Athens University of Economics and Business

School of Business, Department of Management Science and Technology

Doctoral Program in Operations Research

Models and Solution Algorithms for Inventory Routing Problems

by

Pantelis Z. Lappas

Dissertation submitted for partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Athens, May 2017

Athens University of Economics and Business

School of Business, Department of Management Science and Technology

Doctoral Program in Operations Research

Models and Solution Algorithms for Inventory Routing Problems

by

Pantelis Z. Lappas

Committee

Manolis Kritikos (Supervisor)

Assistant Professor of Operations Research and Information Systems

Athens University of Economics and Business
School of Business
Department of Management Science and Technology

George Ioannou

Professor of Production and Operations Management

Athens University of Economics and Business
School of Business
Department of Management Science and Technology

Apostolos Burnetas

Professor of Operations Research

National and Kapodistrian University of Athens
School of Science
Department of Mathematics

To my wife, and my parents

P. Lappas

Περίληψη

Στόχος της παρούσας διατριβής είναι η παρουσίαση αλγοριθμικών προσεγγίσεων για την επίλυση του Προβλήματος Δρομολόγησης Αποθεμάτων (Inventory Routing Problem, IRP) και του Προβλήματος Δρομολόγησης Αποθεμάτων με Χρονικά Παράθυρα (Inventory Routing Problem with Time Windows, IRPTW). Τα ανωτέρω προβλήματα πηγάζουν από την προσέγγιση της Διαχείρισης Αποθεμάτων από τον Προμηθευτή/Πωλητή (Vendor Managed Inventory, VMI) που διαδόθηκε ιδιαίτερα κατά τα τέλη της δεκαετίας του ’80 από τις Wal-Mart και Procter & Gamble και στη συνέχεια υιοθετήθηκε από πολλές εταιρίες όπως οι Johnson & Johnson, Black & Decker κ.ά. Σύμφωνα με το VMI, ο προμηθευτής διανέμει προϊόντα σε έναν αριθμό από γεωγραφικά διάσπαρτους πελάτες αποφασίζοντας ταυτόχρονα για τα ακόλουθα: (1) τους χρόνους εξυπηρέτησης πελατών, (2) τις ποσότητες διανομής και (3) τις διαδρομές που πρέπει να ακολουθηθούν. Οι πρώτες δύο αποφάσεις, σχετίζονται με το Πρόβλημα Ελέγχου Αποθεμάτων (Inventory Control Problem, ICP), ενώ η τρίτη με το Πρόβλημα της Δρομολόγησης Οχημάτων (Vehicle Routing Problem, VRP).

Αξίζει να σημειωθεί πως το IRPTW αποτελεί βασική επέκταση του IRP, καθώς ισχύουν οι ίδιοι περιορισμοί, αλλά για κάθε πελάτη η εξυπηρέτηση πρέπει να ξεκινήσει και να ολοκληρωθεί μέσα σε ένα χρονικό παράθυρο (time window), ενώ το όχημα θα παραμένει στο χώρο του πελάτη για συγκεκριμένο χρόνο εξυπηρέτησης. Κατά συνέπεια, το IRPTW αποτελεί σύνθεση του ICP και του Προβλήματος Δρομολόγησης Οχημάτων με Χρονικά Παράθυρα (Vehicle Routing Problem with Time Windows, VRPTW).

Η διαφοροποίηση των προβλημάτων δρομολόγησης αποθεμάτων έναντι των υπολοίπων προβλημάτων δρομολόγησης (routing problems) οφείλεται στον παράγοντα απόθεμα, ο οποίος προσθέτει στο πρόβλημα τη διάσταση του χρόνου. Ως εκ τούτου, τα IRP και IRPTW αντιμετωπίζονται ως προβλήματα πολλαπλών περιόδων (multi-period problems). Ο παράγοντας απόθεμα περιπλέκει το πρόβλημα σε δύο διαστάσεις. Πρώτον, η περιορισμένη δυνατότητα διατήρησης αποθέματος στον προμηθευτή και/ ή στους πελάτες θα πρέπει να λαμβάνεται υπόψη όταν αποφασίζονται οι ποσότητες που θα διανεμηθούν, ενώ τυχόν κόστη που συνδέονται με τη διατήρηση αποθέματος στον προμηθευτή ή τους πελάτες πρέπει να συμπεριλαμβάνονται στην αντικειμενική συνάρτηση. Τα προβλήματα δρομολόγησης αποθεμάτων ανήκουν στην κλάση πολυπλοκότητας NP και χαρακτηρίζονται ως NP-δυσχερή (NP-Hard), καθώς περικλείουν το κλασικό πρόβλημα της δρομολόγησης οχημάτων.

Με τη μαθηματική μοντελοποίηση των προβλημάτων παρουσιάζεται, επιπλέον, για κάθε πρόβλημα μία αντίστοιχη αλγοριθμική επίλυση. Στην περίπτωση του IRP, η αντικειμενική συνάρτηση του προβλήματος αναπαριστά το συνολικό κόστος που αποτελείται από το κόστος μεταφοράς (transportation cost) και το κόστος

αποθήκευσης/διατήρησης αποθέματος (inventory holding cost) στους πελάτες. Για το IRPTW, η αντικειμενική συνάρτηση του προβλήματος αναπαριστά μόνο το συνολικό κόστος μεταφοράς.

Λόγω της NP-hard φύσης του IRP προτείνεται ένας υβριδικός εξελικτικός αλγόριθμος βελτιστοποίησης (hybrid evolutionary optimization algorithm) που αξιοποιεί δύο ευρέως γνωστούς μεθευρετικούς αλγόριθμους (meta-heuristics): τον Γενετικό Αλγόριθμο (Genetic Algorithm, GA) και τον Αλγόριθμο της Προσομοιωμένης Ανόπτησης (Simulated Annealing Algorithm, SA). Ο GA αξιοποιείται στη φάση του σχεδιασμού (planning) όπου καθορίζονται οι προγραμματισμένες προς αποστολή ποσότητες προϊόντος (delivery quantities), καθώς επίσης και οι χρονικές στιγμές του ορίζοντα όπου οι πελάτες θα λάβουν τις σχετικές ποσότητες (delivery times). Ο SA χρησιμοποιείται στη φάση της δρομολόγησης (routing) για την επίλυση των προβλημάτων δρομολόγησης που προκύπτουν σε κάθε περίοδο του χρονικού ορίζοντα. Τα αποτελέσματα των δύο αλγορίθμων συνδυάζονται επαναληπτικά έως την εύρεση της βέλτιστης λύσης του προβλήματος.

Όσον αφορά το IRPTW, παρουσιάζεται ένας αλγόριθμος επίλυσης δύο φάσεων (two-phase solution algorithm) που βασίζεται σε μία απλή Προσομοίωση (simple simulation) για τη φάση του σχεδιασμού και στον Αλγόριθμο Μεταβλητής Γειτονιάς Αναζήτησης (Variable Neighborhood Search, VNS) για τη φάση της δρομολόγησης. Τέλος, για τη μέτρηση της αποτελεσματικότητας των δύο προτεινόμενων αλγοριθμικών προσεγγίσεων, νέα δεδομένα προβλημάτων (benchmark instances) έχουν σχεδιαστεί για τα IRP και IRPTW, ενώ παρουσιάζονται αναλυτικά υπολογιστικά αποτελέσματα επί των προβλημάτων.

Λέξεις Κλειδιά: Δρομολόγηση, Πρόβλημα Δρομολόγησης Αποθεμάτων, Πρόβλημα Δρομολόγησης Αποθεμάτων με Χρονικά Παράθυρα, Γενετικός Αλγόριθμος, Αλγόριθμος Προσομοιωμένης Ανόπτησης, Εξελικτική Βελτιστοποίηση, Προσομοίωση, Αλγόριθμος Μεταβλητής Γειτονιάς Αναζήτησης

Abstract

The main objective of this thesis is to propose a hybrid evolutionary optimization algorithm for solving the Inventory Routing Problem (IRP). The IRP arises from the application of the Vendor Managed Inventory (VMI) concept, where the supplier (vendor) has to make inventory and routing decisions simultaneously for a given planning horizon. This thesis focuses on a scenario where a single-product type has to be delivered by a fleet of capacitated homogenous vehicles and housed at a depot over a finite and discrete planning horizon. The demand is fully available to the decision maker (supplier) at the beginning of the planning horizon, stock-outs are not allowed, and transportation costs and inventory holding costs of customers are taken into account in the objective function. Due to the NP-hard nature of the IRP, it is very difficult to develop an exact algorithm that can solve large-scale problems within a reasonable computation time. As an alternative, a hybrid evolutionary optimization algorithm based on two well-known meta-heuristics, the Genetic Algorithm and the Simulated Annealing Algorithm, is presented to handle the IRP. Namely, the Genetic Algorithm is related to the planning phase, while the Simulated Annealing Algorithm is associated with the routing phase. A repetitive procedure, containing characteristics from both referred meta-heuristics, is applied to obtain a near-optimal feasible solution. Testing instances with different properties are established to investigate algorithmic performance, and the computational results are then reported.

Finally, a two-phase solution algorithm is presented to handle an extension of the IRP, the Inventory Routing Problem with Time Windows (IRPTW). The IRPTW, which has not been excessively researched in the literature, is a generalization of the standard IRP involving the added complexity that every customer should be served within a given time window. A single-product type has to be delivered by a fleet of capacitated homogenous vehicles and housed at a depot over a finite and discrete planning horizon. The demand is fully available to the decision maker (supplier) at the beginning of the planning horizon, stock-outs are not allowed, and only transportation costs are taken into account in the objective function. The proposed two-phase solution algorithm is based on (a) a simple simulation for the planning phase and (b) the Variable Neighborhood Search Algorithm (VNS) for the routing phase. The computational study underscores the importance of integrating the inventory and vehicle routing decisions. Analytical results and graphic presentation formats are provided to convey meaningful insights into the problem.

Keywords: Routing, Inventory Routing Problem, Inventory Routing Problem with Time Windows, Genetic Algorithm, Simulated Annealing Algorithm, Evolutionary Optimization, Simulation, Variable Neighborhood Search

Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

ALNS	Adaptive Large Neighborhood Search
ATCH	Approximate Transportation Costs Heuristic
CARE	Clustering, Allocation, Routing, Extended
ETCH	Estimated Transportation Costs Heuristic
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
ICP	Inventory Control Problem
ILS	Iterated Local Search
IRP	Inventory Routing Problem
IRPSTW	Inventory Routing Problem with Soft Time Windows
IRPTW	Inventory Routing Problem with (Hard) Time Windows
LNS	Large Neighborhood Search
ML	Maximum Level
OL	Order-up-to Level
PFIH	Push Forward Insertion Heuristic
PRP	Production Routing Problem
PSO	Particle Swarm Optimization
RP	Routing Problem
SA	Simulated Annealing Algorithm
ScM	Supply Chan Management
TS	Tabu Search
TSP	Traveling Salesman Problem
VMI	Vendor Managed Inventory
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with Time Windows

Acknowledgements

First and foremost I would like to thank my supervisor Asst. Prof. Manolis N. Kritikos. I appreciate all his contributions of time, trust and support. My sincere appreciation is extended to Prof. George Ioannou for his invaluable support. I acknowledge my reading committee members: Prof. Apostolos Burnetas, Asst. Prof. Konstantinos Androutsopoulos, Assoc. Prof. George Lekakos, Assoc. Prof. Damianos Chatziantoniou, Asst. Prof. Dimitris Kardaras and Asst. Prof. Alexandros Papalexandris. I would like to thank my wife, Iro Della, for standing beside me throughout all my pursuits. She is my inspiration and motivation for continuing to improve my knowledge and move my career forward. Many thanks to my parents, Zissis and Anastassia, for all their love and encouragement.

Chapter 1

Introduction

In recent years, the Inventory Routing Problem (IRP) has received a great deal of attention from academics, consultants and practitioners. It reflects a multi-functional problem that attempts to integrate two different functions within the supply chain network, i.e., planning and routing (Min and Zhou, 2002). In particular, planning is associated with the Inventory Control Problem (ICP), while routing is related to the Vehicle Routing Problem (VRP). The ICP represents an activity that aims to organize the availability of goods to customers during a given planning horizon (Axsäter, 2006), while the VRP concerns the distribution of goods between suppliers and customers, without taking into account the time scope (Toth and Vigo, 2002). Whereas VRPs typically deal with a single period (e.g., a day), IRPs have to deal with a longer horizon (multi periods: e.g., a sequence of days).

In the context of the IRP, these two widely studied problems in the Operations Research literature are modeled simultaneously since an inter-relationship exists between them (Moin and Salhi, 2007; Archetti and Speranza, 2016). If only the ICP for the customers is concerned and the VRP for the supplier is ignored, the supply chain cost, including the total transportation and total inventory cost, is not minimized optimally, as the VRP decisions cannot be made effectively and vice versa. The IRP arises in environments where Vendor Managed Inventory (VMI) policies are applied. It can be assumed as an extension of the VRP, which integrates routing and inventory allocation decisions. Analytically, the vendor (supplier) monitors the inventory levels of the customers and determines (a) the delivery times (when to visit his customers), (b) the quantities (how much to deliver to each of them when they are served), so that stock-outs are avoided, and (c) the set of routes used by a fleet of vehicles to serve a given set of customers (how to integrate the customers into the vehicle routes).

IRPs can be categorized into three levels (Andersson et al., 2010; Coelho et al., 2013). The first categorization is based on the structural variants presented in IRPs, namely, product, time horizon, network topology, routing, inventory policy, inventory decisions, fleet composition and fleet size. The second categorization is related to the availability of information on the demand, reflecting several types of IRPs, for example, deterministic, stochastic, and dynamic and stochastic IRPs. Moreover, the third categorization is associated with the chosen solution approach. According to Ballou (1989) the modeling of supply chain and logistics problems has traditionally relied on three primary methods, i.e., simulation, optimization (exact algorithm) and heuristics, which can be divided into two categories (Griffis et al., 2012): classic heuristics (construction heuristics, local improvement heuristics) and meta-heuristics (local search meta-heuristics and population search meta-heuristics). The recent

literature has shown an increased interest in so-called matheuristics, methods that combine exact and heuristic approaches (Maniezzo et al., 2009). Archetti and Speranza (2013) classified matheuristics into three classes: decomposition approaches, improvement heuristics and column generation-based approaches.

It is worth noting that IRP decisions can be (a) decisions over time only, in which the delivery times and the quantities have to be determined at the same time, while the routes are given, and (b) decisions over time and space, where delivery times, quantities and routes have to be determined simultaneously (Bertazzi and Speranza, 2012; Bertazzi and Speranza, 2013). Furthermore, the optimal solution of an IRP depends on the objective function that has been chosen (Bertazzi et al., 2008). As a result, an objective function can be (a) the sum of transportation costs only, (b) the sum of transportation and inventory holding costs of the customers or (c) the sum of transportation and inventory holding costs of the supplier and the customers. It should not pass unnoticed that under the VMI concept, stock-outs are not allowed, and therefore, the objective function does not include shortage costs.

In this thesis, the main objective is to propose an approach for solving the IRP with the following characteristics. A single-product type has to be delivered by a fleet of capacitated homogenous vehicles (multiple vehicles) housed at a depot over a finite and discrete planning horizon. The network topology taken into account by the IRP model is one-to-many; that is, one supplier serves many geographically dispersed customers (demand points). A vehicle can visit more than one customer (multiple routing), while a vehicle's trip starts and ends at the depot (supplier). As far as the inventory policy is concerned, a Maximum Level (ML) policy is considered, in which any customer has defined a maximum inventory level and every time a customer is served, the delivered quantity is such that the inventory level at the customer is not greater than the maximum level. It is assumed that the depot has a sufficient supply of products that can cover all customers' demands throughout the planning horizon. Moreover, the inventory is not allowed to become negative (fixed inventory) since the lowest inventory level is either fixed or equal to zero. With respect to the availability of information on customer demand, the proposed IRP model is deterministic since the demand is fully available to the supplier at the beginning of the planning horizon.

Regarding the solution approach, a hybrid evolutionary optimization algorithm that combines a nature-inspired optimization algorithm (local search meta-heuristic), such as the Simulated Annealing Algorithm (SA), as well as a biologically-inspired optimization algorithm (population search meta-heuristic), that is, the Genetic Algorithm (GA), is presented to handle the problem. The SA is associated with the routing decisions (routing phase), while GA is related to the inventory allocation decisions (planning phase). A repetitive procedure, containing characteristics of both meta-heuristics, is applied to obtain a near-optimal feasible solution. In addition, IRP decisions are decisions over time and space, while the objective function represents the sum of transportation and inventory holding costs of the customers.

A second objective of the thesis is to present a two-phase solution algorithm to handle an extension of the IRP, the Inventory Routing Problem with Time Windows (IRPTW). The IRPTW, which has not been excessively researched in the literature, is a generalization of the standard IRP involving the added complexity that every customer should be served within a given time window. A single-product type has to be delivered by a fleet of capacitated homogenous vehicles and housed at a depot over a finite and discrete planning horizon. The demand is fully available to the decision maker (supplier) at the beginning of the planning horizon, stock-outs are not allowed, and only transportation costs are taken into account in the objective function. As far as the inventory policy is concerned, an Order-up-to Level (OL) policy is considered, in which any customer has defined a maximum inventory level and every time a customer is served, the delivered quantity is such that the maximum inventory level at the customer is reached. Moreover, it is assumed that the depot has a sufficient supply of products that can cover all customers' demands throughout the planning horizon. The proposed two-phase solution algorithm is based on (a) a simple simulation for the planning phase and (b) the Variable Neighborhood Search Algorithm (VNS) for the routing phase.

The remainder of this thesis is organized as follows. Chapter 2 presents an overview of the state of the art in research on the Inventory Routing Problems. A problem description and the mathematical formulation for the IRP are presented in Chapter 3. In addition, the proposed solution approach is described and analyzed in detail, while computational results are presented. Chapter 4 is devoted to the presentation of the IRPTW. Finally, Chapter 5 summarizes the main contributions of this thesis and points to some potential research directions.

Chapter 2

Literature Review

Routing problems have attracted attention as a possible solution to many of the complex issues surrounding Supply Chain Management (ScM). In today's economic environment, efficiency for firms is moving from an internal to a supply chain priority since the competition is not among them, but among their supply chains (Croom et al., 2000; Tan, 2001). As a consequence, the ultimate success of a firm depends on its ability to integrate and coordinate different supply chain activities within the supply chain network (Min and Zhou, 2002; Schmid et al., 2013). Routing problem (RP) is the generic name given to a whole class of problems in which transportation is necessary (Diaz-Parra et al., 2014). The issue of RPs can be addressed in two dimensions: (a) classical routing problems, such as the Traveling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP), and (b) highly relevant extensions of classical routing problems like the Inventory Routing Problem (IRP) and the Production Routing Problem (PRP).

