance	Result	CPU	Results	CPU	Results	CPU
10x10	original	23519.41	0.34	23519.41	0.64	23519.41	0.59
	a	86621.88	0.24	87284.18	0.46	87284.18	0.43
	b	27640.54	0.29	27640.54	0.39	27640.54	0.53
	c	23256.74	0.20	23256.74	0.49	23256.74	0.53
	d	23259.04	0.22	23259.04	0.43	23259.04	0.48
	e	23500.87	0.24	23500.87	0.43	23500.87	0.46
	f	16995.81	0.17	16995.81	0.21	16995.81	0.21
	g	17928.60	0.21	18154.72	0.47	18380.83	0.43
50x50	original	182024.47	5.56	181316.65	6.92	181564.44	7.44
	a	497375.77	3.27	495912.92	5.24	497076.42	4.39
	b	208439.56	3.18	208050.27	5.02	207614.36	4.92
	c	180698.97	3.49	179875.24	4.61	180076.91	4.70
	d	179887.42	4.10	180058.42	4.62	180202.38	5.01
	e	181947.40	3.25	181377.12	4.82	182384.99	5.22
	f	172966.06	3.19	171512.85	4.57	173557.39	4.99
	g	182344.67	2.98	182081.45	4.34	182821.98	5.09
100x100	original	421840.69	22.38	421405.46	25.63	421992.81	27.70
	a	953743.33	15.37	953338.53	18.31	952856.09	19.28
	b	469949.55	14.44	470264.12	17.55	470517.52	17.72
	c	418879.74	14.66	418324.58	18.50	419295.32	19.22
	d	419272.58	15.04	419576.36	17.64	419839.81	16.79
	e	422024.82	15.20	421236.72	15.08	422084.71	18.63
	f	412388.61	13.53	413906.97	16.82	413045.22	17.60
	g	286875.48	14.88	284496.90	17.67	288499.36	18.38
200x200	original	1727155.79	116.30	1727699.09	110.47	1728000.15	137.04
	a	11946956.80	84.99	1952874.87	95.38	1963438.74	80.51
	b	1881952.68	80.12	1881045.75	136.35	1880216.02	69.02
	c	1718506.50	86.87	1719233.80	82.45	1719617.89	78.75
	d	1719607.04	76.74	1718489.00	79.22	1719617.89	78.75
	e	1727056.24	73.65	1727864.20	72.54	1727129.74	93.04
	f	1671862.04	79.68	1671479.53	92.26	1671104.30	77.72
	g	1689997.78	78.88	1689838.52	97.20	1691546.34	89.59

Table 5.20: Performance of Tabu Search Algorithm hybridized with VNS

		$2Neighborhoods_{small}$		$2Neighborhoods_{large}$		$3Neighborhoods$	
Size	test instance	Result	CPU	Result	CPU	Result	CPU
10x10	original	23519.41	0.10	23519.41	2.41	23519.41	0.27
	a	87284.18	0.12	87284.18	2.24	87284.18	0.18
	b	27641.54	0.09	27641.54	4.23	27641.54	0.22
	c	23256.74	0.15	23256.74	3.10	23256.74	0.29
	d	23259.04	0.08	23259.04	2.98	23259.04	0.19
	e	23500.87	0.13	23500.87	3.15	23500.87	0.26
	f	16995.81	0.14	16995.81	4.29	16995.81	0.21
	g	18380.83	0.07	18380.83	3.65	18380.83	0.28
50x50	original	183128.57	2.98	183074.23	34.29	183258.38	25.96
	a	500073.82	3.65	498613.94	121.83	499002.91	25.62
	b	210311.72	3.19	210504.84	102.24	210811.58	25.10
	c	181548.89	3.12	181358.62	136.85	181363.57	25.36
	d	180269.03	3.21	180952.95	148.93	181930.43	28.96
	e	182876.97	3.49	181784.82	139.12	183097.50	27.84
	f	172873.21	2.62	173548.01	142.77	173540.31	26.51
	g	182873.27	3.10	182548.92	124.61	182971.69	28.49
100x100	original	423385.39	25.03	423608.67	592.96	423362.33	184.88
	a	955510.42	105.15	955559.54	797.58	955531.06	208.07
	b	472396.00	76.42	472551.69	697.27	472400.18	197.17
	c	420669.29	58.71	420825.35	796.56	420730.00	167.95
	d	421232.73	57.84	421152.40	748.66	421187.75	172.75
	e	423337.96	58.89	423366.44	833.80	433351.93	192.30
	f	414643.36	48.54	414734.12	735.83	414702.61	167.02
	g	294063.28	63.20	295685.60	740.36	295162.91	205.62
200x200	original	1518778.06	285.88	1730012.67	9904.75	1730513.49	430.07
	a	2028492.15	650.62	2024821.54	9842.25	2019541.93	1363.75
	b	1884821.93	557.35	1884529.85	9367.02	1886026.55	1343.69
	c	1821938.84	587.56	1884597.06	9510.31	1885708.48	1338.01
	d	1722850.84	592.61	1724741.82	9899.94	1722993.05	1374.32
	e	1729021.07	621.73	1729316.26	9649.27	1730930.85	1342.09
	f	1675210.93	618.96	1674563.03	9713.15	1675213.42	1438.77
	g	1690542.42	635.84	1692810.72	9695.42	1694023.12	1407.46