The TSP is the most basic routing problem and a typical model of the combinatorial optimization problems whose computation complexity is derived from non-polynomial time (NP-hard problem). In particular, the problem is to find the shortest route (minimum transportation cost) that starts from a depot, visits all customers exactly once, and returns to the depot (Flood, 1956). For a comprehensive review of the proposed solution approaches including exact algorithms, heuristics and meta-heuristics, see Laporte (2010), Rego et al. (2011) and Arram et al. (2014). However, in transportation problems, customers usually have a demand, whereas the depot consists of a fleet of vehicles with limited and known capacity. This situation reflects the VRP (Dantzig and Ramser, 1959), which generalizes the Multiple Traveling Salesman Problem (m-TSP), i.e., the TSP with m vehicles (Bektas, 2006). A survey of the VRP literature as well as the most important exact solutions, classical and modern heuristics are presented by Cordeau et al. (2002), Eksioglu et al. (2009), Laporte (2009) and Potvin (2009). The Vehicle Routing Problem with Time Windows (VRPTW) is a generalization of the VRP involving the added complexity that every customer should be served within a given time window (Bräysy and Gendreau, 2005a; Bräysy and Gendreau, 2005b; El-Sherbeny, 2010).

Furthermore, the IRP is an extension of the VRP, which integrates routing decisions with inventory control (Moin and Salhi, 2007; Andresson et al., 2010; Coelho et al., 2013; Archetti and Speranza, 2016). The problem arises in environments where VMI policies are employed, while the supplier decides the delivery times, the quantities and the vehicle routes at the same time. The main objective is to minimize the total transportation and inventory holding costs. The Inventory Routing Problem with Hard

or Soft Time Windows (IRPTW/IRPSTW) is a generalization of the standard IRP involving the added complexity that every customer should be served within a given time window. Liu and Lee (2011) proposed a two-phase heuristic method for solving the IRPSTW. The first phase of the heuristic algorithm finds an initial solution based on a construction approach, while the second phase improves the initial solution by adopting a variable neighborhood tabu search algorithm. In addition, Zeng and Zhao (2010) represented the stochastic IRPSTW as a discrete time Markov decision process model and solved it by using dynamic programming approximations. Lappas et al. (2015a) presented a two-phase solution algorithm based on the Monte Carlo Simulation and the Genetic Algorithm to solve the IRPTW. The first phase is related to the planning phase of the IRPTW, in which delivery times and quantities are determined by implementing the well-known inventory policy (s,S) for inventory management using the Monte Carlo Simulation. In the second phase, the Genetic Algorithm is applied to combine the customers into the vehicle routes by solving a VRPTW for a specific time period during the planning horizon. Some applications in the context of IRPTW/IRPSTW were presented by Zhang et al. (2013), Li et al. (2015) and Zhang et al. (2015). The IRPTW is obviously NP-hard, being a generalization of the IRP, which reduces to the TSP when the planning horizon is equal to a single period (e.g., one day); there are no inventory holding costs; all the customers need to be served but not in specific time windows; there is a single vehicle and transportation capacity is infinite (Bertazzi and Speranza, 2013; Lappas et al., 2015b; Lappas et al., 2015c) (Fig. 1).

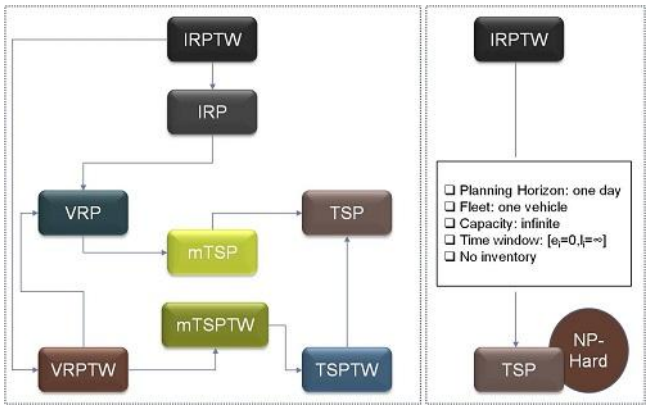


Fig. 1. NP-hard nature of the IRP

The PRP is also a core problem that has to be solved specifically in a VMI replenishment system and can be assumed to be the generalization of the IRP. The vendor monitors the inventory levels of the customers, while production, inventory, distribution and routing decisions have to be made simultaneously. For a comprehensive review of this literature through the year 2015, see Adulyasak (2015).

Several applications of the IRP have been found. The result of an analysis of the scientific literature led to the identification of six main paths of development in the overall field of the IRP: (1) maritime transportation (Ronen, 1993; Arga et al., 2013; Song and Furman, 2013; Hewitt et al., 2013; Arga et al., 2014; Papageorgiou et al.,

2014; Arga et al., 2015; Jiang and Grossmann, 2015; Hemmati et al., 2015; Arga et al., 2016a; Arga et al., 2016b; Hemmati et al., 2016), (2) industrial gas distribution (Bell et al., 1983; Goel et al., 2012; Ghiami et al., 2015; Shao et al., 2015; Singh et al., 2015; Goel et al., 2015; Andersson et al., 2016), (3) distribution of perishable goods (Federgruen and Zipkin, 1984; Federgruen et al., 1986; Le et al., 2013; Soysal et al., 2015; Mirzaei and Seifi, 2015; Soysal et al., 2016; Diabat et al., 2016), (4) fuel delivery (Popović et al., 2012), (5) medical waste collection (Nolz et al., 2014a; Nolz et al., 2014b) and medical drug distribution (Niakan and Rahimi, 2015), in addition to (6) distribution of agriculture products (Liao et al., 2013) and groceries (Mercer and Tao, 1996; Gaur and Fisher, 2004).

The IRP research can be divided into three main streams. In the first stream, exact algorithms have been proposed to solve the IRP. Some of the exact algorithms that have been published through the year 2013 and that can solve an IRP are summarized by Coelho et al. (2013) and Coelho and Laporte (2013). The second stream of research contains approximation approaches. Due to the inability of the exact algorithms to solve large-scale IRP instances, an impressive number of heuristics as well as meta-heuristics have been proposed. Constructive heuristics and improvement heuristics have been developed and presented by Abdelmaguid et al. (2009) for the IRP with backlogging. The proposed construction heuristic, called ETCH (Estimated Transportation Costs Heuristic), estimates a transportation cost value for each customer in each time period to facilitate a comparison between the transportation and the inventory holding and shortage costs. Due to the myopic nature of the ETCH and the fact that partial fulfillment of demand is not allowed, an improvement heuristic was proposed in order to overcome the above limitations. The improvement heuristic is based on the idea of exchanging customer delivery quantities between periods to allow transitions from a given solution to its neighborhood. More recently, Raa (2015) provided a multi-start two-phase heuristic solution method consisting of an insertion-based construction phase and an improvement phase for the Cyclic IRP, while Nambirajan et al. (2016) proposed a three-phase heuristic called CARE (Clustering, Allocation, Routing, Extended) for two-stage multi-product inventory routing problems with replenishments.

Furthermore, several local search meta-heuristics such as Tabu Search (TS) (Archetti et al., 2012; Li et al., 2014; Qin et al., 2014), Greedy Randomized Adaptive Search Procedure (GRASP) (Guemri et al., 2016), Iterated Local Search (ILS) (Vansteenwegen and Mateo, 2014; Santos et al., 2016), Variable Neighborhood Search (VNS) (Mjirda et al., 2012; Mjirda et al., 2014, Mjirda et al., 2016) and Adaptive Large Neighborhood Search (ALNS) (Coelho et al., 2012a; Aksen et al., 2014; Shirokikh and Zakharov, 2015) have been applied to the IRP. An alternative approach that combines simulation with heuristics has been presented by Juan et al. (2014), who described and used a “simheuristic” algorithm to solve the single-period stochastic IRP with stock-outs. Their approach combines the Monde Carlo Simulation with the multi-start randomized heuristic.

A number of population search meta-heuristics have been proposed for the solution of the IRP and its variants. Huang and Lin (2010) presented a modified ant colony optimization algorithm for multi-item IRPs with demand uncertainty. Tatsis et al. (2013) described the multiple suppliers, one retailer (many-to-one) IRP and proposed an ant-based optimization algorithm to solve the problem. In both papers, the main objective is to minimize the total transportation, inventory holding and backlogging costs. A hybrid heuristic method that integrates a Large Neighborhood Search (LNS) into Particle Swarm Optimization (PSO) presented by Liu et al. (2015) to solve the Periodic IRP. In addition, Yang et al. (2015) applied indicator-based evolutionary algorithms and swarm algorithms to find an approximation to the Pareto front of the IRP. Evolutionary optimization algorithms, such as GAs, have also been proposed to solve the IRP. This is particularly clear in the studies cited by Abdelmaguid and Dessouky (2006), Aziz and Moin (2007), Moin et al. (2011) Simić and Simić (2013), Shukla et al. (2013), Cho et al. (2013) and Park et al. (2016).

The third stream of research is associated with mathheuristics, consisting of decomposition approaches (Campbell and Savelsbergh, 2004), improvement heuristics (Coelho et al., 2012b; Bertazzi et al., 2013; Guerrero et al., 2013; Archetti et al., 2014; Bertazzi et al., 2015) and column generation-based approaches (Aghezzaf et al., 2006).

The research presented below represents an attempt to use local search and population search meta-heuristics to solve the IRP. The basic idea of the proposed approach is to combine a nature-inspired evolutionary optimization algorithm, such as the SA, and a biologically-inspired evolutionary optimization algorithm, that is, the GA, to handle the IRP. Therefore, a hybrid evolutionary optimization algorithm is proposed to solve the IRP. The SA is associated with the routing phase of the IRP, while the GA is related to the planning phase of the IRP. Both algorithms are dealt with in an iterative way.

The works most closely related to this theis are most likely those of Abdelmaguid and Dessouky (2006), Aziz and Moin (2007), Moin et al. (2011), Cho et al. (2013), and Park et al. (2016). Abdelmaguid and Dessouky (2006) introduced a genetic algorithm to solve the one-to-many type of the IRP with finite horizon. The objective function includes transportation costs as well as inventory holding and shortage costs on the end inventory positions. In particular, they designed a genetic representation in the form of a two-dimensional matrix based on the delivery schedule and addressed the vehicle routing part using the Clarke and Wright algorithm. In addition, a randomized version of a construction heuristic called ATCH (Approximate Transportation Costs Heuristic) was used to generate the initial random population, while suitable crossover and mutation operators were designed for the improvement phase of the genetic algorithm. In the studies by Aziz and Moin (2007) and Moin et al. (2011), the many-to-one type of IRP with finite horizon is addressed. Both transportation and inventory costs are considered, while a hybrid genetic algorithm combining a genetic algorithm (planning phase) and a simple 2-opt procedure (routing phase) is presented. Cho et al.

(2013) proposed an adaptive genetic algorithm for the time dependent inventory routing problem considering the one-to-many network topology. This paper takes into account the effect of dynamic traffic conditions in an urban context, while the objective function consists of the transportation, inventory holding and shortage costs at the end of the period inventory positions. More recently, Park et al. (2016) presented a genetic algorithm for the inventory routing problem with lost sales under a VMI strategy in a two-echelon supply chain comprised of a single manufacturer and multiple retailers (one-to-many network topology). The objective function consists of the transportation costs, the inventory holding cost of the manufacturer, the inventory holding costs of the retailers and the costs associated with lost sales.

The proposed hybrid evolutionary optimization algorithm shows significant differences:

1. Most of the previous research has considered a one-to-many type of IRP in which the objective function includes shortage costs at the end of the period inventory positions or costs related to lost sales. In this thesis, stock-outs or lost sales are not allowed, and therefore, no shortage costs or costs related to lost sales are included in the objective function.
2. Some of the previously reported research (e.g., Abdelmaguid and Dessouky, 2006; Aziz and Moin, 2007; Moin et al., 2011) has focused only on the planning phase of the IRP, while the routing phase has been addressed by simple heuristics such as the Clarke and Wright algorithm and the 2-opt algorithm. In this thesis, the routing phase of the IRP is addressed by the Simulated Annealing algorithm, a nature-inspired optimization algorithm (local search meta-heuristic) simultaneously improving the solution approach in the context of the vehicle routing problem.
3. In the VRP literature, there exists a classical set of well-known benchmarks commonly used to test new VRP algorithms. However, this is not the case for the IRP. As a result, to provide complete information about the set of benchmarks that are employed so that other researchers can use them, new datasets have been developed by generalizing the well-known dataset P of Augerat et al. (1998). These datasets are divided into two categories: datasets consisting of low inventory holding costs and datasets including high inventory holding costs. Different problem sizes, based on the total number of customers, were designed in each category to evaluate the performance of the proposed solution approach in the context of the one-to-many type of IRP: 15, 20, 22, 39, 44, 50, 54, 59, 64, 69, 75 and 100 customers (first category) and 15, 20, 22, 39, 44, 50 customers (second category).

As far as the IRPTW is concerned, new benchmark instances have been developed by generalizing the well-known datasets of Solomon (1987)[1]. Consequently, the efficiency and the effectiveness of the proposed two-phase solution algorithm cannot

[1] http://web.cba.neu.edu/~msolomon/problems.htm

be compared to other published IRPTW studies using benchmark instances previously introduced. This is due to the differentiated manner in which the proposed algorithm operates based on the assumptions presented in Chapter 4. The basic notion is to formulate a mathematical problem and present a two-phase solution algorithm to prefigure a road-map for future work. Testing instances are established to investigate algorithmic performance, and the computational results are then reported. Finally, this study provides various graphical presentation formats to highlight the insights that are gained. In particular, the analytical results and graphic presentations help to simplify complicated issues and convey meaningful insights into the problem.

Chapter 3

The Inventory Routing Problem

3.1. Problem Description and Mathematical Formulation

This section presents a modeling framework for formulating the IRP. Let $G = (V, E)$ be a complete undirected graph where $V = \{0, \dots, n\}$ is the set of vertices and $E = \{(i, j): i, j \in V, j > i\}$ is the set of edges. Vertices $1, \dots, n$ correspond to the customers, whereas vertex 0 corresponds to the depot. The model presented here deals with the repeated distribution of a single product from a single supplier to a set of geographically dispersed customers $C = V\backslash\{0\} = \{1, \dots, n\}$ over a given time horizon of length H. The set of time horizons is denoted by $T = \{1, \dots, H\}$. Each customer $i \in C$ faces a different demand d_i^t per time period $t \in T$, maintains his own inventory up to capacity U_i, and incurs an inventory holding cost of h_i per period per unit. It is assumed that the depot has a sufficient supply of items that can cover all customers' demands throughout the planning horizon, that is, $U_0 = +\infty$.

A nonnegative cost, c_{ij} is associated with each edge $(i, j) \in E$ and represents the travel cost spent to go from vertex i to vertex j $\forall i, j \in V$. Generally, the usage of the loop edge, (i, i) is not allowed, and this is imposed by defining $c_{ii} = +\infty$ for all $i \in V$. In addition, the cost matrix satisfies the triangle inequality: $c_{ik} + c_{kj} \geq c_{ij}$. In other words, it is not convenient to deviate from the direct link between two vertices. Since G is a complete undirected graph, the cost matrix $[c_{ij}]$ is symmetric, and as a result, $c_{ij} = c_{ji}$ $\forall i, j \in V$. Vertices are associated with points of the plane having the given coordinates $(x_i, y_i) \forall i \in V$, and the cost c_{ij} for each edge $(i, j) \in E$ is defined as the Euclidean distance between the two vertices $i, j \in V$. Therefore, $c_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

An unlimited fleet of identical vehicles with capacity Q is available for the distribution of the product. The fleet of vehicles is denoted by the set $K = \{1,2, \dots\}$. However, to model the problem, an upper bound on the number of vehicles needed to distribute the products should be defined. A trivial upper bound on the maximum fleet size needed is $|K| = |C| = n$. Furthermore, the formulation uses the following decision variables:

- w_{ik}^t: the amount of delivery to customer $i \in C$ in period $t \in T$ by vehicle $k \in K$.
- x_{ijk}^t: the number of times the edge $(i, j) \in E$ is traversed by vehicle $k \in K$ in period $t \in T$.

- y_{ik}^t: a binary variable that is used to assign customers to vehicles, with value 1 indicating that customer $i \in C$ will be visited by vehicle $k \in K$ in period $t \in T$, and 0 otherwise.
- I_i^t: a nonnegative variable indicating the inventory level at customer $i \in C$ at the end of period $t \in T$. It should be mentioned that at the beginning of the planning horizon, each customer $i \in C$ has an initial inventory level of $I_i^0 = 0, \forall i \in C$ of product.

Moreover, stock-outs are not allowed at the customers, while the quantities delivered by each vehicle in each route cannot exceed the vehicle capacity. As far as the replenishment policy is concerned, a Maximum Level (ML) policy is applied. Therefore, any customer has defined a maximum inventory level. Every time a customer is served, the delivered quantity is such that the inventory level at the customer is not greater than the maximum level. After defining the necessary parameters and decision variables, the IRP can be formulated as a mixed integer linear programming as shown below:

$$\text{Min}\left(\sum_{k\in K}\sum_{t\in T}\sum_{i\in V}\sum_{j\in V, j>i} c_{ij}x_{ijk}^t + \sum_{t\in T}\sum_{i\in C} h_i I_i^t\right) \quad (1)$$

Subject to:

$$I_i^t = I_i^{t-1} + \sum_{k\in K} w_{ik}^t - d_i^t, \forall i \in C, \forall t \in T \quad (2)$$

$$I_i^t \geq 0, \forall i \in C, \forall t \in T \quad (3)$$

$$I_i^t \leq U_i, \forall i \in C, \forall t \in T \quad (4)$$

$$\sum_{i\in C} w_{ik}^t \leq Q y_{ik}^t, \forall k \in K, \forall t \in T \quad (5)$$

$$w_{ik}^t \leq U_i y_{ik}^t, \forall i \in C, \forall k \in K, \forall t \in T \quad (6)$$

$$\sum_{k\in K} y_{ik}^t \leq 1, \forall i \in C, \forall t \in T \quad (7)$$

$$\sum_{j\in V, j>i} x_{ijk}^t + \sum_{j\in V, j<i} x_{jik}^t = 2y_{ik}^t, \forall i \in C, \forall k \in K, \forall t \in T \quad (8)$$

$$\sum_{i\in S}\sum_{j\in S, j>i} x_{ijk}^t \leq \sum_{i\in S} y_{ik}^t - y_{sk}^t, \forall S \subseteq C, \forall s \in S, \forall k \in K, \forall t \in T \quad (9)$$

$$w_{ik}^t \geq 0, \forall i \in C, \forall k \in K, \forall t \in T \quad (10)$$

$$x_{ijk}^t \in \{0,1\}, \forall i \in C, \forall j \in C, j > i, \forall k \in K, \forall t \in T \quad (11)$$

$$x_{0jk}^t \in \{0,1,2\}, \forall j \in C, \forall k \in K, \forall t \in T \quad (12)$$

$$y_{ik}^{t} \in \{0,1\}, \forall i \in V, \forall k \in K, \forall t \in T \quad (13)$$

The total cost comprises the transportation costs and the inventory holding costs of the customers as depicted in the objective function (1). Constraints (2) are the inventory balance equations for the customers. Constraints (3) guarantee that no stock-out occurs at the customers, and constraints (4) limit the inventory level of the customers to the corresponding maximum inventory level (ML policy). Constraints (5) ensure that the vehicle capacities are not exceeded in any period $t \in T$ during the planning horizon. Constraints (6) impose the condition that if any quantity is delivered to the customer $i \in C$ in period $t \in T$, the customer i is visited in period t. In addition, a customer can be visited exactly once in each period $t \in T$ (7). Constraints (8) and (9) are the routing constraints. Namely, they guarantee that a feasible route is determined to visit all customers served in period $t \in T$. Finally, constraints (10), (11), (12) and (13) are the domain constraints.

3.2. Solution Approach for the IRP

Due to the NP-hard nature of the IRP, a hybrid evolutionary optimization algorithm based on two well-known meta-heuristics (Genetic Algorithm, Simulated Annealing Algorithm) is proposed to handle the problem. Since the IRP can be described as the combination of the Inventory Control and the Vehicle Routing Problems, the meta-heuristics are used as follows: The Genetic Algorithm is related to the planning phase of the IRP (inventory control problem) determining delivery times and quantities, while the Simulated Annealing Algorithm is associated with the routing phase of the IRP (vehicle routing problem) determining routes. Both algorithms are dealt with in an iterative way to define the re-optimization phase. Hence, a repetitive procedure is applied to obtain a near-optimal feasible solution.

3.2.1. Planning Phase – A Genetic Algorithm Approach

Genetic Algorithms (GAs) have been developed by John Holland and his collaborators at the University of Michigan in the 1970s (Holland, 1975). They are based on the principles of biological evolution and the natural selection process of the survival of the fittest. This process actually reflects an optimization process based on an initial, randomly generated, population of solutions (population-based meta-heuristic). A solution is referred to as an *individual*, while its data structure representation corresponds to the *chromosome* or *genotype*. A chromosome consists of *genes* that represent the decision variables within a solution. One iteration of creating a new population through the optimization algorithm is called a *generation*.

The population is maintained and evolved from generation to generation using genetic operators such as *evaluation*, *reproduction (selection)*, *recombination (crossover)* and *mutation*. The *fitness* of each individual is associated with the evaluation function or the objective function, while the *phenotype* represents how an individual operates during the fitness assessment.

Furthermore, a selection process allows *parent* solutions with high fitness to be selected from the current population. Then, crossover and mutation operators are applied to generate children (*offspring*). In particular, the crossover operator intends to inherit some characteristics (genes) of the two parents to generate the offspring, while the mutation operator represents a slight change to a single individual. The offspring compete with the parents for their place in the next generation (survival of the fittest), thus constructing the next population. In the following subsections, a detailed description of the developed genetic approach regarding the IRP is given.

3.2.1.1. Chromosome Representation

A small sample problem of a distribution system that comprises a single supplier and six customers can be considered to illustrate the proposed chromosome representation (Fig. 2).

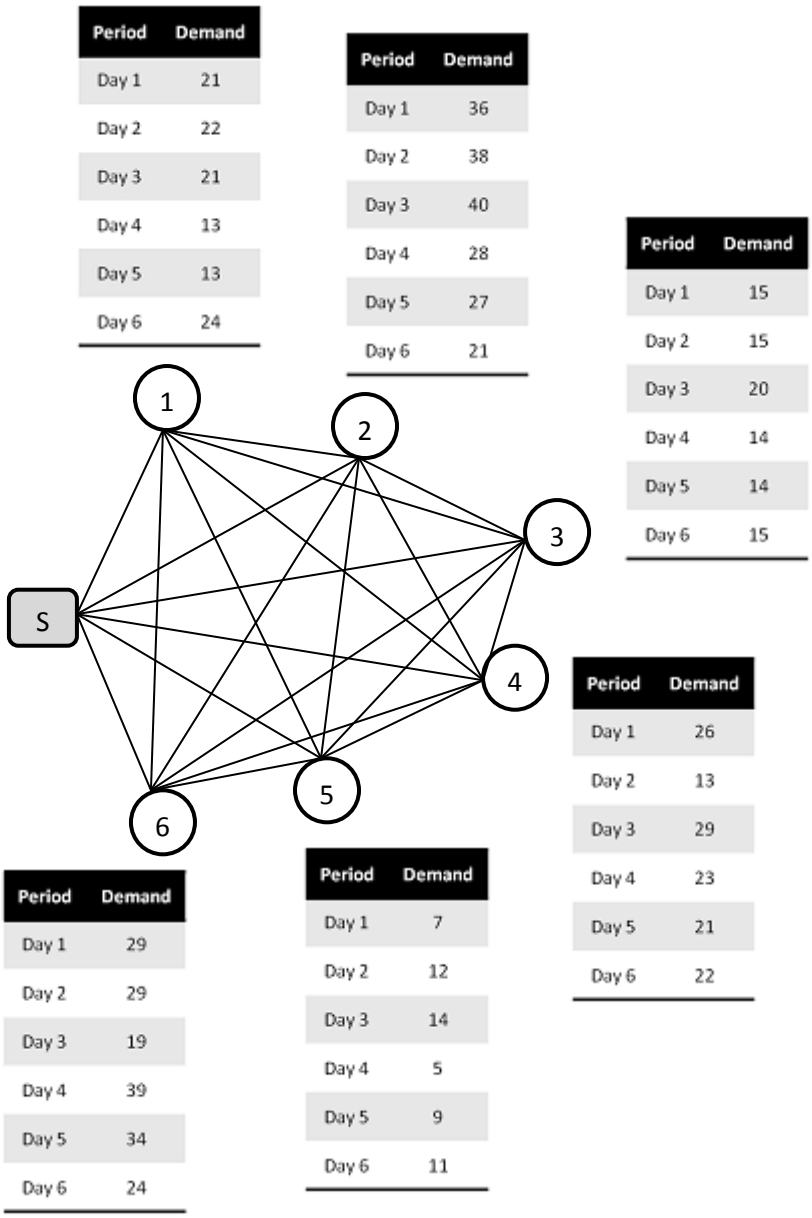


Fig. 2. Illustrative example

The planning horizon is equal to six days. At the beginning of the planning horizon, all customers have zero inventory levels, whereas each customer has a daily demand. Stock-outs are not allowed, while inventory holding costs exist only at the demand points. Each customer has a sufficient maximum inventory level to satisfy his storage needs during the planning horizon. Furthermore, it is assumed that the supplier has a sufficient supply of products that can cover all customers' demands throughout the planning horizon. Table 1 provides information about the maximum inventory level as well as the inventory holding cost of each customer.

Table 1

Inventory information for the illustrative example

Customer	Inventory Holding Cost (per unit per period)	Maximum Inventory Level
1	0.4649	115
2	0.3723	190
3	0.3545	95
4	0.4054	135
5	0.4908	60
6	0.1219	175

A chromosome can be represented by a two-dimensional matrix with six rows and six columns (Fig. 3).

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	43	0	21	13	13	24
Customer 2	74	0	40	55	0	21
Customer 3	15	15	34	0	14	15
Customer 4	26	42	0	23	21	22
Customer 5	7	12	14	5	9	11
Customer 6	29	87	0	0	34	24

Fig. 3. Chromosome representation

The rows and the columns of the matrix correspond to the customers and the time periods of the planning horizon, respectively. Each cell of the matrix represents the total amount of product that should be delivered to a specific customer in a specific time period. For example, the total amount of product that should be delivered to customer 2 in day 3 is equal to 40. Since stock-outs are not allowed, it should be observed that each delivery quantity satisfies the current demand of the customer. If a delivery to a customer does not take place in a specific time period, the period's demand is satisfied through the available inventory from a previous delivery. For instance, the delivery quantities of period 1 for Customer 2 are enough to satisfy the demands of Period 1 and 2, respectively (43 = 21 + 22). Therefore, for each customer (row of a matrix), the sum of delivery quantities is equal to the sum of customer demand during the planning horizon.

3.2.1.2. Generation of Initial Population

Based on a pre-defined population size, a random procedure is followed to generate the initial population. To begin with, each individual in the population is represented

by a randomly generated binary matrix (Fig. 4). Each cell contains a 1/0 value indicating whether a customer is visited in a specific time period.

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	1	0	1	1	1	1
Customer 2	1	0	1	1	0	1
Customer 3	1	1	1	0	1	1
Customer 4	1	1	0	1	1	1
Customer 5	1	1	1	1	1	1
Customer 6	1	1	0	0	1	1

Fig. 4. Binary matrix representation

Since at the beginning of the planning horizon all customers have zero initial inventory levels and stock-outs are not allowed, the first column of the binary matrix contains only 1-values. The remaining columns of the binary matrix are randomly generated. Below, an algorithm (Algorithm 1) is presented that generates a binary matrix.

Algorithm 1. *Generate a binary matrix*

Inputs: NC (number of customers), NP (number of periods)

$tempBM1 \leftarrow ones(NC, 1), tempBM2 \leftarrow randi([0,1], NC, NP-1)$

$BinaryMatrix \leftarrow [tempBM1, tempBM2]$

Output: $BinaryMatrix$

Analytically, $ones$ creates an NC-by-1 array of ones, while $randi$ creates an NC-by-$(NP-1)$ array of 1/0 values. Afterward, the algorithm combines the two arrays into one array to create the binary matrix that corresponds to an individual of the population. Given a population size, $PopSize$, this procedure can be repeated to create the initial population (Algorithm 2).

Algorithm 2. *Generate a population of binary matrices*

Inputs: $NC, NP, PopSize$

$\boldsymbol{for}\ i = 1{:}\ PopSize\ \boldsymbol{do}$

$\boldsymbol{Call}\ Algorithm\ 1$

$\boldsymbol{end-for}$

Output: $PopBM$ (population of binary matrices)

According to a binary matrix, a real-value matrix that consists of delivery quantities in each time period of the planning horizon can be easily produced (Algorithm 3). This two-dimensional matrix reflects the chromosome representation shown in section 3.2.1.1 (Fig. 5).

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	43	0	21	13	13	24
Customer 2	74	0	40	55	0	21
Customer 3	15	15	34	0	14	15
Customer 4	26	42	0	23	21	22
Customer 5	7	12	14	5	9	11
Customer 6	29	87	0	0	34	24

Fig. 5. Delivery quantities matrix (chromosome representation)

Algorithm 3. *Produce chromosome representations of population's individuals*

Inputs: DM (demand matrix), $PopBM$, $PopSize$

$\boldsymbol{for}\ i = 1{:}\ PopSize\ \boldsymbol{do}$

$Population\{i\} \leftarrow convertBM(DM, PopBM)$

$\boldsymbol{end-for}$

Output: $Population$

Given the customers' demands during the planning horizon and their binary matrix representations, Algorithm 3 produces real-value matrices that reflect the initial population with respect to the assumption that stock-outs are not allowed. In particular, after each iteration, $convertBM$ creates a delivery quantity matrix according to the demand matrix of each customer and the relative binary matrix. As a result, after each iteration, an individual is added to the population. Moreover, based on a delivery quantity matrix, inventory levels and inventory holding costs of each customer can be easily determined, as shown in figures 6 and 7, respectively.

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	22	0	0	0	0	0
Customer 2	38	0	0	27	0	0
Customer 3	0	0	14	0	0	0
Customer 4	0	29	0	0	0	0
Customer 5	0	0	0	0	0	0
Customer 6	0	58	39	0	0	0

Fig. 6. Inventory level matrix

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	10.2281	0	0	0	0	0
Customer 2	14.1470	0	0	10.0518	0	0
Customer 3	0	0	4.9631	0	0	0
Customer 4	0	11.7567	0	0	0	0
Customer 5	0	0	0	0	0	0
Customer 6	0	7.0716	4.7551	0	0	0

Fig. 7. Inventory holding cost matrix

3.2.1.3. Fitness Evaluation and Selection

An important issue is the choice of an appropriate fitness function that determines the selection criterion in the IRP. The fitness quantifies the optimality of a solution (i.e., a chromosome) in the proposed hybrid evolutionary algorithm so that a particular chromosome may be ranked against all the other chromosomes. Therefore, optimal chromosomes are allowed to breed and mix their genes by any of several techniques, producing a new generation that will be even better. For the IRP, it is assumed that candidate solutions with lower total costs (inventory holding costs plus transportation costs) imply better solutions. Since the IRP is a minimization problem, the fitness for each chromosome is defined as follows:

$$fitness = \frac{1}{\sum_{k\in K}\sum_{t\in T}\sum_{i\in V}\sum_{j\in V, j>i} c_{ij}x_{ijk}^{t} + \sum_{t\in T}\sum_{i\in C} h_i I_i^t} \quad (14)$$

Therefore, each individual has a probability of being selected that is proportional to its fitness. The higher the individual's fitness is, the more likely it is to be selected. In this context, the roulette-wheel selection approach is adopted as the selection process (Algorithm 4).

Algorithm 4. *Roulette-wheel selection*

Inputs: $fitness(1), fitness(2), \dots, fitness(PopSize)$

$f_{sum} \leftarrow \sum_{i=1}^{i=PopSize} fitness(i)$

$Generate\ a\ uniformly\ distributed\ random\ number\ rN \in [0, f_{sum}]$

$FV \leftarrow fitness(1)$

$iter \leftarrow 1$

$\boldsymbol{while}\ FV < rN\ \boldsymbol{do}$

$iter \leftarrow iter + 1$

$FV \leftarrow FV + fitness(iter)$

$\boldsymbol{end - while}$

$Parent \leftarrow iter$

Output: $Parent$

Algorithm 4 shows how to select a parent from a population of $PopSize$ individuals. To keep the population size constant across generations, suitable pairs of mates are picked. The goal is to select every time two parents to produce two offspring. This process is repeated until the population of offspring is the same as the population of parents.

3.2.1.4. Crossover Operator

Since two parents are selected, a crossover operator can be applied. For the reported chromosome representation, a single-point crossover operator as well as a double-point crossover operator has been designed and can be used randomly to produce two offspring. The two-dimensional matrix structure can be broken horizontally considering that delivery quantities for a selected set of customers will be exchanged between two parent solutions. Hence, the crossover point is relevant to a specific row of the two-dimensional matrix.

3.2.1.4.1. Single-Point Crossover Operator

Based on the two-dimensional matrix structure, the single-point crossover indicates that one crossover position (a row of the matrix) is selected uniformly at random and the rows are exchanged between the individuals about this point. Then, two new offspring are produced. Consider the following example (Fig. 8).

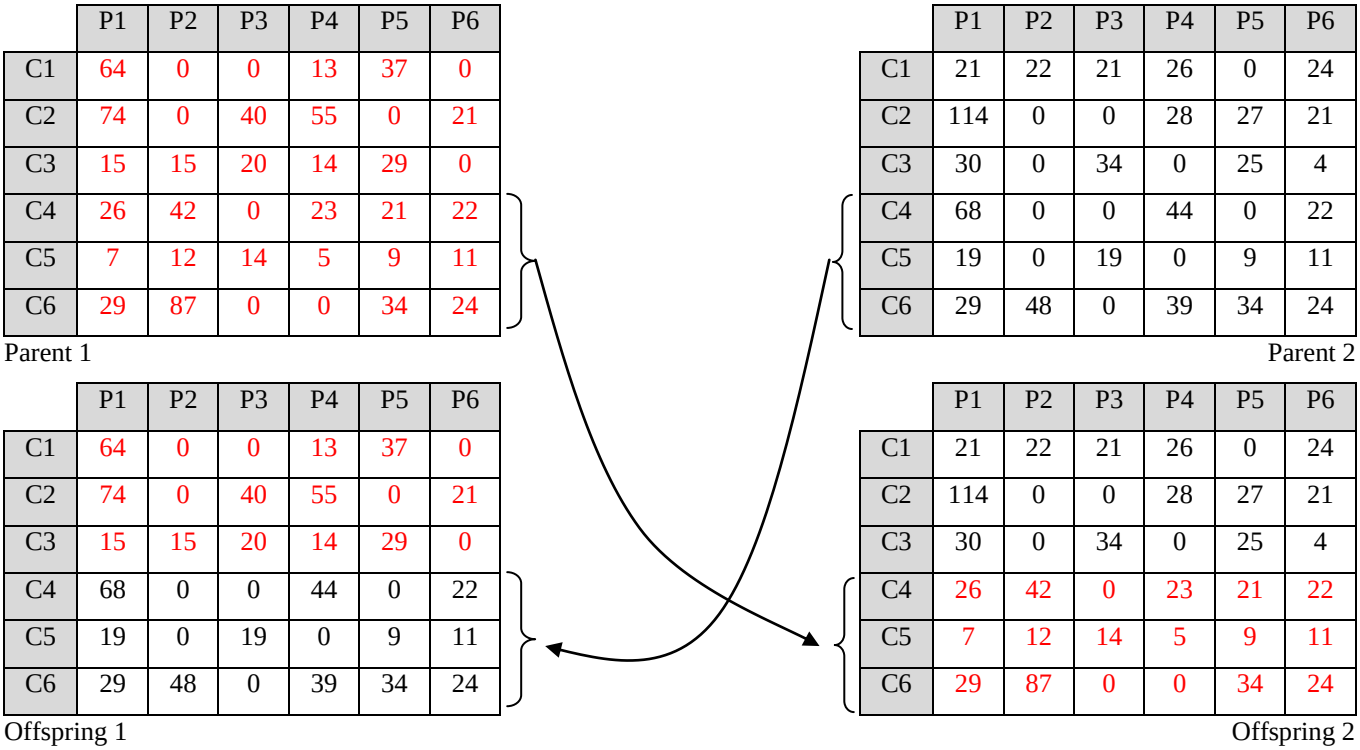


Fig. 8. Single-point crossover operator

In the above case, the third row of the matrix is considered as the crossover point. As a result, Parent 1 and Parent 2 exchange rows 4, 5 and 6 with each other, thus producing two offspring. The algorithm that shows the functionality of the single-point crossover operator is presented below (Algorithm 5). Assuming that two parents are selected, $parent(1)$ and $parent(2)$ from a given population, $population$, a crossover point is randomly generated from $[2, NC-1]$, where NC is the given number of customers. Since the crossover point is known, two offspring are produced. The first offspring, O_1, as well as the second one, O_2, maintain the first C$rossPoint$ rows of P_1 and P_2, respectively. In addition, the remaining rows of P_1, $CrossPoint+1,\dots,NC$, are copied to O_2, while the remaining rows of P_2 are copied to O_1. Furthermore, to guarantee the continuity of the process, the relative binary matrices for O_1 and O_2 are produced, called O_1^{BM} and O_2^{BM}, respectively.