Table 5.21: Performance of Genetic Algorithm hybridized with VNS

		Population size=20		Population Size=40		Population Size=60	
Size	test instance	Result	CPU	Result	CPU	Result	CPU
10x10	original	23519.41	3.10	23519.41	5.23	23519.41	8.03
	a	87284.18	4.48	87284.18	4.98	87284.18	7.43
	b	27640.54	3.86	27641.54	5.38	27641.54	6.84
	c	23256.74	2.31	23256.74	4.77	23256.74	6.91
	d	23259.04	3.84	23259.04	4.10	23259.04	6.89
	e	23500.87	2.86	23500.87	4.47	23500.87	6.67
	f	16995.81	2.44	16995.81	5.67	16995.81	7.52
	g	18380.83	3.29	18380.83	5.28	18380.83	8.12
50x50	original	183153.795	59.66	183273.82	121.79	183273.82	623.21
	a	497871.65	50.58	498519.14	188.98	499628.63	623.42
	b	209153.38	55.53	208191.38	103.05	209191.69	638.87
	c	180526.14	50.08	180396.72	111.23	181399.10	592.00
	d	180180.33	54.40	181022.67	100.75	181618.11	559.64
	e	182633.72	51.85	182710.01	107.01	181858.78	626.00
	f	173417.26	48.21	172966.06	102.52	173176.26	571.91
	g	182118.53	48.05	182821.98	106.38	182081.45	521.77
100x100	original	422665.87	358.92	423133.98	568.76	422689.97	1236.54
	a	953680.92	342.50	954620.56	541.01	952597.18	845.33
	b	469481.59	367.51	471120.52	554.08	470472.72	1383.61
	c	418524.99	352.79	420036.70	541.58	419060.43	836.34
	d	419165.01	379.86	420898.87	595.91	419551.96	832.63
	e	421806.16	356.04	421638.65	522.94	422336.97	1371.45
	f	412007.22	372.09	414168.80	551.80	413033.75	1123.58
	g	286701.70	340.79	290459.14	740.23	285608.00	1216.02
200x200	original	1657432.93	1004.77	1719875.09	2006.86	1701238.74	8423.00
	a	2006045.83	990.29	2021259.32	1993.22	2017051.00	7692.39
	b	1884971.83	1012.76	1885364.15	2021.25	1885619.32	7900.52
	c	1723334.96	1011.87	1724014.24	2043.70	1723787.95	8259.35
	d	1722807.38	1014.10	1723607.44	2024.67	1722991.79	8331.82
	e	1729193.45	1028.01	1730155.43	2034.04	1729659.83	7764.84
	f	1674281.45	1041.58	1675073.91	2065.55	1674905.60	8723.98
	g	1691732.42	1454.65	1692480.12	2038.22	1690731.69	7692.66

Table 5.22: Deviation SAVNS from UB found by linearization

		Hyperbolic cooling schedule		Exponential cooling schedule		Linear cooling schedule	
Size	test instance	%deviation	CPU	%deviation	CPU	%deviation	CPU
10x10	original	4.951%	0.34	4.951%	0.64	4.951%	0.59
	a	4.942%	0.24	4.215%	0.46	4.215%	0.43
	b	3.442%	0.29	3.442%	0.39	3.442%	0.53
	c	2.743%	0.20	2.743%	0.49	2.743%	0.53
	d	3.107%	0.22	3.107%	0.43	3.107%	0.48
	e	2.136%	0.24	2.136%	0.43	2.136%	0.46
	f	2.200%	0.17	2.200%	0.21	2.200%	0.21
	g	4.313%	0.21	3.106%	0.47	1.899%	0.43
50x50	original	3.441%	5.56	3.816%	6.92	3.685%	7.44
	a	4.373%	3.27	4.655%	5.24	4.431%	4.39
	b	3.865%	3.18	4.045%	5.02	4.246%	4.92
	c	4.586%	3.49	5.021%	4.61	4.914%	4.70
	d	2.970%	4.10	2.878%	4.62	2.800%	5.01
	e	3.942%	3.25	4.243%	4.82	3.711%	5.22
	f	2.822%	3.19	3.639%	4.57	2.490%	4.99
	g	3.689%	2.98	3.828%	4.34	3.437%	5.09
100x100	original	2.619%	22.38	2.720%	25.63	2.584%	27.70
	a	2.815%	15.37	2.856%	18.31	2.906%	19.28
	b	0.769%	14.44	0.702%	17.55	0.649%	17.72
	c	1.937%	14.66	2.067%	18.50	1.840%	19.22
	d	3.934%	15.04	3.864%	17.64	3.804%	16.79
	e	3.714%	15.20	3.893%	15.08	3.700%	18.63
	f	1.970%	13.53	1.609%	16.82	1.814%	17.60
	g	5.087%	14.88	5.874%	17.67	4.550%	18.38
200x200	original	5.636%	116.30	5.606%	110.47	5.590%	137.04
	a	14.003%	84.99	13.742%	95.38	13.275%	80.51
	b	7.784%	80.12	7.828%	136.35	7.869%	69.02
	c	8.925%	86.87	8.887%	82.45	8.867%	78.75
	d	8.640%	76.74	8.699%	79.22	8.639%	78.75
	e	9.977%	73.65	9.935%	72.54	9.973%	93.04
	f	8.884%	79.68	8.905%	92.26	8.925%	77.72
	g	8.775%	78.88	8.783%	97.20	8.691%	89.59

Table 5.23: Deviation TSVNS from UB found by linearization

		$2Neighborhoods_{small}$		$2Neighborhoods_{large}$		$3Neighborhoods$	
Size	test instance	Result	CPU	Result	CPU	Result	CPU
10x10	original	4.951 %	0.10	4.951%	2.41	4.951%	0.27
	a	4.215%	0.12	4.215%	2.24	4.215%	0.18
	b	3.438%	0.09	3.438%	4.23	3.438%	0.22
	c	2.743%	0.15	2.743%	3.10	2.743%	0.29
	d	3.107%	0.08	3.107%	2.98	3.107%	0.19
	e	2.136%	0.13	2.136%	3.15	2.136%	0.26
	f	2.200%	0.14	2.200%	4.29	2.200%	0.21
	g	1.899%	0.07	1.899%	3.65	1.899%	0.28
50x50	original	2.842%	59.66	2.778%	121.79	2.778%	623.21
	a	3.855%	3.65	4.135%	121.83	4.060%	25.62
	b	3.002%	3.19	2.913%	102.24	2.771%	25.10
	c	4.137%	3.12	4.237%	136.85	4.235%	25.36
	d	2.764%	3.21	2.395%	148.93	1.868%	28.96
	e	3.451%	3.49	4.028%	139.12	3.335%	27.84
	f	2.874%	2.62	2.495%	142.77	2.499%	26.51
	g	3.410%	3.10	3.581%	124.61	3.358%	28.49
100x100	original	2.263%	25.03	2.211%	592.96	2.268%	184.88
	a	2.635%	105.15	2.630%	797.58	2.633%	208.07
	b	0.252%	76.42	0.219%	697.27	0.251%	197.17
	c	1.519%	58.71	1.482%	796.56	1.504%	167.95
	d	3.484%	57.84	3.503%	748.66	3.495%	172.75
	e	3.414%	58.89	3.407%	833.80	1.129%	192.30
	f	1.434%	48.54	1.413%	735.83	1.420%	167.02
	g	2.709%	63.20	2.172%	740.36	2.345%	205.62
200x200	original	5.453%	285.88	5.480%	9904.75	5.453%	430.07
	a	10.402%	650.62	10.564%	9842.25	10.797%	1363.75
	b	7.643%	557.35	7.657%	9367.02	7.584%	1343.69
	c	3.444%	587.56	0.123%	9510.31	0.064%	1338.01
	d	8.467%	592.61	8.367%	9899.94	8.460%	1374.324
	e	9.875%	621.73	9.859%	9649.27	9.775%	1342.09
	f	8.701%	618.96	8.737%	9713.15	8.701%	1438.77
	g	8.745%	635.84	8.623%	9695.42	8.557%	1407.46