Algorithm 5. *Single-point crossover*

Inputs: $PopBM, Population, Parent(1), Parent(2), NC$

$P_1 \leftarrow Parent(1), P_2 \leftarrow Parent(2), CrossPoint \leftarrow random\ integer\ from\ [2, NC-1]$

$O_1 \leftarrow Population\{P_1\}, O_2 \leftarrow Population\{P_2\}, O_1^{BM} \leftarrow PopBM\{P_1\}, O_2^{BM} \leftarrow PopBM\{P_2\}$

$O_1(CrossPoint+1:end,:) \leftarrow Population\{P_2\}(CrossPoint+1:end,:)$

$O_2(CrossPoint+1:end,:) \leftarrow Population\{P_1\}(CrossPoint+1:end,:)$

$O_1^{BM}(CrossPoint+1:end,:) \leftarrow PopBM\{P_2\}(CrossPoint+1:end,:)$

$O_2^{BM}(CrossPoint+1:end,:) \leftarrow PopBM\{P_1\}(CrossPoint+1:end,:)$

Outputs: $O_1, O_2, O_1^{BM}, O_2^{BM}$

3.2.1.4.2. Double Crossover Operator

In the double-point crossover operator, two crossover positions are selected uniformly at random and the rows are exchanged between the individuals between these points. Then, two new offspring are produced. Consider the following example.

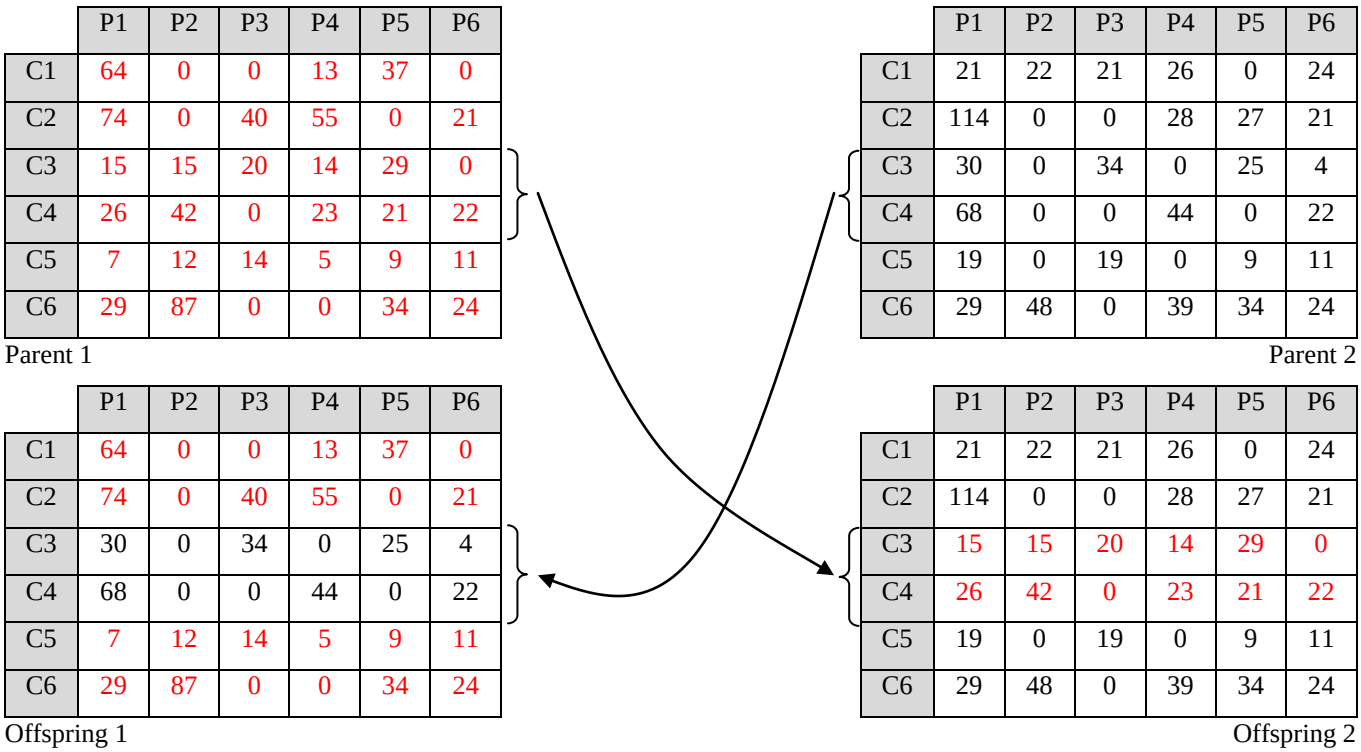


Fig. 9. Double-point crossover operator

This example depicts two crossover positions, row 2 and row 5. Therefore, rows 3 and 4 of the Parent 1, P_1, are copied to the Offspring 2, O_2, while rows 3 and 4 of the Parent 2, P_2, are copied to the Offspring 1, O_1. Analogous to the simple-point crossover operator, an algorithm of the double-point crossover operator is presented below (Algorithm 6).

Algorithm 6. *Double-point crossover*

Inputs: $PopBM, Population, Parent(1), Parent(2), NC$

$P_1 \leftarrow Parent(1), P_2 \leftarrow Parent(2), CrossPoint_1 \leftarrow random\ integer\ from\ [2, NC-1]$

$CrossPoint_2 \leftarrow random\ integer\ from\ [2, NC-1]$

$sort(CrossPoint_1, CrossPoint_2)\ //CrossPoint_1 \neq CrossPoint_2$

$a \leftarrow CrossPoint_1$,b$\leftarrow CrossPoint_2$

$O_1 \leftarrow Population\{P_1\}, O_2 \leftarrow Population\{P_2\}, O_1^{BM} \leftarrow PopBM\{P_1\}, O_2^{BM} \leftarrow PopBM\{P_2\}$

$O_1(a+1:b-1,:) \leftarrow Population\{P_2\}(a+1:b-1,:)$

$O_2(a+1:b-1,:) \leftarrow Population\{P_1\}(a+1:b-1,:)$

$O_1^{BM}(a+1:b-1,:) \leftarrow PopBM\{P_2\}(a+1:b-1,:)$

$O_2^{BM}(a+1:b-1,:) \leftarrow PopBM\{P_1\}(a+1:b-1,:)$

Outputs: $O_1, O_2, O_1^{BM}, O_2^{BM}$

3.2.1.5. Mutation Operator

After the crossover, an individual is subjected to mutation. In particular, the mutation prevents the algorithm from being trapped in a local minimum. Therefore, through the crossover, a current solution is exploited to find better ones, whereas the mutation is supposed to help to explore the whole search space. In the context of the proposed solution approach, the mutation operator presented by Abdelmaguid and Dessouky (2006), called the backward delivery exchange, is adopted. The backward delivery exchange process is chosen due to the restriction that stock-outs are not allowed. Accordingly, part of a customer's delivery amount can be transferred only to a preceding period. The following figure (Fig. 10) illustrates an example of using the backward delivery exchange operator.

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	43	0	21	13	13	24
Customer 2	74	0	40	55	0	21
Customer 3	15	15	34	0	14	15
Customer 4	26	42	0	23	21	22
Customer 5	7	~~12~~ 22	~~14~~ 4	5	9	11
Customer 6	29	87	0	0	34	24

Fig. 10. Backward delivery exchange operator

The example shows that the delivery quantity for Customer 5 scheduled in period 3 is reduced by 10 units, and this amount is transferred to period 2.

Below, the algorithm of the backward delivery exchange operator is presented (Algorithm 7). An important parameter in the mutation process is the mutation

probability, $MutationRate$, which decides how often parts of a chromosome will be mutated. Since a mutation takes place, a random integer, $RandNC$, is generated in the interval from $[1, NC]$, where NC indicates the number of the customers. Then, for $RandNC$ times, the following process is repeated. A period, $RandPer$, is selected randomly. If no deliveries are scheduled for this period, $RandPer$ is re-generated randomly. Afterwards, $find$ returns the customers, $ListCust$, that are scheduled to be visited in the $RandPer$ time period of the planning horizon. A customer, $RandMutCust$, from $ListCust$ is randomly selected and his scheduled delivery amount, $DelX$, is saved. Next, the amount that could be transferred to a preceding period, $BcDelX$, is randomly selected in the interval from $[1, DelX]$. From previous periods where a customer has scheduled deliveries, the nearest period, $PrecPer$, is selected to transfer $BcDelx$ units of product. Subsequently, the scheduled delivery quantity in period $RandPer$ is reduced by $BcDelX$ units, and this amount is transferred to period $PrecPer$.

Algorithm 7. *Backward delivery exchange operator*

Input: O (child solution)

$\boldsymbol{if}\ (rand \leftarrow U(0,1)) < MutationRate\ \boldsymbol{then}$

$RandNC \leftarrow random\ integer\ from\ [1, NC]$

$\boldsymbol{for}\ mutInd = 1{:}\ RandNC\ \boldsymbol{do}$

$RandPer \leftarrow random\ integer\ from\ [2, NP]$

$\boldsymbol{while}\ sum(O(:, RandPer)) = 0\ \boldsymbol{do}$

$RandPer \leftarrow random\ integer\ from\ [2, NP]$

$\boldsymbol{end - while}$

$ListCust \leftarrow find(O(:, RandPer)), n \leftarrow length(ListCust)$

$RandMutCust \leftarrow random\ integer\ from\ [1, n]$

$DelX \leftarrow O(ListCust(RandMutCust), RandPer)$

$BcDelX \leftarrow random\ integer\ from\ [1, DelX]$

$PrecPer \leftarrow findPrecedingPeriod(RandPer)$

$tempDel \leftarrow O(ListCust(RandMutCust), PrecPer)$

$O(ListCust(RandMutCust), PrecPer) \leftarrow tempDel + BcDelX$

$O(ListCust(RandMutCust), RandPer) \leftarrow DelX - BcDelX$

$\boldsymbol{end - for}$

$\boldsymbol{end - if}$

Output: O (child solution after mutation)

3.2.2. Routing Phase – A Simulated Annealing Algorithm Approach

Since the Genetic Algorithm focuses only on the planning phase by determining the delivery times and quantities, the vehicle routes should be constructed. The routing

phase is related to the usage of a Simulated Annealing Algorithm for solving a vehicle routing problem for each time period of the planning horizon where delivery quantities have been scheduled. The Simulated Annealing Algorithm is a nature-inspired optimization algorithm introduced by Kirkpatrick et al. (1983). Contrary to Genetic Algorithms, it is a single-individual stochastic algorithm, as it does not involve a population of candidate solutions. The algorithm mimics the annealing process of heating and cooling a material in order to re-crystallize it (Talbi, 2009). In particular, the annealing process starts with an initial system state at a very high temperature, which is slowly decreased to obtain a strong crystalline structure. The strength of the structure depends on the rate of decrease, which is subjected to a cooling process until it converges to an equilibrium state (steady frozen state). However, to reach an equilibrium state at each temperature, a number of sufficient transitions must be applied.

Similarly, the Simulated Annealing algorithm (Algorithm 8) consists of two cycles, the external and the internal cycle. The algorithm begins with an initial feasible solution, x_0, and a high temperature T_{max} and proceeds in $EXTcyc$ iterations (external cycle). Then, the algorithm proceeds in $INTcyc$ iterations (internal cycle). Throughout the internal cycle, the temperature is constantly trying to converge to an equilibrium state at the end of $INTcyc$ iterations. At each iteration of the internal cycle, a neighboring solution, x, is generated by perturbing the current solution. A cost function, $CostFunction()$, exists to measure the quality of each solution. If the cost of the neighboring solution, $CostFunction(x)$, is less than the cost of the current solution, $CostFunction(x_0)$, it is accepted. Otherwise, it is accepted with probability $e^{-\frac{\Delta E}{T}}$, where T is a control parameter (temperature) and ΔE represents the difference in the objective value between the current solution and the generated neighboring solution. The control parameter T is decreased gradually through the external cycle. The temperature is updated using a geometric schedule that corresponds to the formula $T \leftarrow alpha \times T$, where $alpha \in [0,1]$. Therefore, as the algorithm progresses, the probability that a non-improving generated neighboring solution is accepted decreases. The set of parameters related to the high value of control parameter (*temperature*), T_{max}, the rate of decrease (*cooling rate*), $alpha$, and the stopping condition of the internal ($INTcyc$ iterations) as well as external cycle ($EXTcyc$ iterations) of the algorithm is called the *annealing (cooling) schedule*.

Algorithm 8. *Simulated annealing algorithm*

Inputs: $T_{max}, EXTcyc, INTcyc, alpha$

$x_0 \leftarrow GenerateInitialSolution()$ // *Algorithm* 9

$T \leftarrow T_{max}, i \leftarrow 1$

while $i \leq EXTcyc$ ***do***

$j \leftarrow 1$

while $j \leq INTcyc$ ***do***

$x \leftarrow CreateNeighboringSolution(x_0)$ // *Algorithm* 10

if $CostFunction(x) < CostFunction(x_0)$ ***then***

$x_0 \leftarrow x$

else

$\Delta E \leftarrow CostFunction(x_0) - CostFunction(x), randN \leftarrow U(0,1)$

if $randN < e^{-\frac{\Delta E}{T}}$ ***then***

$x_0 \leftarrow x$

$\boldsymbol{end - if}$

$\boldsymbol{end - if}$

$j \leftarrow j + 1$

$\boldsymbol{end - while}$

$T \leftarrow alpha * T, i \leftarrow i + 1$

$\boldsymbol{end - while}$

Output: *best solution found*

In terms of the optimization process, the annealing schedule controls the transition from the exploration to the exploitation. Particularly, at the beginning of the algorithm, the temperature has a high value, which is decreased until a final temperature is reached. This final temperature is typically close to zero. As a consequence, at the beginning of the algorithm, the exploration is high and the exploitation is low, while at the end of the algorithm, the exploitation is high and the exploration is low. The main objective is to obtain a balance between exploration and exploitation to sufficiently explore the search space and simultaneously exploit good solutions.

3.2.2.1. Solution Representation

Assume an individual in a population obtained by the planning phase described in section 3.2.1. The Simulated Annealing algorithm should be applied to each time period of the planning horizon where scheduled delivery quantities exist (Fig. 11).

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Customer 1	64	0	0	26	0	24
Customer 2	36	38	68	0	44	4
Customer 3	30	0	20	43	0	0
Customer 4	39	0	52	0	43	0
Customer 5	19	0	39	0	0	0
Customer 6	58	0	116	0	0	0
Customer 7	23	0	17	8	23	0
Customer 8	110	0	0	25	0	0
Customer 9	30	0	0	0	17	0
Customer 10	10	13	0	13	4	0
Customer 11	19	21	0	0	0	0
Customer 12	34	0	52	0	0	0
Customer 13	14	0	14	0	16	0
Customer 14	51	0	0	69	0	0
Customer 15	38	0	0	7	18	0
	VRP	VRP	VRP	VRP	VRP	VRP

Fig. 11. Solving a VRP problem at each time period of the planning horizon

Path representation is the most natural way of representing the routes of a VRP. Since a VRP consists of one or more routes, the length of each path is variable. On account of this, a dynamic variable, x, can be used to represent the solution of the VRP. x contains all the routes of a specific time period of the planning horizon. For instance, in the first time period $x = \{x.R_1, x.R_2, x.R_3\}$, where (a) $x.R_1 = [0,2,13,,8,3,10,0]$ is the first route, (b) $x.R_2 = [0,4,11,15,12,1,0]$ is the second route and (c) $x.R_3 = [0,6,7,9,14,5,0]$ is the third route. The zero value in each row vector represents the supplier, while the other numbers represent the customers.

3.2.2.2. Initial Solution

In order to generate an initial solution to start solving a VRP with Simulated Annealing algorithm, a random approach is followed, $GenerateInitialSolution()$. The approach iterates over a pre-defined list of customers that will be visited in a specific time period according to the planning phase. The algorithm (Algorithm 9) proceeds as follows. If there are n customers in the pre-defined list, a customer is selected randomly to start creating a route. Each route corresponds to a specific vehicle of a fleet with capacity Q. Moreover, each customer who is added in a route should have delivery quantity, $DQ_{SelectedCustomer}$, such that it does not exceed the

vehicle's capacity, VC, whereas this customer is excluded from the list since he is associated with a specific route. If a customer's delivery quantity is greater than the remaining vehicle's capacity, VC, a new route is designed that is related to a new vehicle with capacity $VC = Q$. Finally, the combination of the routes construct the random initial non-optimal feasible solution, x_0, of the VRP.

Algorithm 9. *Generate an initial VRP solution*

Inputs: $ListOfCustomers, Q$

$n \leftarrow length(ListOfCustomers), i \leftarrow 1, j \leftarrow 1, VC \leftarrow Q$

while $i \leq n$ ***do***

$SelectedCustomer \leftarrow generate\ random\ integer\ from\ [ListOfCustomers]$

if $DQ_{SelectedCustomer} \leq VC$ ***then***

if $DQ_{SelectedCustomer} = VC$ ***then***

$R_j \leftarrow add\ SelectedCustomer, Update\ ListOfCustomers$

$x_0 \leftarrow R_j, j \leftarrow j + 1, VC \leftarrow Q$

$i \leftarrow i + 1$

else

$R_j \leftarrow add\ SelectedCustomer, Update\ ListOfCustomers$

$VC \leftarrow VC - DQ_{SelectedCustomer}, i \leftarrow i + 1$

end − if

else

$x_0 \leftarrow R_j, j \leftarrow j + 1, VC \leftarrow Q$

$R_j \leftarrow add\ SelectedCustomer, Update\ ListOfCustomers$

$VC \leftarrow VC - DQ_{SelectedCustomer}, i \leftarrow i + 1$

end − if

end − while

$update\ x_0$

Output: $x_0 = \{x_0.R_1, x_0.R_2, \dots\}$

3.2.2.3. Cost Function

The objective of the Simulated Annealing algorithm is to minimize the cost associated with all proposed routes of a specific time period of the planning horizon. Therefore, if a solution x consists of k routes, the cost function, $CostFunction$, is equal to: $\sum_{j=1}^{j=k} Cost(x.R_j)$.

3.2.2.4. Neighboring Solution

At each iteration of the internal cycle of the Simulated Annealing algorithm, a neighboring solution, x, is generated by perturbing the current solution, $CreateNeighboringSolution(x_0)$ (Algorithm 10). The generation of the neighboring solution is based on a random selection among three inter-route improvement algorithms: (a) the move improvement algorithm, (b) the swap improvement algorithm and (c) the cyclic improvement algorithm. The three algorithms attempt to reduce the total route length by moving one or more customers to a different route. It is worth noting that a move is feasible if the demand of the moved customer does not violate the vehicle capacity on the route it is moved to. All of the algorithms are analytically described by Goetschalckx (2011).

Algorithm 10. *Generate a neighboring solution*

Input: x_0

$randN \leftarrow generate\ random\ integer\ from$ [1,3]

$\boldsymbol{if}\ randN = 1\ \boldsymbol{then}$

$x \leftarrow move(x_0)$

$\boldsymbol{end-if}$

$\boldsymbol{if}\ randN = 2\ \boldsymbol{then}$

$x \leftarrow swap(x_0)$

$\boldsymbol{end-if}$

$\boldsymbol{if}\ randN = 3\ \boldsymbol{then}$

$x \leftarrow cyclic(x_0)$

$\boldsymbol{end-if}$

Output: $x = \{x.R_1, x.R_2, \dots\}$

3.2.3. Re-optimization Phase – A Hybrid Approach

Both approaches that are presented in sections 3.2.1 (Genetic Algorithm) and 3.2.2 (Simulated Annealing) are dealt with in an iterative way, thus constructing a hybrid evolutionary optimization algorithm (Algorithm 11) that is related to a re-optimization phase. Hence, a repetitive procedure is applied to obtain a near-optimal feasible solution. The algorithm starts by creating the IRP model based on a specific IRP data set, $IRPInstance$. Then, Algorithm 2 is called to generate an initial population of $PopSize$ individuals as far as the random binary matrices are concerned. Algorithm 3 is called to generate the population of the genetic algorithm based on the random binary matrices. Since an initial population has been constructed, a population of VRP problems is created. Each element of the population consists of a set of VRPs that correspond to each time period of the planning horizon of each individual of the genetic algorithm population. Consequently, for each individual of the population,

Algorithm 8 is used to solve a VRP problem for each time period of the planning horizon where scheduled delivery quantities exist. Since the delivery quantities and times as well as the VRP solutions are available for each individual of the population, respective populations containing information about the inventory levels, $PopIM$, the inventory costs, $PopICM$, and the vehicle routing costs, $PopVRPCM$, can be created. In addition, the total inventory routing cost is calculated for each individual of the population since $PopICM$ and $PopVRPCM$ are available. With respect to the minimum inventory routing cost, the best individual of the population is selected, $BestIRPSol$, whereas the population is sorted.

After initialization, the algorithm proceeds as follows. For each generation, an internal cycle takes place to produce the offspring. At each iteration of the internal cycle, two parents are selected according to their fitness (see Section 3.2.1.3) using Algorithm 4. Algorithm 5 or Algorithm 6 is used randomly to apply a crossover operator to produce two offspring. For each offspring, a mutation operator may be applied using Algorithm 7. After the internal cycle, a new population has been created consisting of both parents, $Population$, and offspring, $newPopulation$. Furthermore, to avoid duplicate individuals, a procedure called $replaceDuplicatesIRP()$ is applied. This procedure uses Algorithm 7, applying the proposed mutation operator to duplicate individuals. Afterward, the new best IRP solution, $BestIRPSol2$, is calculated and compared with the previous best IRP solution, $BestIRPSol$. If the second solution is better, it is accepted. Otherwise, the new population is sorted and only the first $PopSize$ individuals are selected to keep the population size constant from one generation to the next.

Algorithm 11. *Hybrid evolutionary optimization algorithm*

Inputs: $PopSize, MaxGen, MutationRate, Q, IRPInstance$

$IRPmodel \leftarrow LoadIRPData(IRPInstance), PopBM \leftarrow$ ***call*** $Algorithm\ 2$

$Pop \leftarrow$ ***call*** $Algorithm\ 3$

$pVRPs \leftarrow createVRPs(IRPmodel, Pop, Q)$

for *each element in* $pVRPs$ ***do***

$PopVRPs \leftarrow$ ***call*** $Algorithm\ 8$

end* − *for

$PopIM \leftarrow createPopIM(IPRmodel, Pop), PopICM \leftarrow createPopICM(IRPmodel, PopIM)$

$PopVRPCM \leftarrow createPopVRPCM(PopVRPs)$

$PopIRPCM \leftarrow createPopIRPCM(PopICM, PopVRPCM)$

$BestIRPSol \leftarrow [PopBM\{best\}, Pop\{best\}, PopVRPs\{best\}, PopIRPCM\{best\}]$

$[PopulationBM, Population, PopulationIRPCM] \leftarrow popSort(PopBM, Pop, PopIRPCM)$

$selectedParents \leftarrow zeros(1,2)$

for $indx = 1{:}\,MaxGen$ ***do***

for $k = 1{:}\,2{:}\,PopSize$ ***do***

$Fitness \leftarrow CreateFitness(PopulationIRPCM)$

$selectedParents(1) \leftarrow$ **call** *Algorithm* 4

$selectedParents(2) \leftarrow$ **call** *Algorithm* 4, $randN \leftarrow U(0,1)$

if $randN < 0.5$ **then**

call *Algorithm* 5

else

call *Algorithm* 6

end − if

for $i = k{:}\, k+1$ **do**

call *Algorithm* 7

end − for

end − for

$PopA \leftarrow [PopulationBM, newPopulationBM]$

$PopB \leftarrow [Population, newPopulation]$

$[PopA, PopB] \leftarrow replaceDuplicatesIRP(PopA, PopB), PopBM \leftarrow PopA$

$Pop \leftarrow PopB$

$pVRPs \leftarrow createVRPs(IRPmodel, Pop, Q)$

for *each element in* $pVRPs$ **do**

$PopVRPs \leftarrow$ **call** *Algorithm* 8

end − for

$PopIM \leftarrow createPopIM(IPRmodel, Pop)$

$PopICM \leftarrow createPopICM(IRPmodel, PopIM)$

$PopVRPCM \leftarrow createPopVRPCM(PopVRPs)$

$PopIRPCM \leftarrow createPopIRPCM(PopICM, PopVRPCM)$

$BestIRPSol2 \leftarrow [PopBM\{best\}, Pop\{best\}, PopVRPs\{best\}, PopIRPCM\{best\}]$

if $BestIRPSol2 < BestIRPSol$ **then**

$BestIRPSol \leftarrow BestIRPSol2$

else

$[PopulationBM, Population, PopulationIRPCM] \leftarrow popSort(PopBM, Pop, PopIRPCM)$

$PopulationBM \leftarrow PopulationBM(1{:}\, PopSize)$

$Population \leftarrow Population(1{:}\, PopSize)$

end − if

end − for

Output: *BestIRPSol*

Based on the example presented in Section 3.2.2.1, the following figure illustrates the IRP solution related to the best individual of the population through 50 generations (Fig. 12).

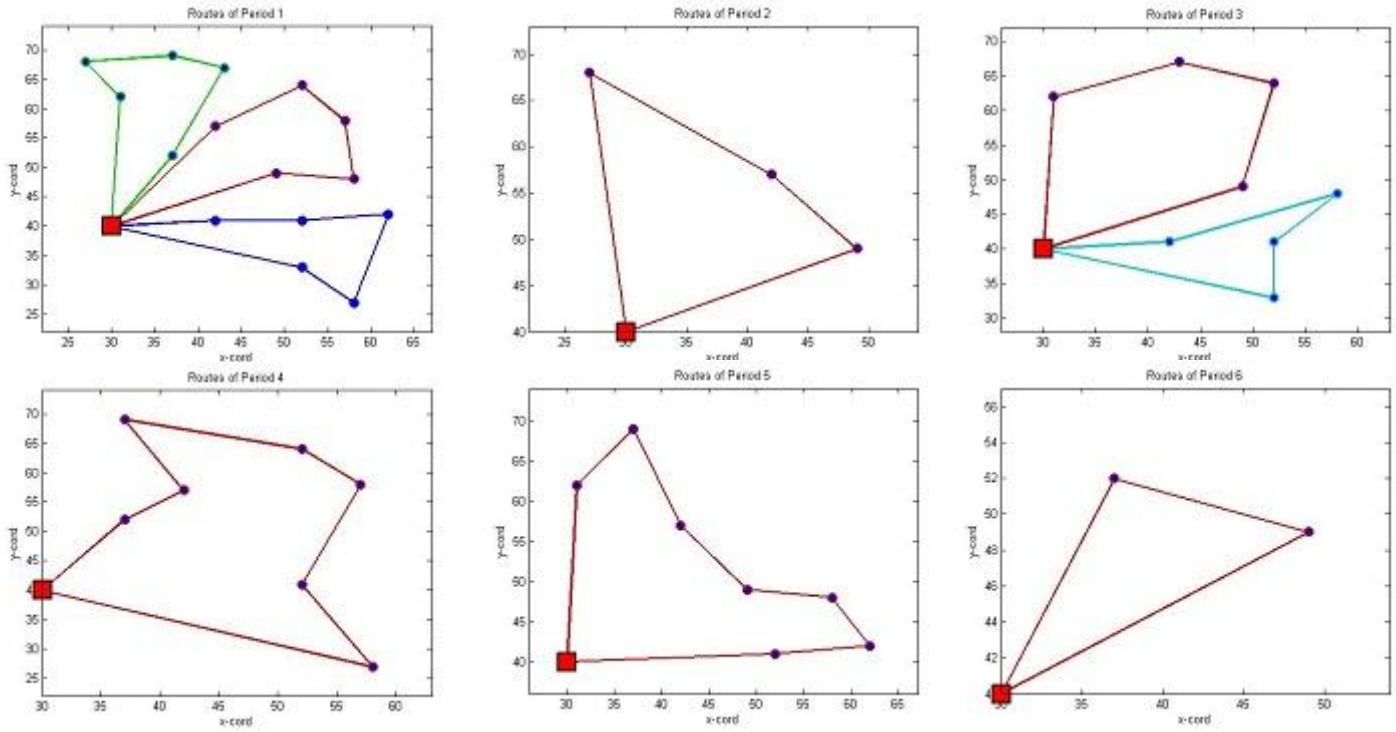


Fig. 12. IRP solution for the IRP sample problem – 50 generations

Table 2 presents the routes that take place in each time period of the planning horizon.

Table 2

Cost information and routes for the IRP sample problem – 50 generations

IRP Solution		
Routes of Period 1	**Routes of Period 2**	**Routes of Period 3**
Route 1: 0-2-13-8-3-10-0 Route 2: 0-4-11-15-12-1-0 Route 3: 0-6-7-9-14-5-0	Route 1: 0-2-10-11-0	Route 1: 0-2-3-12-4-0 Route 2: 0-6-13-7-5-0
Routes of Period 4	**Routes of Period 5**	**Routes of Period 6**
Route 1: 0-1-10-15-3-8-7-14-0	Route 1: 0-7-9-13-2-10-15-4-0	Route 1: 0-2-1-0
Total VRP Cost	**Total Inventory Control Cost**	**Total IRP Cost**
736.7608	245.1916	981.9524

If the number of generations is increased (e.g., 100 instead of 50), then a better solution is obtained (Fig. 13, Table 3).

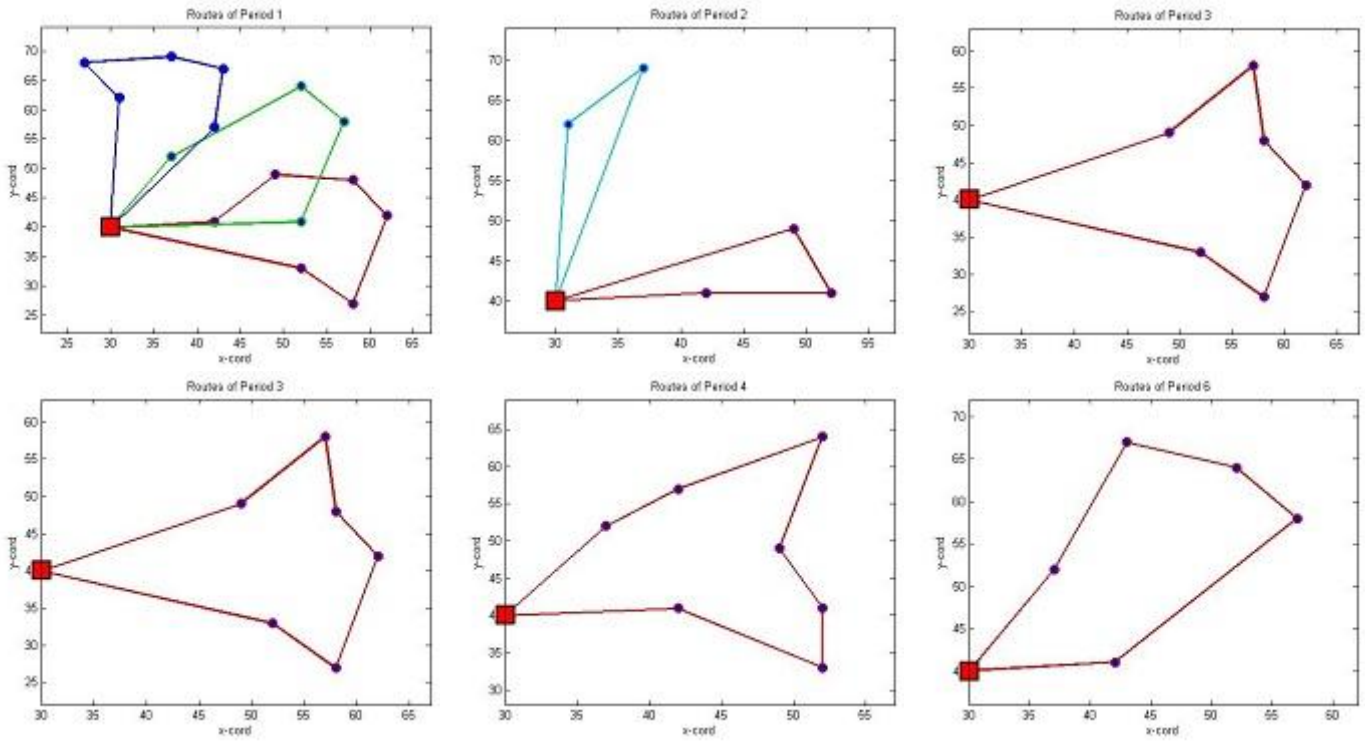


Fig. 13. IRP solution for the IRP sample problem – 100 generations

Table 3

Cost information and routes for the IRP sample problem – 100 generations

IRP Solution		
Routes of Period 1	**Routes of Period 2**	**Routes of Period 3**
Route 1: 0-5-14-9-13-2-6-0 Route 2: 0-1-3-8-7-0 Route 3: 0-4-11-15-12-10-0	Route 1: 0-6-7-2-0 Route 2: 0-4-15-0	Route 1: 0-2-8-13-9-14-5-0
Routes of Period 4	**Routes of Period 5**	**Routes of Period 6**
Route 1: 0-1-10-3-2-7-5-6-0	Route 1: 0-2-7-9-14-5-0	Route 1: 0-6-8-3-12-1-0
Total VRP Cost	**Total Inventory Control Cost**	**Total IRP Cost**
711.8943	231.7815	943.6758

The next figure (Fig. 14) illustrates a typical graph of the minimum IRP cost in the population as a function of generation number. Specifically, the figure shows a typical evolutionary algorithm convergence behavior for the IRP sample problem. In the first case (50 generations), the evolutionary algorithm has mostly converged after 40 generations; in the second case (100 generations), after 95 generations. However, it appears that in both cases, the best candidate solution continues to improve for a few more generations.

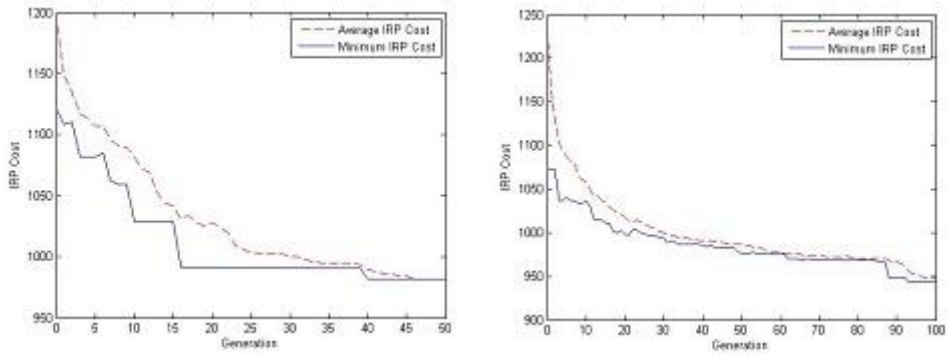


Fig. 14. Evolutionary algorithm convergence behavior

3.2.4. Computational Experiments and Results

This section presents the computational results of the proposed hybrid evolutionary optimization algorithm described in Section 4. The algorithm was developed in the MATLAB programming language and executed on a DELL personal computer with an Intel® Core™ i3-2120, clocked at 3.30 GHz, a microprocessor with 4 GB of RAM memory under the operating system Microsoft Windows 7 Professional. As mentioned in Chapter 2, new benchmark instances were designed. Consequently, the efficiency and the effectiveness of the proposed algorithm cannot be compared to other published IRP studies using benchmark instances previously introduced. This is due to the differentiated manner in which the proposed algorithm operates based on the assumptions presented in Chapter 2 and Section 3.1, respectively. However, this section validates the evolutionary algorithm and then evaluates its performance by comparing the algorithm's solutions with solutions obtained by solving a VRP problem for each time period of the planning horizon based on the known demands (the planning phase is ignored). The algorithm has been tested on a newly introduced set of 18 IRP benchmark instances described in the following. All benchmark instances and their computational results are available at http://www.msl.aueb.gr/files/GaSaIRP.zip.