Table 5.24: Deviation GAVNS from UB found by linearization

		Population size=20		Population Size=40		Population Size=60	
Size	test instance	Result	CPU	Result	CPU	Result	CPU
10x10	original	4.951 %	3.10	4.951%	5.23	4.951%	8.03
	a	4.215%	4.48	4.215%	4.98	4.215%	7.43
	b	3.442%	3.86	3.438%	5.38	3.438%	6.84
	c	2.743%	2.31	2.743%	4.77	2.743%	6.91
	d	3.107%	3.84	3.107%	4.10	3.107%	6.89
	e	2.136%	2.86	2.136%	4.47	2.136%	6.67
	f	2.200%	2.44	2.200%	5.67	2.200%	7.52
	g	1.899%	3.29	1.899%	5.28	1.899%	8.12
50x50	original	2.429%	358.92	2.321%	568.76	2.423%	1236.54
	a	2.822%	342.5	2.726%	541.01	2.932%	845.33
	b	0.868%	367.51	0.521%	554.08	0.658%	1383.61
	c	2.021%	352.79	1.667%	541.58	1.895%	836.34
	d	3.958%	379.86	3.561%	595.91	3.870%	832.63
	e	3.763%	356.04	3.802%	522.94	3.642%	1371.45
	f	2.061%	372.09	1.547%	551.8	1.817%	1123.58
	g	5.144%	340.79	3.901%	740.23	5.506%	1216.02
100x100	original	2.429%	358.92	2.321%	568.76	2.423%	1236.54
	a	2.822%	342.5	2.726%	541.01	2.932%	845.33
	b	0.868%	367.51	0.521%	554.08	0.658%	1383.61
	c	2.021%	352.79	1.667%	541.58	1.895%	836.34
	d	3.958%	379.86	3.561%	595.91	3.870%	832.63
	e	3.763%	356.04	3.802%	522.94	3.642%	1371.45
	f	2.061%	372.09	1.547%	551.8	1.817%	1123.58
	g	5.144%	340.79	3.901%	740.23	5.506%	1216.02
200x200	original	9.445%	1004.77	6.034%	2006.86	7.052%	8423
	a	11.393%	990.29	10.721%	1993.22	10.907%	7692.39
	b	7.636%	1012.76	7.616%	2021.25	7.604%	7900.52
	c	8.670%	1011.87	8.634%	2043.7	8.646%	8259.35
	d	8.470%	1014.1	8.427%	2024.67	8.460%	8331.82
	e	9.866%	1028.01	9.815%	2034.04	9.841%	7764.84
	f	8.752%	1041.58	8.709%	2065.55	8.718%	8723.98
	g	8.681%	1454.65	8.641%	2038.22	8.735%	7692.66

5.4 Analysis of the Results

According to the results presented in Table 5.22, Table 5.23 and Table 5.24 the most effective and efficient algorithm seems to be Tabu Search Algorithm hybridized with VNS for all problem sizes. The detailed analysis can be seen as follows.

5.4.1 Analysis and comparison of SAVNS, TSVNS and GAVNS

According to Tables 5.19 and 5.22, it is observed that SAVNS outperforms the other heuristic methods in terms of CPU time. On the other hand, from Table 5.22 we observe that when the problem size increases, SAVNS is not as efficient as GAVNS and TSVNS in terms of solution quality.

From Table 5.22, we observe that for n=10x10, SAVNS with the linear cooling schedule ($SAVNS_{LIN}$) finds the best solutions 80 times out of 80 runs, whereas the SAVNS with exponential cooling schedule ($SAVNS_{EXP}$)and the SAVNS with hyperbolic cooling schedule($SAVNS_{HYP}$) finds the best solutions less than 80.(78 and 67 respectively)

The differences in results can be explained as follows. Temperature reduces gradually in the linear cooling schedule, therefore, possibility of accepting a nonimproved solution decreases gradually. However, in exponential cooling schedule, acceptance rate of a nonimproved solution decreases more rapidly, because the trend of schedule goes rapidly to lower temperatures than linear schedule, since the probability function based on "Boltzmann" distribution restricts the approval of nonimproved solutions at lower temperatures. For this reason, diversification of search is prevented. Unlike the exponential cooling schedule, the hyperbolic schedule spends more time at higher temperatures. Therefore it has more chance to accept a nonimproved solution at the beginning of the search. This extends the search area and may direct to the far-optimal solutions in the rest of search.

Acoording to Table 5.22, the average runtime of SAVNS is less than a second whereas

the average run time of TSVNS and GAVNS are 1.20 seconds and 5.19 seconds. Even though, the average CPU time of $TSVNS_{2NS}$ and $TSVNS_{3N}$ are less than a second, the average CPU time of $TSVNS_{2NL}$ is 3.26 seconds and increases the value on average. In the same manner, when population size increases, CPU times increase for GAVNS. While the $GAVNS_{20}$ runs in 0.11 seconds on average, $GAVNS_{60}$ runs 7.30 seconds on average. According to average CPU times and results, $TSVNS_{2NS}$ and $TSVNS_{3N}$ outperforms the other metaheuristics for n=10x10.

When we compare the optimum results found by CPLEX with $TSVNS_{2NS}$ and $TSVNS_{3N}$, it is obvious that for n=10x10, TSVNS presents optimum results with much more smaller runtime.