3.2.4.1. Set of Benchmark Instances

New datasets have been developed by generalizing the well-known dataset P of Augerat et al. (1998). These datasets are divided into two classes. The first class (Class A) contains the instances with planning horizon $H = 6$ time periods (days) and a high inventory holding cost of the customers, $h_i \in [0.1,0.5] \forall i \in C$. The second class (Class B) contains the instances with planning horizon $H = 6$ time periods (days) and low inventory holding costs of the customers, $h_i \in [0.01,0.05] \forall i \in C$. The datasets are named in the form of "IRP_n**X**_p**Y**_HC" (first class) or

"IRP_n**X**_p**Y**_LC" (second class) strings, where "**X**" stands for the number of customers and "**Y**" stands for the number of time periods. For instance, the problem IRP_n15_p6_HC represents a test problem with 15 customers, a planning horizon of 6 days and high inventory holding costs at the customers. Different problem sizes, based on the total number of customers, were designed in each class. Class A contains problems with 15, 20, 22, 39, 44, 50, 54, 59, 64, 69, 75 and 100 customers, while the Class B contains problems with 15, 20, 22, 39, 44 and 50 customers. Vertex coordinates are kept the same as in the study by Augerat et al. (1998). The distance matrix is obtained by calculating the Euclidean distances (symmetric cost matrix). Demand exists for each customer at each time period of the planning horizon. Customer demand at each time period was generated according to the Poisson distribution, $Poisson(\lambda)$, where λ is the rate parameter. For each customer, the rate parameter is equal to his demand in the single-period VRP problem of Augerat et al. (1998). An unlimited fleet of identical vehicles with capacity Q is available for the distribution of the product. The vehicle capacity varies from 200 to 300 units of product. At the beginning of the planning horizon, all customers have zero inventory levels. Each customer has a sufficient maximum inventory level to satisfy his storage needs during the planning horizon. Namely, for each customer $i \in C$, $\sum_{t=1}^{t=H} d_i^t \leq U_i$. Finally, the supplier has a sufficient supply of products that can cover customers' demands throughout the planning horizon.

3.2.4.2. Parameter Setting

The proposed hybrid evolutionary algorithm has seven parameters to be set. Four of the parameters are associated with the Simulated Annealing algorithm. T_{max} determines the initial value of the temperature. $EXTcyc$ and $INTcyc$ are the maximum number of iterations for the external and internal cycle, respectively. In addition, $alpha$ reflects the cooling rate of the geometric schedule. The other three parameters are related to the Genetic Algorithm. Particularly, $PopSize$ defines the size of the population, $MaxGen$ sets the maximum number of generations (i.e., maximum number of iterations), while $MutationRate$ corresponds to the mutation rate. Based on the minimal cost criterion, the value of each parameter is determined after some experiments in the context of the VRP and the Inventory Control Problem, respectively. The values of the above parameters for each instance are presented in Table 4.

Table 4

Parameters of the hybrid evolutionary optimization algorithm

Instance	T_{max}	$EXTcyc$	$INTcyc$	$alpha$	$PopSize$	$MaxGen$	$MutationRate$
Class A Instances							
IRP_n15_p6_HC	100	1500	100	0.98	10 (20)	50 (100)	0.08
IRP_n20_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n22_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n39_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n44_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n50_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n54_p6_HC	100	1500	100	0.98	10 (20)	50 (70)	0.08
IRP_n59_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n64_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n69_p6_HC	100	1500	100	0.98	10	50	0.08
IRP_n75_p6_HC	100	1500	100	0.98	10 (20)	50 (70)	0.08
IRP_n100_p6_HC	100	1500	100	0.98	10 (20)	50 (70)	0.08
Class B Instances							
IRP_n15_p6_LC	100	1500	100	0.98	10	50	0.08
IRP_n20_p6_LC	100	1500	100	0.98	10	50	0.08
IRP_n22_p6_LC	100	1500	100	0.98	10	50	0.08
IRP_n39_p6_LC	100	1500	100	0.98	10	50	0.08
IRP_n44_p6_LC	100	1500	100	0.98	10	50	0.08
IRP_n50_p6_LC	100	1500	100	0.98	10	50	0.08

3.2.4.3. Results

This section presents the computational results for the 12 and 6 instances of Class A and B, respectively. Since the algorithm cannot be compared to other published IRP studies, the best solution obtained from the proposed algorithm (IRP) is compared to the best solution obtained if the planning phase is ignored ($p-VRP$). In the aftermath of ignoring the planning phase, a VRP problem needs to be solved for each day of the planning horizon according to daily demand. The proposed Simulated Annealing algorithm for the routing phase is then used to solve a daily VRP problem through the planning horizon. To compare the results, the following gap percentage formula is used: $Gap\ (\%) = (Sol_{IRP} - Sol_{p-VRP}) \times \frac{1}{Sol_{p-VRP}} \times 100$. The Sol_{p-VRP} corresponds to the solution obtained by solving the daily VRPs according to the known daily demands, while the Sol_{IRP} determines the solution obtained by applying the proposed hybrid evolutionary optimization algorithm. Since the Sol_{IRP} is compared with the Sol_{p-VRP}, a positive gap means that the Sol_{p-VRP} is outperformed. The computational

results obtained are summarized in Tables 5 (Class A instances) and 6 (Class B instances). For the $p-VRP$ problem, the total vehicle routing cost is presented, whereas for the IRP problem, the total cost is separated in terms of its transportation and inventory cost. In addition, the last column of the table shows the gap between the two problems reflecting the respective relative error.

Table 5

Experimental results (first class of instances)

Instance	**p-VRP**	**IRP**			**p-VRP – IRP**
	Vehicle Routing Cost	Vehicle Routing Cost	Inventory Holding Cost	Total Cost	Gap (%)
IRP_n15_p6_HC	1141.4608	736.7608	245.1916	**981.9524**	-13.9741
		711.8943	231.7815	**943.6758**	-17.3274
IRP_n20_p6_HC	1313.139	947.7804	323.2335	**1271.0139**	-3.2080
IRP_n22_p6_HC	1348.1933	1073.4663	245.6849	**1319.1511**	-2.1542
IRP_n39_p6_HC	2481.3249	1638.3222	648.6909	**2287.0132**	-7.8310
IRP_n44_p6_HC	2832.1262	1311.7218	743.3693	**2055.0912**	-27.4365
IRP_n50_p6_HC	2934.1541	1546.7005	780.6426	**2327.3431**	-20.6810
IRP_n54_p6_HC	2991.2924	1931.812	964.661	**2896.4729**	-3.1669
		2281.1826	801.8551	3083.0377	3.0671
IRP_n59_p6_HC	**3472.2657**	2560.0989	958.1427	3518.2415	1.3241
IRP_n64_p6_HC	**3707.7601**	2691.5032	1143.1868	3834.69	3.4234
IRP_n69_p6_HC	**4014.4543**	3015.5997	1215.7251	4231.3248	5.4022
IRP_n75_p6_HC	**3923.5927**	3029.7364	1208.7383	4238.4747	8.0253
		2930.6974	1190.7582	4121.4556	5.0429
IRP_n100_p6_HC	4854.3807	3791.122	1190.755	4981.877	2.6264
		3645.1169	1188.8117	**4833.9286**	-0.4213

Table 6

Experimental results (second class of instances)

Instance	**p-VRP**	**IRP**			**p-VRP – IRP**
	Vehicle Routing Cost	Vehicle Routing Cost	Inventory Holding Cost	Total Cost	Gap (%)
IRP_n15_p6_LC	1141.4608	664.3223	25.0305	**689.3528**	-39.6078
IRP_n20_p6_LC	1313.139	774.6633	36.3499	**811.0132**	-38.2386
IRP_n22_p6_LC	1348.1933	986.0693	26.7627	**1012.832**	-24.8749
IRP_n39_p6_LC	2481.3249	1406.3136	63.409	**1469.7226**	-40.7686
IRP_n44_p6_LC	2832.1262	1428.9866	73.6921	**1502.6787**	-46.9417
IRP_n50_p6_LC	2934.1541	1276.7132	191.8418	**1468.555**	-49.9496

Based on Table 5, it can be concluded that better solutions are obtained when the planning phase is considered. The ability of each customer to have storage enables a significant decrease in the vehicle routing cost, reducing the total number of routes

during the planning horizon. For 8 of the 12 instances, the evolutionary algorithm provides better solutions with gaps in the interval of -27.4365 percent to -0.4213 percent. It should not pass unnoticed that even in cases where the $p-VRP$ provides better solutions, a change in parameters of the evolutionary algorithm, such as population size and maximum number of generations (see Table 4), results in a gap improvement. Specifically, for the first, the eleventh and the twelfth instance of Class A, the gap was improved by 24%, 37.16% and 116.04%, respectively. In particular, in the last case, the improvement of the gap was such that the best solution of the instance was improved significantly. Despite the better solution obtained from the $p-VRP$, the change in the parameters led to the evolutionary algorithm providing an even better solution.

Furthermore, Table 6 shows the computational results related to the instances of Class B. As can be observed, in all cases, the evolutionary algorithm provides better solutions than the $p-VRP$, with gaps in the interval of -49.9496 percent to -24.8749 percent. The results indicate that if a small inventory holding cost is applied to each customer, better solutions can be obtained, significantly reducing the total vehicle routing cost and designating the importance of integrating supply chain activities.

To illustrate in more detail the behavior of the proposed algorithm, more information is presented about the vehicles (number of routes) used in each time period of the planning horizon in Tables 7 and 8.

Table 7

Number of vehicles used during the planning horizon (first class of instances)

Instance	**p-VRP**							**IRP**						
	P1	P2	P3	P4	P5	P6	No. of Routes	P1	P2	P3	P4	P5	P6	No. of Routes
IRP_n15_p6_HC	2	2	2	2	2	2	12	3	1	2	1	1	1	9
								3	2	1	1	1	1	9
IRP_n20_p6_HC	2	2	2	2	2	2	12	4	2	1	2	1	1	11
IRP_n22_p6_HC	2	2	2	2	2	2	12	4	2	1	2	2	1	12
IRP_n39_p6_HC	3	3	3	3	3	3	18	6	2	3	2	1	1	15
IRP_n44_p6_HC	3	3	3	3	3	3	18	14	0	1	1	0	0	16
IRP_n50_p6_HC	3	3	3	3	3	3	18	16	0	2	1	1	0	20
IRP_n54_p6_HC	4	4	4	4	4	4	24	16	2	3	1	1	1	24
								7	4	4	3	4	2	24
IRP_n59_p6_HC	4	4	4	5	4	4	25	8	4	4	5	3	2	26
IRP_n64_p6_HC	5	5	5	5	5	4	29	12	5	5	3	2	1	28
IRP_n69_p6_HC	5	5	5	5	5	5	30	9	6	4	4	4	3	30
IRP_n75_p6_HC	5	5	5	5	5	5	30	11	5	5	4	4	3	32
								10	5	5	5	3	3	31

IRP_n100_p6_HC	6	6	5	5	6	6	34	9	6	5	5	6	3	34
								11	5	5	5	5	3	34

Table 8

Number of vehicles used during the planning horizon (second class of instances)

Instance	p-VRP							IRP						
	P1	P2	P3	P4	P5	P6	No. of Routes	P1	P2	P3	P4	P5	P6	No. of Routes
IRP_n15_p6_LC	2	2	2	2	2	2	12	4	2	2	1	0	0	9
IRP_n20_p6_LC	2	2	2	2	2	2	12	7	2	1	0	1	0	11
IRP_n22_p6_LC	2	2	2	2	2	2	12	3	2	2	2	1	1	11
IRP_n39_p6_LC	3	3	3	3	3	3	18	8	3	2	1	1	0	15
IRP_n44_p6_LC	3	3	3	3	3	3	18	14	1	1	1	1	0	18
IRP_n50_p6_LC	3	3	3	3	3	3	18	17	1	0	0	0	0	18

Both tables show that the maximum number of vehicles is used mainly at the initial time periods of the planning horizon. This can be explained (a) by the fact that the inventory level of each customer is equal to zero at the beginning of the planning horizon and (b) by the usage of the backward delivery exchange mutation operator. Actually, the operator satisfies constraint (3), thus avoiding any stock-out. However, it is interesting to observe that with higher inventory holding costs (Table 5), the optimal solution visits customers more frequently. Furthermore, if low-inventory holding costs (Table 6) are applied to the customers, a decrease in the number of times a customer is visited during the planning horizon can be observed since most of the delivery quantities are scheduled at the initial time periods. On the other hand, in the context of the p-VRP, the number of vehicles is nearly the same, as a specific VRP problem should be solved on a daily basis.

To visually verify the above conclusions, the following figures illustrate the solutions of (a) the IRP_n50_p6 (no inventory holding costs), (b) the IRP_n50_p6_HC (high inventory holding cost) and (c) the IRP_n50_p6_LC (low inventory holding cost) benchmark instances. As regards the first solution, 3 routes are scheduled for each day of the planning horizon since the inventory allocation problem is ignored. Concerning the other two solutions, it can be observed that the evolutionary algorithm changes its behavior based on inventory holding cost information. Specifically, the second solution shows that routes are scheduled only for the time periods 1, 3, 4 and 5 (4 of the 6 days). The third solution, due to the low inventory holding costs, indicates the routes that should be scheduled for the time periods 1 and 2 (2 of the 6 days). Therefore, with higher inventory holding costs, the solution visits customers more frequently.

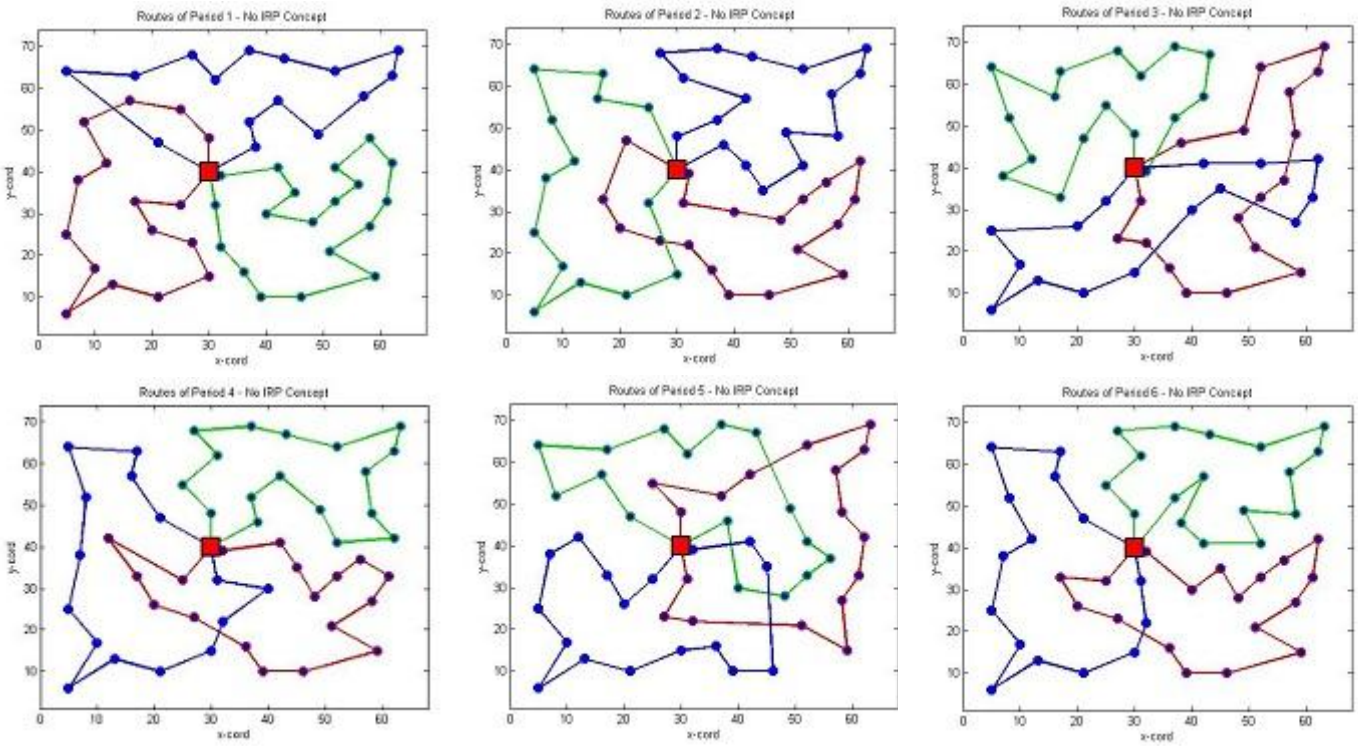


Fig. 15. Solving a VRP problem on a daily basis (ignoring the planning phase)

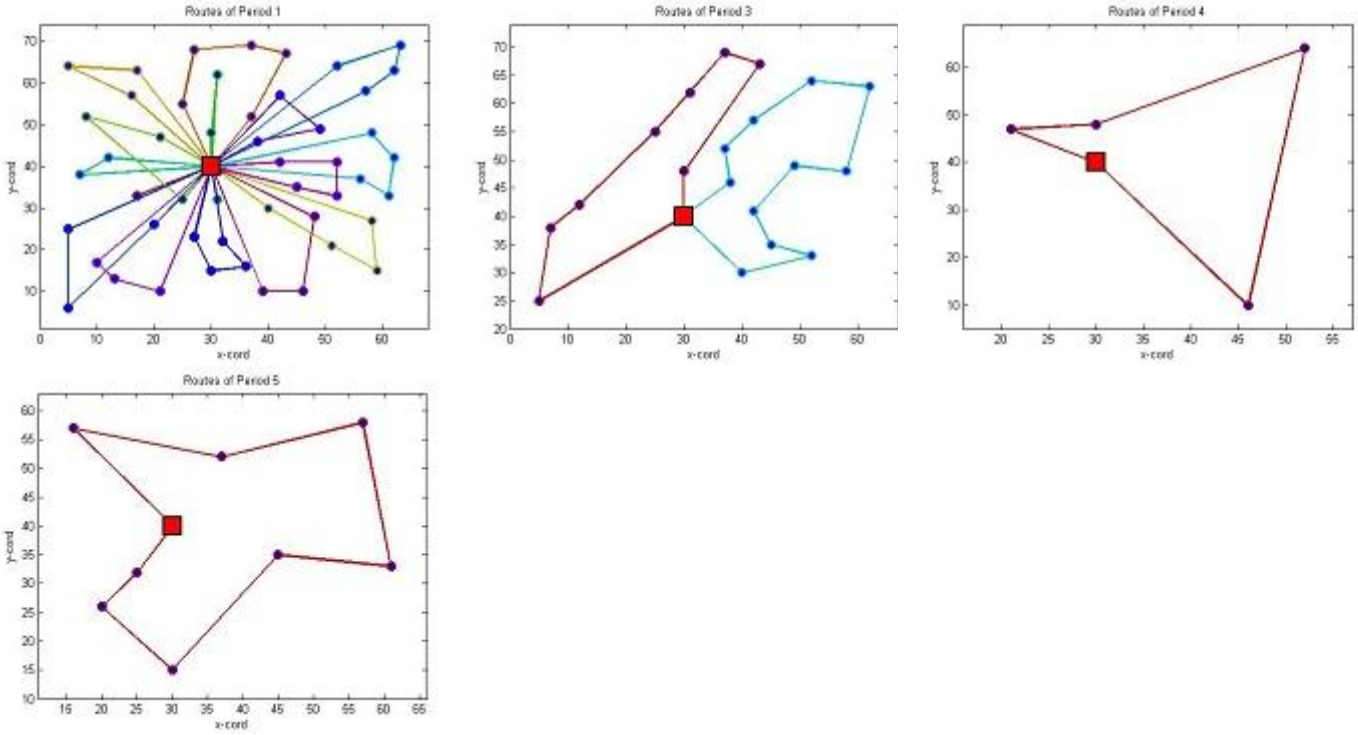


Fig. 16. Solving the IRP with high inventory holding costs at the customers

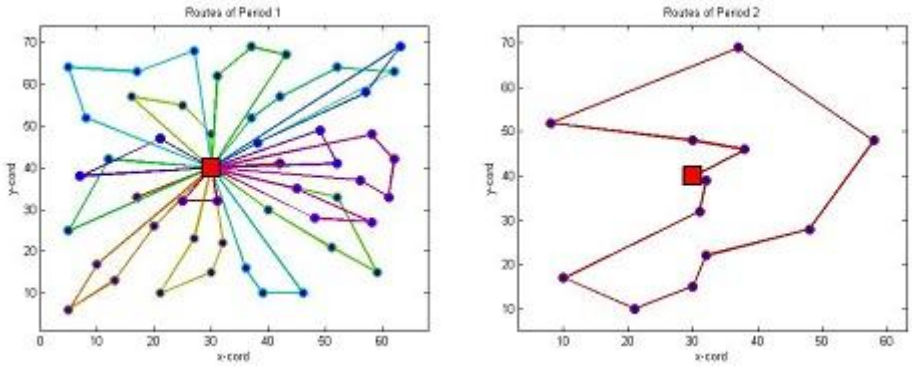
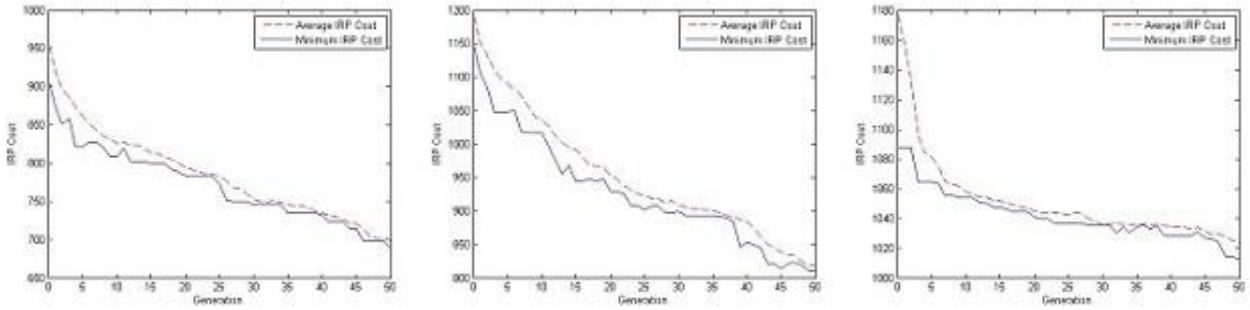
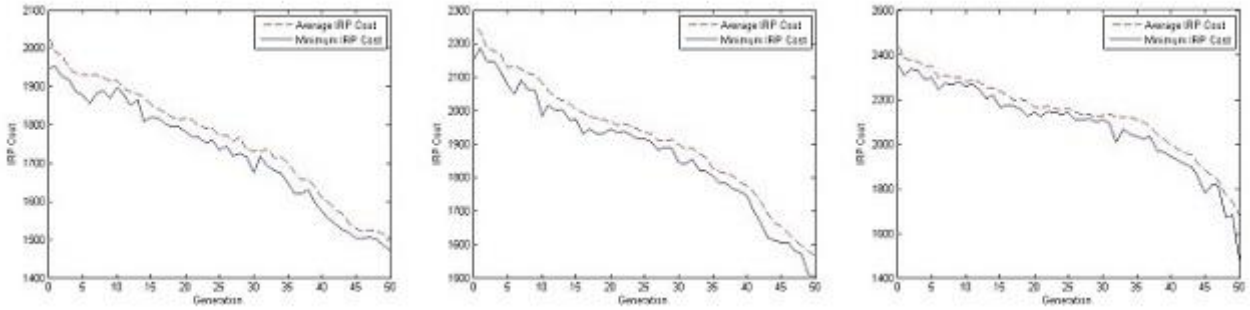


Fig. 17. Solving the IRP with low inventory holding costs at the customers

Finally, the convergence of fitness values regarding the instances of Class B and A is presented in figures 18 and 19, respectively.

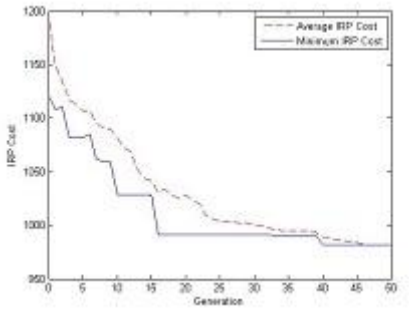


IRP_n15_p6_LC (50 Generations) IRP_n20_p6_LC (50 Generations) IRP_n22_p6_LC (50 Generations)

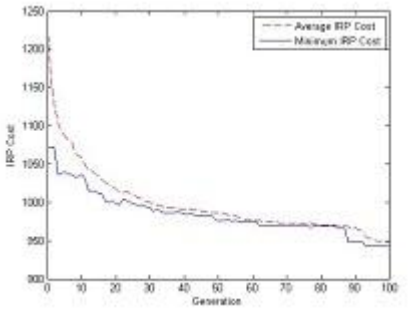


IRP_n39_p6_LC (50 Generations) IRP_n44_p6_LC (50 Generations) IRP_n50_p6_LC (50 Generations)

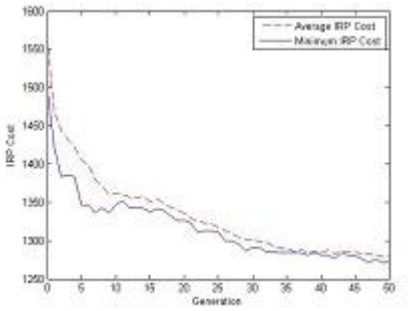
Fig. 18. Convergence of fitness values (instances of class B)



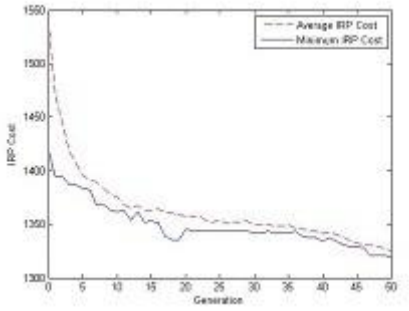
IRP_n15_p6_HC (50 Generations)



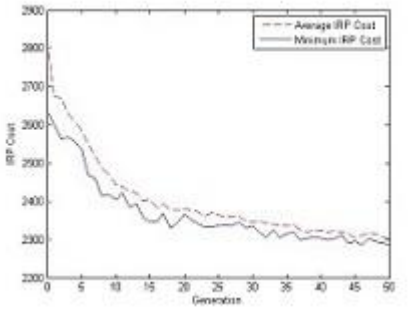
IRP_n15_p6_HC **(100 Generations)**



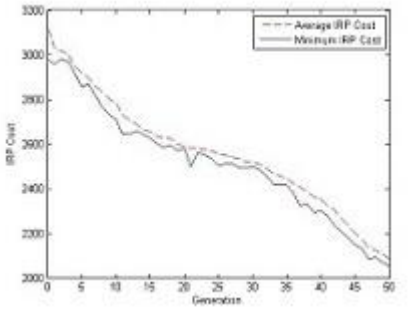
IRP_n20_p6_HC (50 Generations)



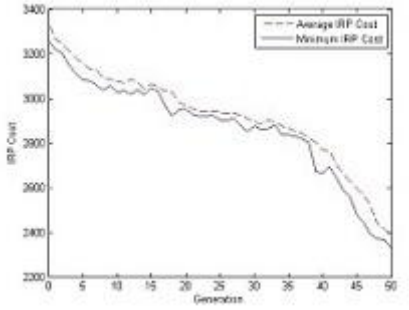
IRP_n22_p6_HC (50 Generations)



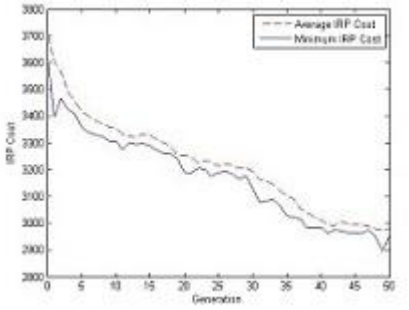
IRP_n39_p6_HC (50 Generations)



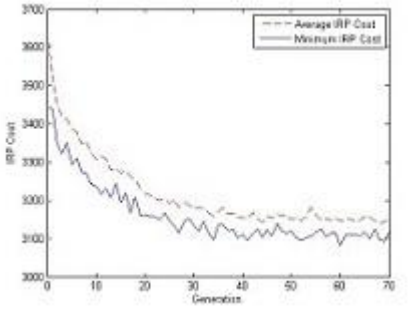
IRP_n44_p6_HC (50 Generations)



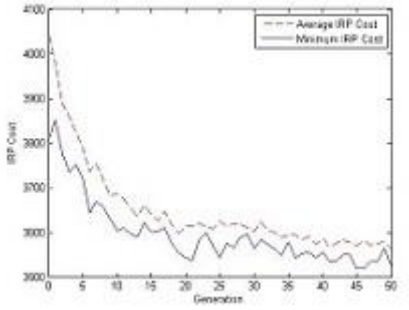
IRP_n50_p6_HC (50 Generations)



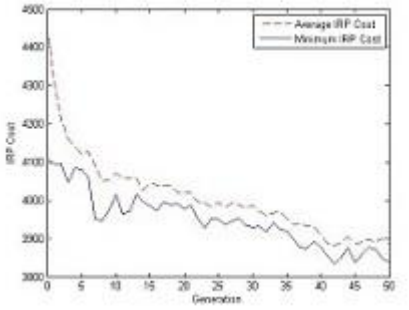
IRP_n54_p6_HC (50 Generations)



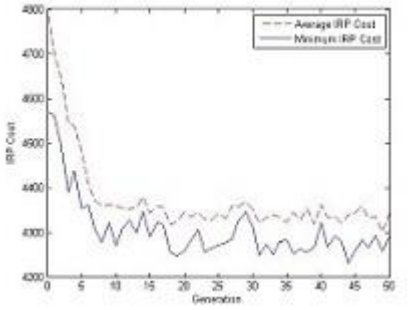
IRP_n54_p6_HC **(70 Generations)**



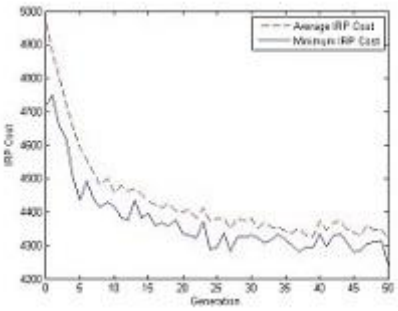
IRP_n59_p6_HC (50 Generations)



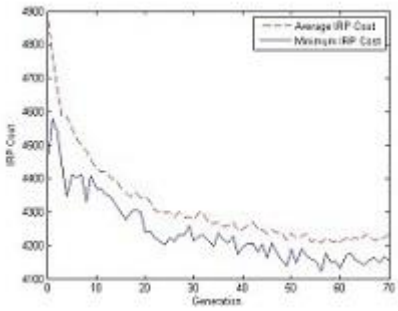
IRP_n64_p6_HC (50 Generations)



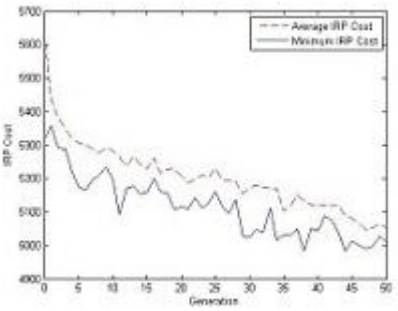
IRP_n69_p6_HC (50 Generations)



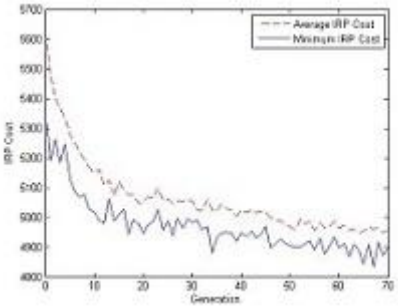
IRP_n75_p6_HC (50 Generations)



IRP_n75_p6_HC **(70 Generations)**



IRP_n100_p6_HC (50 Generations)



IRP_n100_p6_HC **(70 Generations)**

Fig. 19. Convergence of fitness values (instance of class A)

For each instance, fluctuations can be observed during convergence. However, the entirety direction of evolution indicates improvement with respect to the minimization of inventory routing problem cost. For the majority of instances, it appears that the best candidate solution would continue to improve for a few hundred more generations.

3.2.5. Conclusions and Future Work

In this chapter, a hybrid evolutionary optimization algorithm was introduced to handle the IRP. The chapter gives more emphasis to how a Genetic Algorithm (population-based search meta-heuristic) can be used in hybrid synthesis with a Simulated Annealing Algorithm (single-point search meta-heuristic) for the solution of the IRP. Particularly, the Genetic Algorithm is related to the planning phase of the hybrid approach to determine the delivery times and quantities, while the Simulated Annealing algorithm is associated with the routing phase to determine the routes of each individual of the population. Both algorithms are dealt with in an iterative way to define a re-optimization phase. In this study, stock-outs or lost sales are not allowed, and therefore no shortage costs or costs related to lost sales are included in the objective function. This is a characteristic that differentiates the proposed algorithm from other works most closely related to this study.

The algorithm has been tested on a newly introduced set of 18 IRP benchmark instances by comparing the algorithm's solutions with the solutions obtained by solving a VRP problem for each time period of the planning horizon based on known demand (the planning phase is ignored). The computational results show that the proposed algorithm is outperformed, simultaneously verifying the benefits obtained by the integration of the inventory and the vehicle routing decisions. The algorithm can be even further improved. In terms of future research, the goals are to (a) explore more deeply the parameters of the Genetic Algorithm and the Simulated Annealing Algorithm, (b) explore the algorithm behavior in other problems (instances) and (c) focus on the development of other meta-heuristic approaches for the solution of the IRP. Finally, the proposed algorithm can be extended to complicated problems such as the Inventory Routing Problem with Time Windows (IRPTW) and its variations.

Chapter 4

The Inventory Routing Problem with Time Windows

The purpose of this chapter is to propose a solution algorithm for the Inventory Routing Problem with Time Windows (IRPTW). The IRPTW reflects a multi-functional problem that attempts to integrate two different functions within the supply chain network, i.e., planning and routing. In particular, planning is associated with the Inventory Control Problem (ICP), while routing is related to the Vehicle Routing Problem with Time Windows (VRPTW). It is worth noting that the integration of ICP-VRPTW problems has scarcely been studied in the literature. The IRPTW is a generalization of the IRP involving the added complexity that every customer should be served within a given time window. The basic notion of this Chapter is (a) to formulate a mathematical problem and (b) to present a two-phase solution algorithm to handle the IRPTW. Testing instances are established to investigate algorithmic performance, and the computational results are then reported.

4.1. Problem Description and Mathematical Formulation

The IRPTW is a variation of the classical Vehicle Routing Problem with Time Windows (VRPTW) formulation. Whereas the VRPTW focuses on a single period, the IRPTW considers a multi-period time horizon, typically measured in terms of days. The IRPTW can be defined on a complete directed graph $G = (N, A)$ where $N = \{0, n+1\} \cup \{1, \dots, n\}$ is the set of nodes and $A = \{(i,j): i,j \in N, i \neq j\}$ is the set of arcs. Arcs $1, \dots, n$ correspond to the customers, whereas 0 and $n+1$ represent the single depot (origin-depot and destination-depot). The set of arcs represents connections between the depot and the customers and among customers. No arc terminates in node 0, and no arc originates from node $n+1$. The proposed model deals with the repeated distribution of a single product from a single supplier to a set of geographically dispersed customers $C = \{1, \dots, n\}$ over a given time horizon of length H. The set of time horizons is denoted by $T = \{1, \dots, H\}$. Each customer $i \in C$ faces a different demand d_i^t per time period $t \in T$. It is assumed that the depot has a sufficient supply of items that can cover all customers' demands throughout the planning horizon. To each arc $(i,j) \in A$, where $i \neq j$, a travel cost c_{ij} and travel time t_{ij} are associated. The cost and travel time matrices satisfy the triangle inequality. Nodes are associated with points of the plane having the given coordinates $(x_i, y_i) \forall i \in N$, and the travel cost c_{ij} for each arc $(i,j) \in A$ is defined as the Euclidean distance between the two nodes $i, j \in N$.