When the problem size increases to n=50x50; on average, the deviation between the upper bounds and the average results of all instances found by SAVNS, increases to 3.81%. When we compare the average results, $SAVNS_{LIN}$ still outperforms $SAVNS_{EXP}$and $SAVNS_{HYP}$. While in $SAVNS_{LIN}$, the average deviation from the upper bound is 3.71%, $SAVNS_{EXP}$ and $SAVNS_{HYP}$ has the deviation 4.01%and %3.81. TSVNS gives 3.31% deviation on average of all problem instances. In TSVNS, $TSVNS_{3N}$ has 3.16% deviation on average and outperforms $TSVNS_{2NS}$ and $TSVNS_{3N}$. On the other hand, GAVNS gives 3.48% deviation on average. The increase of deviation for GAVNS comes from the $GAVNS_{20}$ and $GAVNS_{60}$ as 3.55 % and 3.45% on average. The $GAVNS_{40}$ runs with 3.44% deviation however it is still not enough to catch up the performance of TSVNS.

The average run time of SAVNS is 4.62 seconds on average. In TSVNS, $TSVNS_{2NS}$ runs 3.20 seconds on average and has the shortest CPU time in all metaheuristics. The average run time of $TSVNS_{2NL}$ and $TSVNS_{3N}$ increases to 130.91 seconds and 26.84 seconds respectively. The sharp increase on average run time of $TSVNS_{2NL}$ results from the increase in neighborhood size. In $TSVNS_{2NL}$, if the status are different, consecutive 3 facility status are changed, therefore, the increase in problem size affects deeply the average

run time. The average run time of GAVNS is 254.87 seconds. While $GAVNS_{20}$ runs 52.30 seconds on average, the average run time of $GAVNS_{40}$ and $GAVNS_{60}$ is 117.71 seconds and 594.60 seconds respectively. It is easily observed that the increase in population number linearly increases the average run time.
According to the results in terms of both deviation and average run time, $TSVNS_{3N}$ still outperforms the other metaheuristics for n=50x50.

For n=100x100; the deviation from the upper bounds on average with SAVNS is 2.845%. $SAVNS_{LIN}$ outperforms the $SAVNS_{EXP}$ and $SAVNS_{HYP}$ and has 2.731% deviation. Like n=10x10 and n=50x50; TSVNS finds better solution and has 2.07 % deviation on average for all problem instances. In TSVNS, the larger deviation from the upper bounds results from $TSVNS_{2NS}$ as 2.21% on average. $TSVNS_{2NL}$ and $TSVNS_{3N}$ runs with 0.121% and 0.092% deviation on average. The GAVNS, like SAVNS, also does not run as efficient as TSVNS. The average deviations from the upper bounds are 2.88%, 2.51% and 2.84% for $GAVNS_{20}$, $GAVNS_{40}$ and $GAVNS_{60}$, therefore on average GAVNS runs with 2.74% error. The increase on population size does not result the better solution, for n=50x50, $GAVNS_{40}$ gives better results than $GAVNS_{60}$.

The average run time of SAVNS is still shorter than TSVNS and GAVNS and it is 17.83 seconds on average. $TSVNS_{2NS}$outperforms $TSVNS_{2NL}$ and $TSVNS_{3N}$ in terms of CPU time. While $TSVNS_{2NS}$ runs in 61.72 seconds on average, $TSVNS_{2NL}$ and $TSVNS_{3N}$ run in 742.88seconds and 186.97 seconds on average. GAVNS is still the slowest algorithm and the average run time increases to 680.51 seconds. The average run times of $GAVNS_{20}$, $GAVNS_{40}$ and $GAVNS_{60}$ are 358.81 seconds, 577.04 seconds, 1105.69seconds.

When we compare the SAVNS, TSVNS and GAVNS for n=100x100; $TSVNS_{3N}$ provides better results without much longer run times.

When the population size increases to n=200x200; the deviation from the upper bound and run times increase a bit more for SAVNS. On average, SAVNS has 9.03 % deviation.

The average deviation with $SAVNS_{LIN}$ is 8.98% whereas $SAVNS_{EXP}$ and $SAVNS_{HYP}$ are 9.04% and 0.908% on average. TSVNS still outperforms SAVNS and GAVNS and has 7.56% deviation on average. In TSVNS, the average deviations of $TSVNS_{2NS}$ and $TSVNS_{2NL}$ are 7.84% and 7.42%. On the other hand, the average deviation from UB is 7.42 for $TSVNS_{3N}$ and still beats the others. GAVNS has 0.316% on average. The average deviations of $GAVNS_{20}$, $GAVNS_{40}$and $GAVNS_{60}$ are 8.58% 8.75% 0.254% respectively.

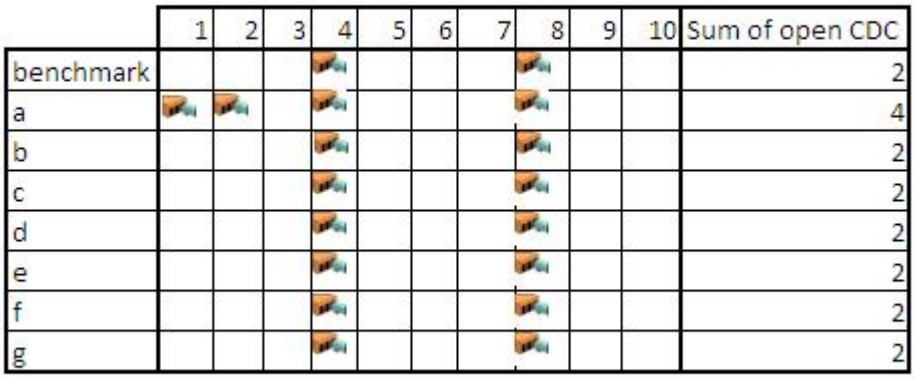
Even though, the average run time of SAVNS increases to 89.48seconds, it is still faster than TSVNS and GAVNS. In SAVNS, the faster algorithm is $SAVNS_{HYP}$ and runs in 95.74 seconds on average. $SAVNS_{LIN}$ and $SAVNS_{EXP}$ run in 88.05 seconds and 84.65 seconds on average. $TSVNS_{2NS}$ runs in 568.82 seconds on average, while $TSVNS_{2NL}$ and $TSVNS_{3N}$ run in 9697.76 seconds and 12547.75 seconds on average. GAVNS is still the slowest algorithm and average run time reaches to 5704.75seconds. From Table 5.21 the results show that the problem is getting harder and being time consuming as problem size increases. In GAVNS, the average run time of $GAVNS_{20}$, $GAVNS_{40}$and $GAVNS_{60}$ are 1069.75 seconds, 2028.44 seconds and 8098.57 seconds respectively.

Throughout all the experiments and analysis, the results presented above indicate that the TSVNS, especially $TSVNS_{3N}$,is competitive with GAVNS for n=10x10 and outperforms all others for n= 50x50, 100x100 and 200x200. Run times presented in Tables 5.19-5.21 also support the efficiency of the TSVNS algorithm.

5.4.2 Effect of Parameters on test instances

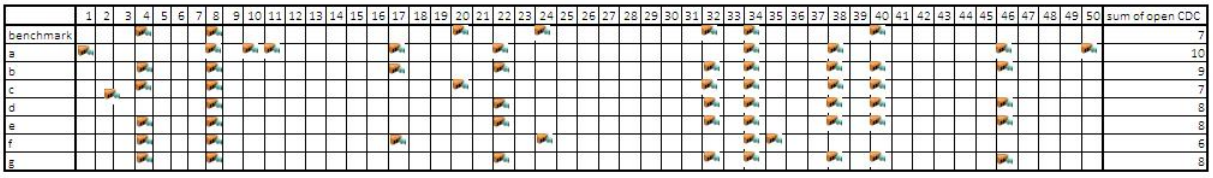
To examine the effect of the parameters on data generation, we create 7 different problem instances. In these instances, we keep all parameters constant except the one whose effects are analyzed. In each instances, the effect of parameter change on the solution can be seen for 10x10 and 50x50 in the Figure 5.1 and 5.2.

In instance "a", we concentrate on the demand rate. When the demand rate is dou-



	1	2	3	4	5	6	7	8	9	10	Sum of open CDC
benchmark											2
a											4
b											2
c											2
d											2
e											2
f											2
g											2

Figure 5.1: Location of the opened CDC for problem size 10x10



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	sum of open CDC
benchmark																																																			7
a																																																			10
b																																																			9
c																																																			7
d																																																			8
e																																																			8
f																																																			6
g																																																			8

Figure 5.2: Location of the opened CDC for problem size 50x50

bled, the result increases more than twice for n=10x10 and n=50x50. Also, the number of opened facilities increases in instance "a". The opened facilities in the original case are still opened, moreover, new facilities are used to served the customers. But when the problem size increases, the effect of the doubled return rate decreases. For n=200x200, the doubled return rate can not double the profit on average.

In instance "b"; the return rate is doubled by keeping the demand rate constant. Unlike instance "a", the increase on the results is not doubled. This explains that the effect of return rate on the total profit is much smaller than the demand rate effect. In instance "b", the number of opened facilities and locations remain the same with the original case.

In instance "c"; the fixed ordering costs are doubled on average. However, the results remain nearly constant. Like instance "b"; the results present that the effect of the doubled fixed ordering cost on total profit is not so big as doubled return rate. When we analyze the facility status, the number of opened facilities number and locations are the same with the original case.

In instance "d" and "e"; the holding cost of products and returns are doubled. However, like instance "b" and "c" the results remain constant. Like instance "b"; the results present that the effect of the doubled fixed ordering cost on total profit is not so big as doubled return rate.When we analyze the facility status, the opened facility number and location is same with the original case. When we compare the effects of hp_i and hr_i, it is observed that the effect of hp_i is more on total profit.

In instance "f"; the fixed operation costs(F) are doubled on average. For n=10x10, the results decrease nearly 70% from the original case. However, when the problem sizes are larger, the effect of "F" decreases on total profit.

Finally, in instance "g"; the transportation costs are doubled on average. Like instance "f"; the results decrease nearly 80% for n=10x10. Again, when the sizes are larger, the effects of the transportation costs are smaller.

From the results of all the experiments, according to our generated data, the change on demand rate affects the profit more than any other parameter.

Chapter 6

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

6.1 Conclusion

Growing environmental awareness leads to increase the number of scientific publications in the field of reverse supply chain. In this thesis, we study on a mixed integer location and pricing model for closed loop supply chain. The differentiating aspect of our model is that the pricing and location decisions are done simultaneously while optimizing the total profit. Under the name of pricing decision, differently from the literature, optimum incentive value of a return is also analyzed. The model is strongly NP-hard and was not studied in the literature before.

We provide three hybrid metaheuristic algorithms to solve the model. The most well-known three metaheuristics, Simulated Annealing, Tabu Search and Genetic Algorithm are hybridized with Variable Neighborhood Search and named as SAVNS, TSVNS and GAVNS. We generate new data set for our new model that can be used in further studies. To evaluate the effects of the parameters used in the data set, we provide different types of test instances with different problem sizes.

Using different neighborhood structures, we study with wider neighborhoods and we compare the performances of the SAVNS, TSVNS and GAVNS. Since the optimal solution is not found by using any commercial program like GAMS, we try to find an upper bound to measure the quality of the solutions. Using a linearization technique for inventory part of the model, we achieve the upper bounds by the help of commercial solver, GAMS.

Our computational study shows that the TSVNS gives better result than SAVNS and

GAVNS in terms of revenue gained for all instances. Despite of the fact that the run time of SAVNS is slightly shorter than TSVNS, the deviation of SAVNS from the best found solution is larger than TSVNS.

The success of TSVNS may resulted from the following factors: Firstly, $TSVNS_{3N}$ finds better results than other metaheuristic methods. The different neighborhood structure (3-swap) found in $TSVNS_{3N}$ creates a wider neighborhood. Consequently, the extensive solution space can be searched and the algorithm has more chance to find a better solution without being trapped. Secondly, accepting a nonimproved solution strategy in TSVNS is more systematic than SAVNS. SAVNS decides any rejection with a random probability, TSVNS choose the solution by controlling the history.

Despite of the complexity of the model, TSVNS gives better results in terms of solution quality and runs fast. For this reason, the proposed algorithm yields a viable solution method and can be used in real life problems.

6.2 Future Research

6.2.1 Exact Method Development

As mentioned in the previous section, the model has not been found in literature before and ia strongly NP-hard. For this reason, an exact method may be time consuming for solving the problem. However, a well-developed branch and bound algorithm may be proposed to solve the problem at optimality.

6.2.2 Upper Bound Improvement

We analyze the effectiveness of any metaheuristic by comparing the average results with the best found solution. Therefore, more effective upper bounds may be found using relaxation techniques for further studies.

6.2.3 Limitations to the Model

In our model, some characteristics of the real problem have not been considered such as CDC capacity limits, multiproduct production, quality differentiation for return type and uncertainty in demand and return rate. For further studies, the model may be extended to cover these limitations.

BIBLIOGRAPHY

[1] Srivastava K.,Green supply-chain management: A state of the art literature review.*International Journal of Management Reviews* Volume 9 Issue 1 pp. 5380, 2007.

[2] Clark J.M., Business Acceleration and the Law of Demand: A Technical Factor in Economic Cycles. *Journal of Political Economy* Vol. 25, No. 3, pp. 217-235, 1917.

[3] Brandeau M.L.,Chiu S.S., An Overview of Representative Problems in Location Research. *Management Science*, Vol. 35, pp. 645-674,1989.

[4] S.L. Hakimi, Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Researc*h 12pp. 450-459, 1964.

[5] Mladenovic N.,Brimberg J.,Hansen P., Moreno-Prez J.A., The p-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research*, Volume 179, Issue 3, pp 927-939, 2007.

[6] C.S. ReVelle, H.A. Eiselt, M.S. Daskin, A bibliography for some fundamental problem categories in discrete location science. *European Journal of Operational Research*, Volume 184 ,pp 817848., 2008.

[7] P.B. Mirchandani, R.L. Francis, Discrete Location Theory, *Wiley*, New York, 1990.

[8] D.S.Rogers, Tibben-Lembke R.S., Going Backwards: Reverse Logistics Trends and Practices.*Reverse Logistics Executive Council* ISBN 0-9674619-0-1,Pittsburgh, PA, 1999.

[9] Spengler T., Pchert H., Penkuhn T.,Rentz O.,Environmental integrated production and recycling management.European Journal of Operational Research Volume 97, Issue 2, pp. 308326, 1997.

[10] Barros A.I.,Dekker R., Scholten V., A two-level network for recyclingsand: A case study. *European Journal of Operational Research* Volume 110, Issue 2, pp. 199214, 1998.

[11] Listes O.,Dekker R., A stochastic approach to a case study for product recovery network design. *European Journal of Operational Research* Volume 160, Issue 1, pp. 268287, 2005.

[12] Louwers D., Kip B.J., Peters E., Souren F., Flapper S.D.P., A facility location allocation model for reusing carpet materials. *Computers & Industrial Engineering* Volume 36, Issue 4, pp. 855869, 1999.

[13] Krikke H.R., Harten A. van, Schuur P.C., Reverse logistic network re-design for copiers *OR SPECTRUM* Volume 21, Number 3, pp. 381-409, 1999.

[14] Jayaraman V. , Guide Jr. V. D. R., Srivastava R., A Closed-Loop Logistics Model for Remanufacturing. *The Journal of the Operational Research Society* Vol. 50, No. 5, pp. 497-508, 1999.

[15] M. Fleischmann, Reverse logistics network structures and design, ERIM Report Series Research In Management, ERS-2001-52-LIS, Erasmus University Rotterdam, 2001.

[16] Shih L.H, Reverse logistics system planning for recycling electrical appliances and computers in Taiwan, *Conservation and Recycling* Volume 32, Issue 1, pp. 5572, 2001

[17] Jayaraman V., Patterson R.A., Rolland E.,The design of reverse distribution networks: Models and solution procedures, *European Journal of Operational Research* Volume 150, pp.128149, 2003

[18] Schultmann F., Engels B.,and Rentz O., Closed-Loop Supply Chains for Spent Batteries. *Informs*, Volume 33, No. 6, pp. 57-71, 2003.

[19] Kusumastuti R. D., Piplani R., Lim G. H.,An Approach to Design Reverse Logistics Networks for Product Recovery Centre for Supply Chain Management, *Engineering Management Conference*, IEEE International, 2004.

[20] J. B. Sheu , Y. H. Chou, and C.C. Hu, An integrated logistics operational model for green-supply chain management. *Transportation Research Part E: Logistics and Transportation Review* Rev., Volume 41, No. 4, pp. 287-313, 2005.

[21] A.Nagurney, F. Toyasaki, Reverse supply chain management and electronicwaste recycling: a multitiered network equilibrium framework for e-cycling. *Transportation Research Part E: Logistics and Transportation Review* Volume 41, Issue 1, pp. 128, 2005.

[22] F.Schultmann, M.Zumkeller, O.Rentz, Modeling reverse logistic tasks within closed-loop supply chains: An example from the automotive industry.*European Journal of Operational Research*, Volume 171, issue 3, pp. 1033-1050, 2006.

[23] M. Biehl, E. Prater, M.J. Realff, Assessing performance and uncertainty in developing carpet reverse logistics systems. *Computers & Operations Research*, Volume 34, Issue 2, pp. 443463, 2007

[24] Z. Lu, N. Bostel, A facility location model for logistics systems including reverse flows: The case of remanufacturing activities. *Computers & Operations Research* Volume 34, Issue 2, Pages 29932, 2007.

[25] H.J. Ko, G. W. Evans, A genetic algorithm-based heuristic for the dynamic integrated forward/reverse logistics network for 3PLs . *Computers & Operations Research* Volume 34, Issue 2, pp. 346366, 2007.

[26] K. Lieckens, N. Vandaele, Reverse logistics network design with stochastic lead times.*Computers & Operations Research* Volume 34, Issue 2, pp. 395416, 2007.

[27] O. Listes, A generic stochastic model for supply-and-return network design. *Computers & Operations Research* Volume 34, Issue 2, pp. 417442, 2007.

[28] M.I.G.Salema, A.P.Barbosa-Povoa, A.Q.Novais, An optimization model for the design of a capacitated multi-product reverse logistics network with uncertainty.*European Journal of Operational Research*, Volume 179, issue 3, pp. 1063-1077, 2007.

[29] R. D. Kusumastuti, R. Piplani, and G. Lim, Redesigning closed-loop service network at a computer manufacturer: A case study. *International Journal of Production Economics*, Volume 111, Issue 2, pp. 244-260, 2008.

[30] J. B. Sheu, Green supply chain management, reverse logistics and nuclear power generation. *Transportation Research* Part E, Volume 44, Issue. 1, pp. 19-46, 2008.

[31] N.Aras, D. Aksen, A. G. Tanugur. Locating collection centers for incentive-dependent returns under a pick-up policy with capacitated vehicles. *European Journal of Operational Research* Volume 191, Issue 3, Pages 12231240, 2008.

[32] R. K. Pati, P. Vrat, and P. Kumar, A goal programming model for paper recycling system, *Omega*, Volume. 36, no. 3, pp. 405-417, 2008.

[33] H. Kim, J. Yang, K.D Lee, Vehicle routing in reverse logistics for recycling end-of-life consumer electronic goods in South Korea. *Transportation Research Part D: Transport and Environment*, Volume 14, Issue 5, pp. 291299, 2009.

[34] H. Min, H.J.Kob, The dynamic design of a reverse logistics network from the perspective of third-party logistics service providers. *International Journal of Production Economics*, Volume 113, Issue 1, pp. 176192, 2008.

[35] M. El-Sayed, N. Afia, A. El-Kharbotly, A stochastic model for forwardreverse logistics network design under risk, Design and Production Engineering Dept., Ain-Shams University, Faculty of Engineering, Cairo, Egypt

[36] Y.Zhou, S. Wang, Generic Model of Reverse Logistics Network Design, *Journal of Transportation Systems Engineering and Information Technology* Volume 8, Issue 3, pp. 7178, 2008

[37] Feng Du, G.W. Evans, A bi-objective reverse logistics network analysis for post-sale service. *Computers & Operations Research*, Volume 35, Issue 8, pp. 26172634, 2008.

[38] A.Mutha, S. Pokharel, Strategicnetworkdesign for reverselogistics and remanufacturing using new and oldproductmodules. *Computers & Industrial Engineering*, Volume 56, Issue 1, pp. 334346, 2009.

[39] R.Cruz-Rive, J. Ertel, Reverse logistics network design for the collection of End-of-Life Vehicles in Mexico. *European Journal of Operational Research* Volume 196, Issue 3, pp. 930939, 2009.

[40] E. Iakovou, N. Moussiopoulos, A. Xanthopoulos, Ch. Achillas, N. Michailidis, M. Chatzipanagioti, C. Koroneos, K.-D. Bouzakis, V. Kikis A methodological framework for end-of-life management of electronic products. *Conservation and Recycling*,Volume 53, Issue 6, pp. 329339, 2009.

[41] M. Grunow, C. Gobbi, Designing the reverse network for WEEE in Denmark. *CIRP Annals - Manufacturing Technology*, Volume 58, pp.391394, 2009.

[42] G. Kannan, P. Sasikumar, K. Devika, A genetic algorithm approach for solving a closed loop supply chain model: A case of battery recycling. *Applied Mathematical Modelling* Volume 34, Issue 3, pp. 655670, 2010

[43] R. D. Kusumastuti, R. Piplani, and G. Lim, An approach to design reverse logistics networks for product recovery, In Proceedings of IEEE international engineering management conference, Singapore, pp. 1239-1243, 2004.

[44] M.S.Pishvaee, R.Z.Farahani, W.Dullaert, A memetic algorithm for bi-objective integrated forward/reverse logistics network design. *Computers and Operations Research*Volume 37, Issue 6, 2010.

[45] F. Altiparmak ,M. Gen, L.Lin, T.Paksoy, A genetic algorithm approach formulti- objective optimization of supply chain networks. *Computers and Industrial Engineering* Volume 51, Issue 1, pp. 196215, 2006.

[46] M.I.G. Salema, A.P.Barbosa-Povoa, A.Q. Novais, Simultaneousdesign and planning of supplychains with reverse flows: A genericmodellingframework. *European Journal of Operational Research* Volume 203, Issue 2, pp 336349, 2010

[47] S. Kara, F. Rugrungruang, and H. Kaebernick, Simulation modelling of reverses logistics networks. *International Journal of Production Economics*, vol. 106, no. 1, pp. 61-69, 2007.

[48] B.M. Beamon, C. Fernandes, Supply-chain network configuration for product recovery. *Production Planning & Control: The Management of Operations*, Volume 15, Issue 3, 2004.

[49] M. Chouinard, S. DAmours, D. Ait-Kadi, Integration of reverse logistics activities within a supply chain information system. *Computers in Industry* Volume 56, Issue 1, Pages 105-124, 2005.

[50] K.Inderfurth, Impact of uncertainties on recovery behavior in a remanufacturing environment: A numerical analysis. *International Journal of Physical Distribution and Logistics Management*, Volume 35, Issue 5, pp.318-336, 2005.

[51] S. Kumar, P. Malegeant, Strategic alliance in a closed-loop supply chain, a case of manufacturer and eco-non-profit organization. *Technovation* Volume 26, Issue 10, Pages 11271135, 2006.

[52] Network design for reverse logistics Samir K. Srivastava. *Omega* Volume 36, Issue 4, pp. 535548, 2008.

[53] D.H. Lee, M. Dong, A heuristic approach to logistics network design for end-of-lease computer products recovery. *Transportation Research Part E: Logistics and Transportation Review* Volume 44, Issue 3, pp 455474, 2008.

[54] M.V.Fuente, L. Rosa, M. Cardo, Integrating Forward and Reverse Supply Chains: Application to a metal-mechanic company. *International Journal of Production Economics* Volume 111, Issue 2, pp 782792, 2008.

[55] S.Z. Selim, K. Alsultan,A simulated annealing algorithm for the clustering problem. *Pattern Recognition*, Volume 24, Issue 10, pp. 10031008, 1991.

[56] A.T. Murray, R.L. Church, Heuristic solution approaches to operational forest planning problems. *OR Spektrum*,Volume 17, pp. 193-203, 1995

[57] J.Nelson ,JD Brodie,Comparison of random search algorithm and mixed integer programming for solving area-based forest plans. *Canadian Journal of Forest Research*,

Volume 20, pp. 934-942, 1990.

[58] A.T. Murray ,R.L. Church, Applying simulated annealing to location-planning models. *Journal of Heuristics* Volume 2, pp. 3153, 1996.

[59] J.E. Beasley, OR-Library: Distributing test problems by electronic mail. *Journal of the Operational Research Society* Volume 41, pp.10691072, 1990.

[60] R.K. Kincaid, Good solutions to discrete noxious location problems via metaheuristics, *Annals of Operations Research* Volume 40, pp.265-281, 1992.

[61] F. Chiyoshi and R.D.Galvo, A statistical analysis of simulated annealing applied to the p-median problem.*Annals of Operations Research*, Volume 96, pp. 61-74, 2000.

[62] M.B. Teitz and P. Bart, Heuristic methods for estimating the generalized vertex median of a weighted graph, *Operations Research* Volume 16, pp.955961, 1968.

[63] T.V. Levanova and M.A. Loresh, Algorithms of Ant System and Simulated Annealing for the p-median Problem. *Automation and Remote Control*, Volume 65, pp.431-438, 2004.

[64] M.A. Arostegui, S.N. Kadipasaoglu, B.M. Khumawala, An empirical comparison of Tabu Search, Simulated Annealing, and Genetic Algorithms for facilities location problems *International Journal of Production Economics* Volume 103, Issue 2, pp. 742754, 2006.

[66] E. Rolland, D.A. Schilling, J.R. Current, An efficient tabusearch procedure for the p-Median Problem. *European Journal of Operational Research* Volume 96, Issue 2, pp. 329342, 1997.

[66] E. Rolland, D. A. Schilling, J.R. Current, An efficient tabusearch procedure for the p-Median Problem. *European Journal of Operational Research* Volume 96, Issue 2, pp. 329342, 1997.

[67] K.E. Rosing, C.S. ReVelle, E. Rolland, D.A. Schillingd, J.R. Current, Heuristic concentration and Tabu search: A head to head comparison.*European Journal of Operational Research* Volume 104, Issue 1, pp 9399, 1998.

[68] L.Michel, P.V. Hentenryck,A simple tabu search for warehouse location. *European Journal of Operational Research* Volume 157, Issue 3, pp. 576591, 2004.

[69] K.S. Al-Sultan and M.A. Al-Fawzan, A tabu search approach to the uncapacitated facility location problem.*Annals of Operations Research* Volume 86, pp.91-103, 1999.

[70] C. M. Hosage, M. F. Goodchild, Discrete space location-allocation solutions from genetic algorithms. *Annals of Operations Research* Volume 6, Issue 2 pp. 35-46, 1986.

[71] D.Gong, M. Gen, G. Yamazaki, W. Xu, Hybrid evolutionary method for capacitated location-allocation problem. *Computers & Industrial Engineering* Volume 33, Issues 34, pp. 577580, 1997.

[72] Cooper, L., Heuristic Methods for Location- Allocation Problems, *SIAM Review*, Vol.

6, No. 1, pp. 1-18, 1964.

[73] J. H. Jaramillo,J. Bhadury, R. Batta, On the use of genetic algorithms to solve location problems. *Computers & Operations Research* Volume 29, Issue 6, pp.761779, 2002.

[74] E.S. Correa, M.T.A. Steiner, A.A. Freitas, C. Carnieri, A Genetic Algorithm for the P-median Problem. *Genetic and Evolutionary Computation Conference* (GECCO-2001).

[75] X. Li, A.G. Yeh, Integration of genetic algorithms and GIS for optimal location search. *International Journal of Geographical Information Science*, Volume 19, Issue 5, 2005.

[76] M. N. Neema, K. M. Maniruzzaman, A. Ohgai, New Genetic Algorithms Based Approaches to Continuous p-Median Problem. *Networks and Spatial Economics* Volume 11, Number 1,pp. 83-99, 2011.

[77] S.Sasaki, A. J. Comber, H. Suzuki, C. Brunsdon, Using Genetic Algorithms to Optimise Current and Future Health Planning - The example of ambulance locations. *International Journal of Health Geographics* Volume 9, Number 4, 2010.

[78] D. A. Schrady, A deterministic inventory model for repairable items. *Naval Research Logistics Quarterly*, Volume 14, Issue 3, pp.391398, 1967.

[79] S. Nahmias,H. Rivera, A deterministic model for a repairable item inventory system with a finite repair rate. *International Journal of Production Research*, Volume 17, Issue 3, pp. 215221, 1979.

[80] M. C. Mabini,L. M. Pintelon, L. F.Gelders, EOQ type formulation for controlling repairable inventories. *International Journal of Production Economics* Volume 28, pp.2133, 1992.

[81] J.D. Camm, T.E. Chorman, F.A. Dull, J.R. Evans, D.J. Sweeney, G.W. Wegryn, Blending OR/MS, judgement, and GIS: Restructuring Procter and Gamble's supply chain, *Interfaces*, Volume 27, pp.128-142, 1997.

[82] M. S. Daskin, C R. Coullard, Z-J Shen, An Inventory-Location Model: Formulation, Solution Algorithm and Computational Results. *Annals of Operations Research*, Volume 110, Numbers 1-4, pp.83-106, 2002

[83] Z.J. Shen, Efficient algorithms for various supply chain problems, Ph.D. dissertation, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL (2000).

[84] Z.J. Shen, C.R. Coullard and M.S. Daskin, A joint location-inventory model, *Transportation Science* Volume 37, Number 1, pp. 4055, 2003.

[85] R. Wojanowski,T. Boyac, V. Verter, Retail-collection network design under deposit-refund. *Computers & Operations Research* Volume 34, pp.324345,2007.

[86] M.R. Garey and D.S.Johnson, Computers and Intractability : A Guide to the Theory of NP Completeness,Freeman, San Francisco, 1979.

[87] R. Conn, G. Cornuejols, A projection method for the uncapacited facility location problem, Mathematical Programming Volume 46, 273298, 1990.

VITA

Buşra Ürek was born in Ankara, Turkey, on March 28, 1986. She received her B.Sc. in Chemical Engineering from Middle East Technical University, Ankara, in 2009. In September 2009, she started her M.Sc. degree at Koç University. Until July 2011, she worked as a teaching assistant. Her research focus was on Reverse Logistics Design and Metaheuristics.