A fleet of m homogenous vehicles, with capacity Q, is available for the distribution of the product. The fleet of vehicles is denoted by $K = \{1, \dots, m\}$. Each customer $i \in C$ is associated with a time interval $[e_i, l_i]$, called a time window and a service time s_i, where $e_i \leq l_i \ \forall i \in C$. The service of each customer must start within the associated time window, and the vehicle must stop at the customer location for s_i time instants, where $0 \leq s_i \leq l_i - e_i \ \forall i \in C$. Moreover, in case of early arrival at the location of customer $i \in C$, the vehicle generally is allowed to wait until time instant e_i, i.e., until the service may start. Therefore, a vehicle must arrive at the customer $i \in C$ before l_i. It can arrive before e_i but the customer $i \in C$ will not be serviced before. The depot has also time windows $[e_0, l_0]$ and $[e_{n+1}, l_{n+1}]$ where $e_0 = e_{n+1}$ and $l_0 = l_{n+1}$. The time windows associated with the depot represent the earliest possible departure from the depot as well as the latest possible return time at the depot, respectively. As a result, vehicles may not leave the depot before e_0 and must be back before or at time l_{n+1}. In addition, $s_0 = s_{n+1} = 0$. Each customer maintains his own inventory up to capacity $U_i \ \forall i \in C$. At the beginning of the planning horizon each customer $i \in C$ has an initial inventory level of $I_i^0 = U_i$ of product.

Furthermore, the formulation uses the following decision variables:

- w_{ik}^t: the amount of delivery to customer $i \in C$ in period $t \in T$ by vehicle $k \in K$.
- x_{ijk}^t: a binary variable that is equal to 1 if vehicle $k \in K$ drives from node i to node $j \ \forall (i,j) \in A$ where $i \neq j, j \neq n+1, j \neq 0$, and 0 otherwise.
- a_{ik}^t: the time vehicle $k \in K$ starts to service customer $i \in C$.
- y_{ik}^t: a binary variable that is equal to 1 if customer $i \in C$ is visited by vehicle $k \in K$ in period $t \in T$, and 0 otherwise.
- z_k^t: a binary variable that is equal to 1 if vehicle $k \in K$ is used in period $t \in T$, and 0 otherwise.
- I_i^t: a nonnegative variable indicating the inventory level at customer $i \in C$ at the end of period $t \in T$.

Moreover, stock-outs are not allowed at the customers, while the quantities delivered by each vehicle in each route cannot exceed the vehicle capacity. As far as the replenishment policy is concerned, an Order-up-to Level (OL) policy is considered, in which any customer has defined a maximum inventory level and every time a customer is served, the delivered quantity is such that the maximum inventory level at the customer is reached. After defining the necessary parameters and decision variables, the IRPTW can be formulated as shown below:

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} \sum_{k \in K} \sum_{t \in T} x_{ijk}^t \tag{15}$$

Subject to:

$$I_i^0 = U_i, \forall i \in C \tag{16}$$

$$I_i^{t-1} - I_i^t + \sum_{k \in K} w_{ik}^t = d_i^t, \forall i \in C, \forall t \in T \quad (17)$$

$$I_i^t \leq U_i, \forall i \in C, \forall t \in T \quad (18)$$

$$\sum_{i \in C} w_{ik}^t \leq Q z_k^t, \forall k \in K, \forall t \in T \quad (19)$$

$$\sum_{k \in K} y_{ik}^t \leq 1, \forall i \in C, \forall t \in T \quad (20)$$

$$\sum_{k \in K} y_{0k}^t = m, \forall t \in T \quad (21)$$

$$\sum_{k \in K} y_{n+1,k}^t = m, \forall t \in T \quad (22)$$

$$\sum_{j \in N} x_{jik}^t = y_{ik}^t, \forall i \in N \backslash \{0\}, \forall k \in K, \forall t \in T \quad (23)$$

$$\sum_{j \in N} x_{ijk}^t = y_{ik}^t, \forall i \in N \backslash \{n+1\}, \forall k \in K, \forall t \in T \quad (24)$$

$$a_{ik}^t + s_i + t_{ij} \leq a_{jk}^t + M\left(1 - x_{ijk}^t\right), \forall i, j \in N, \forall k \in K, \forall t \in T \quad (25)$$

$$a_{ik}^t \geq e_i y_{ik}^t, \forall i \in N, \forall k \in K, \forall t \in T \quad (26)$$

$$a_{ik}^t \leq l_i y_{ik}^t, \forall i \in N, \forall k \in K, \forall t \in T \quad (27)$$

$$x_{ijk}^t \in \{0,1\}, \forall i, j \in N, \forall k \in K, \forall t \in T \quad (28)$$

$$y_{ik}^t \in \{0,1\}, \forall i \in N, \forall k \in K, \forall t \in T \quad (29)$$

$$I_i^t \geq 0, \forall i \in C, \forall t \in T \quad (30)$$

$$w_{ik}^t \geq 0, \forall i \in C, \forall k \in K, \forall t \in T \quad (31)$$

$$a_{ik}^t \geq 0, \forall i \in C, \forall k \in K, \forall t \in T \quad (32)$$

$$z_k^t \leq \sum_{i \in C} y_{ik}^t \ \forall k \in K, \forall t \in T \quad (33)$$

$$z_k^t n \geq \sum_{i \in C} y_{ik}^t \ \forall k \in K, \forall t \in T \quad (34)$$

The total cost includes only the transportation costs as depicted in the objective function (15). This case corresponds to an environment in which the transportation cost represents the major cost component (e.g., the supplier and the customers represent entities of one and the same company). Constraints (16) indicate that each customer $i \in C$ has an initial inventory level equal to his maximum inventory level. Constraints (17) are the inventory balance equations for the customers. Constraints

(18) limit the total amount of inventory to $U_i, \forall i \in C$. Constraints (19), (33) and (34) ensure that the vehicle capacities are not exceeded on any day $t \in T$ during the planning horizon. Constraints (20)-(24) impose that each customer is visited exactly once, that m vehicles leave the depot, and that the same vehicle enters and leaves a given customer. Constraints (25) ensure feasibility in terms of the time necessary when traveling from node i to node j $\forall i, j \in N$. In addition, ensure simultaneously the elimination of subtours where M is a large constant. Constraints (26) and (27) impose that service may only start within the given interval $[e_i, l_i] \forall i \in N$. Constraints (28)-(32) are the domain constraints.

4.2. Solution Approach for the IRPTW

Due to the NP-hard nature of the IRPTW, a two-phase solution algorithm based on (a) a simple simulation and (b) the Variable Neighborhood Search Algorithm (VNS) is proposed to handle the problem. The first phase (Phase I) is related to the planning phase of the IRPTW, in which delivery times and quantities are determined by implementing the well-known inventory policy (s,S) for inventory management using a simple simulation. In the second phase (Phase II), the VNS is applied to combine the customers into the vehicle routes by solving a VRPTW for a specific time period during the planning horizon.

In particular, (s,S) inventory policy reflects the OU policy, where s and S correspond to a minimum and a maximum inventory level, respectively. An order for S-s units is placed immediately when the inventory level is reduced to s. Since stock-outs are not allowed, inventory policy (s_i, S_i) is applied to each customer $i \in C$ setting $s_i = 0 \; \forall i \in C$. In addition, each customer has an initial inventory level equal to his maximum inventory capacity $U_i \; \forall i \in C$. At the end of the planning horizon, each customer should have an inventory level equal to his initial inventory level.

Since the demands are fully available to the supplier at the beginning of the planning horizon, by applying an (s,S) inventory policy to each customer, Phase I of the algorithm enables the supplier to run an inventory simulation to determine delivery times and quantities, so that stock-outs are avoided. A sample problem of a distribution system that comprises a single supplier and twenty-five customers can be considered to explain the inventory simulation algorithm (Table 9).

Table 9

IRPTW sample problem

Customer	**Demand**						**(s,S) Inventory Policy**	
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**s**	**S**
1	11	16	9	7	9	9	0	20
2	24	34	36	33	32	30	0	60

3	8	14	7	11	11	13	0	20
4	10	14	9	7	11	5	0	20
5	13	14	9	5	10	9	0	20
6	16	15	23	21	12	20	0	40
7	24	21	23	16	17	26	0	40
8	18	18	20	19	13	18	0	40
9	15	6	2	17	13	9	0	20
10	11	15	8	11	11	7	0	20
11	8	5	9	10	9	12	0	20
12	10	27	21	26	17	13	0	40
13	15	23	27	19	30	34	0	60
14	8	11	10	8	14	16	0	20
15	39	36	42	40	39	39	0	80
16	42	38	40	49	42	34	0	80
17	14	19	14	10	8	16	0	40
18	15	23	27	25	18	20	0	40
19	13	10	12	8	8	11	0	20
20	6	9	8	16	12	9	0	20
21	25	21	17	28	15	19	0	40
22	13	21	23	21	17	15	0	40
23	9	15	9	13	10	14	0	20
24	9	16	12	15	11	12	0	20
25	34	47	57	43	42	39	0	80

Below, an algorithm (Algorithm 12) is presented that applies the (s,S) policy to customers. Initially, based on a specific test problem ($IRPTWdata$), the number of customers (NC) as well as the length of the planning horizon (H) are defined. Then, for each customer i, his (s_i, S_i) inventory policy and demands during the planning horizon (d) are taken into account to determine the delivery quantities and times ($deliveries$) as well as the inventory levels ($inventories$). It is worth noting that the time starts from zero, where customer demand is equal to zero and an initial inventory level exists for each customer. To define the delivery quantities the (s,S) policy is applied to each customer. Analytically, for each time period of the planning horizon, if the inventory level (IL) is less than s_i, a delivered quantity (OQ) is defined such that the maximum inventory level at the customer is reached. To define the inventory levels, the inventory balance equation is applied. Namely, the amount of inventory in the next time period must be equal to the current inventory plus the amount of delivered quantity minus the demand in the next time period.

Algorithm 12. *Simple simulation (Phase I)*

Inputs: $IRPTWdata$

$NC \leftarrow getNoCustomers(IRPTWdata)$

$H \leftarrow getPlanningHorizon(IRPTWdata)$

$Inventories \leftarrow []$

$Deliveries \leftarrow []$

$\boldsymbol{for}\ i = 1{:}\ NC\ \boldsymbol{do}$

$s_i \leftarrow gets_i(IRPTWdata)$

$S_i \leftarrow getS_i(IRPTWdata)$

$d \leftarrow [0, d_i^1, d_i^2, \dots, d_i^H]$

$IL(1) \leftarrow S_i$

$j \leftarrow 1$

$\boldsymbol{while}\ j \leq H\ \boldsymbol{do}$

$j \leftarrow j + 1$

$\boldsymbol{if}\ IL(j-1) < s_i\ \boldsymbol{then}$

$OQ(j-1) \leftarrow S_i - IL(j-1)$

$\boldsymbol{else}$

$OQ(j-1) \leftarrow 0$

$\boldsymbol{end-if}$

$IL(j) \leftarrow IL(j-1) + OQ(j-1) - d(j)$

$\boldsymbol{end-while}$

$n \leftarrow j$

$OQ(n) \leftarrow S_i - IL(n)$

$IL(n) \leftarrow S_i$

$Deliveries \leftarrow [Deliveries; OQ]$

$Inventories \leftarrow [Inventories; IL]$

$\boldsymbol{end-for}$

Output: $Deliveries, Inventories$

Based on the above example, the following figures (Fig. 20, Fig. 21, Fig. 22 and Fig. 23) illustrate for each customer the relative inventory levels during the planning horizon and the delivery times.

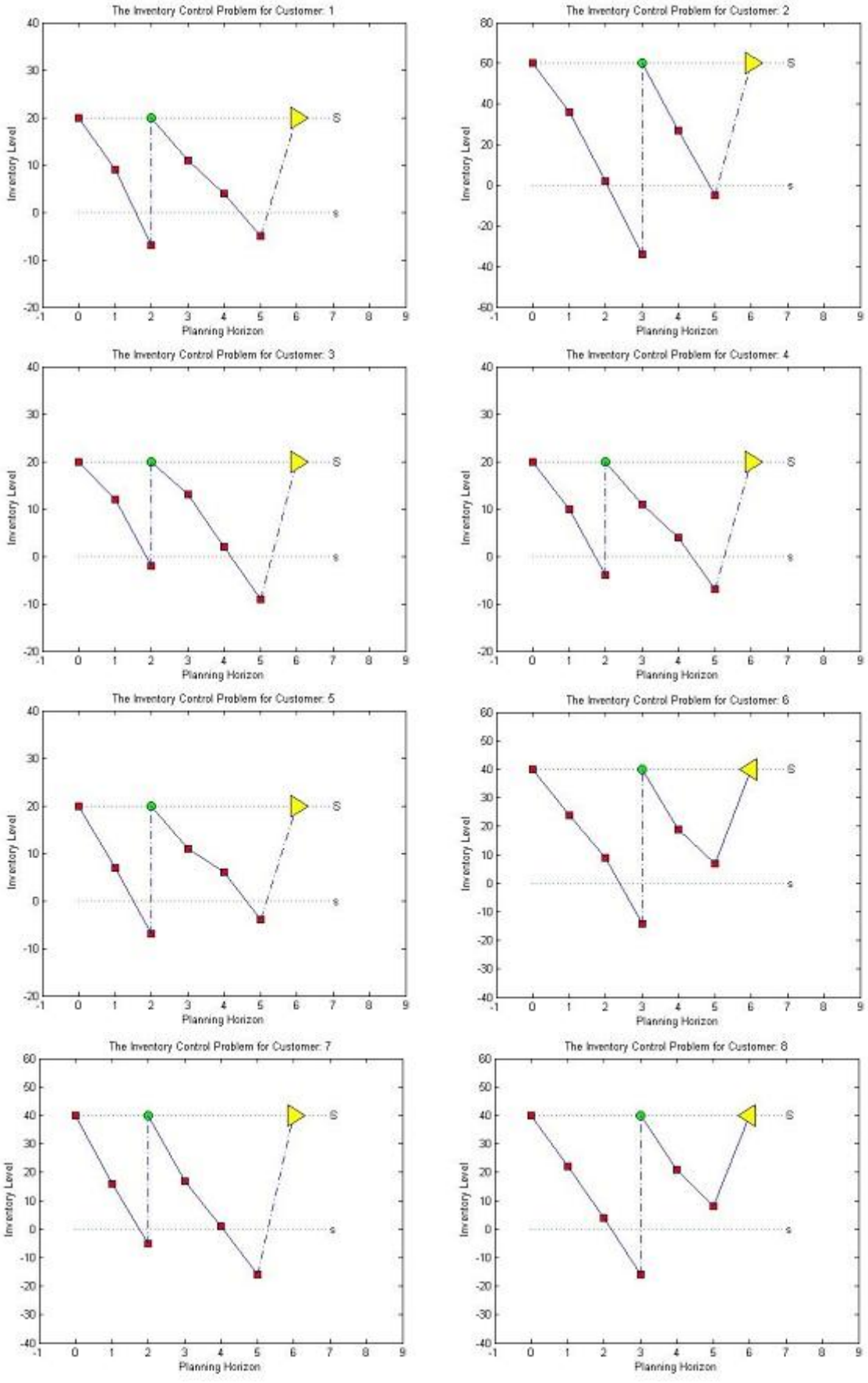


Fig. 20. Inventory simulation (Customer 1 – Customer 8)

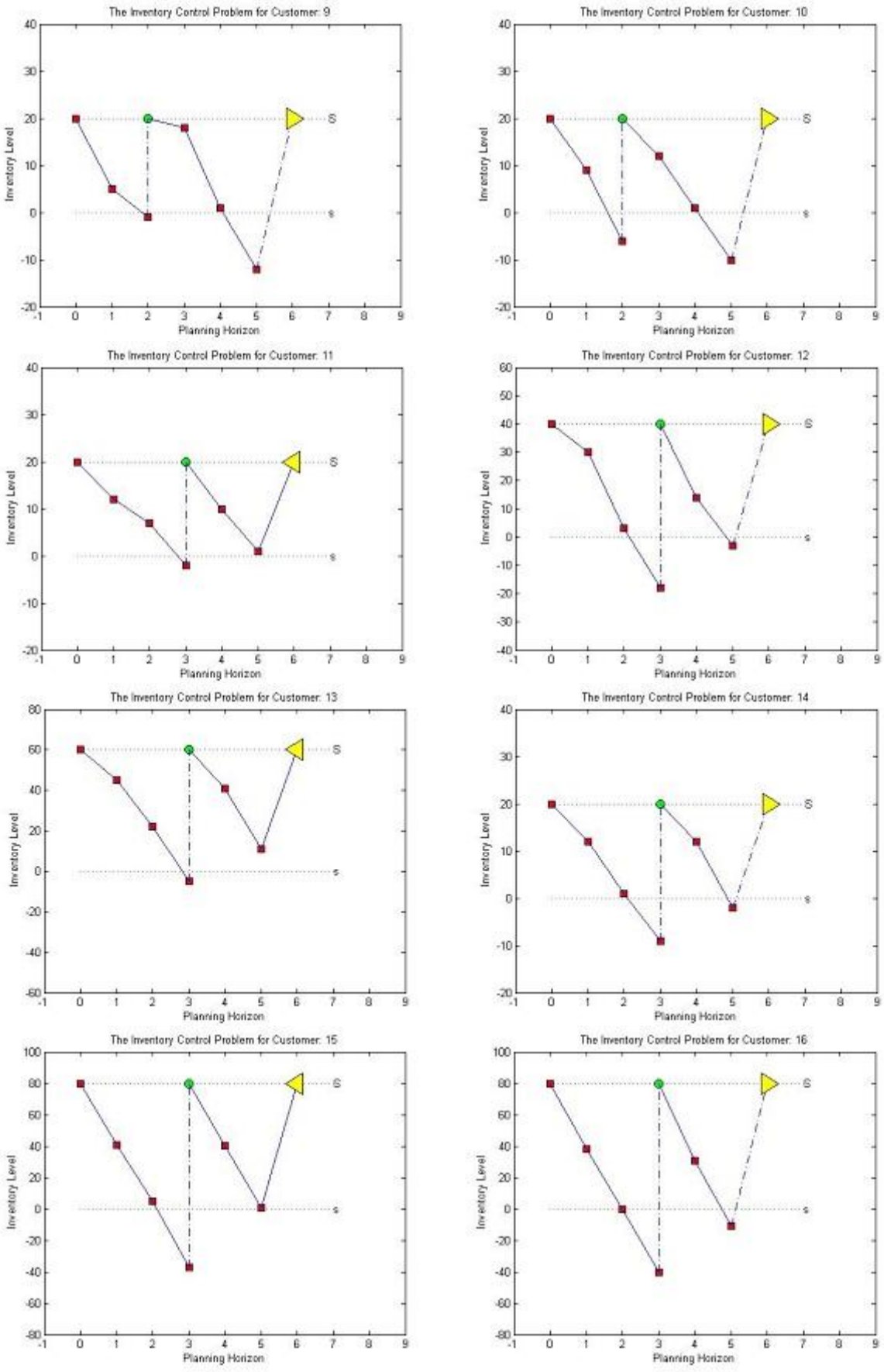


Fig. 21. Inventory simulation (Customer 9 – Customer 16)

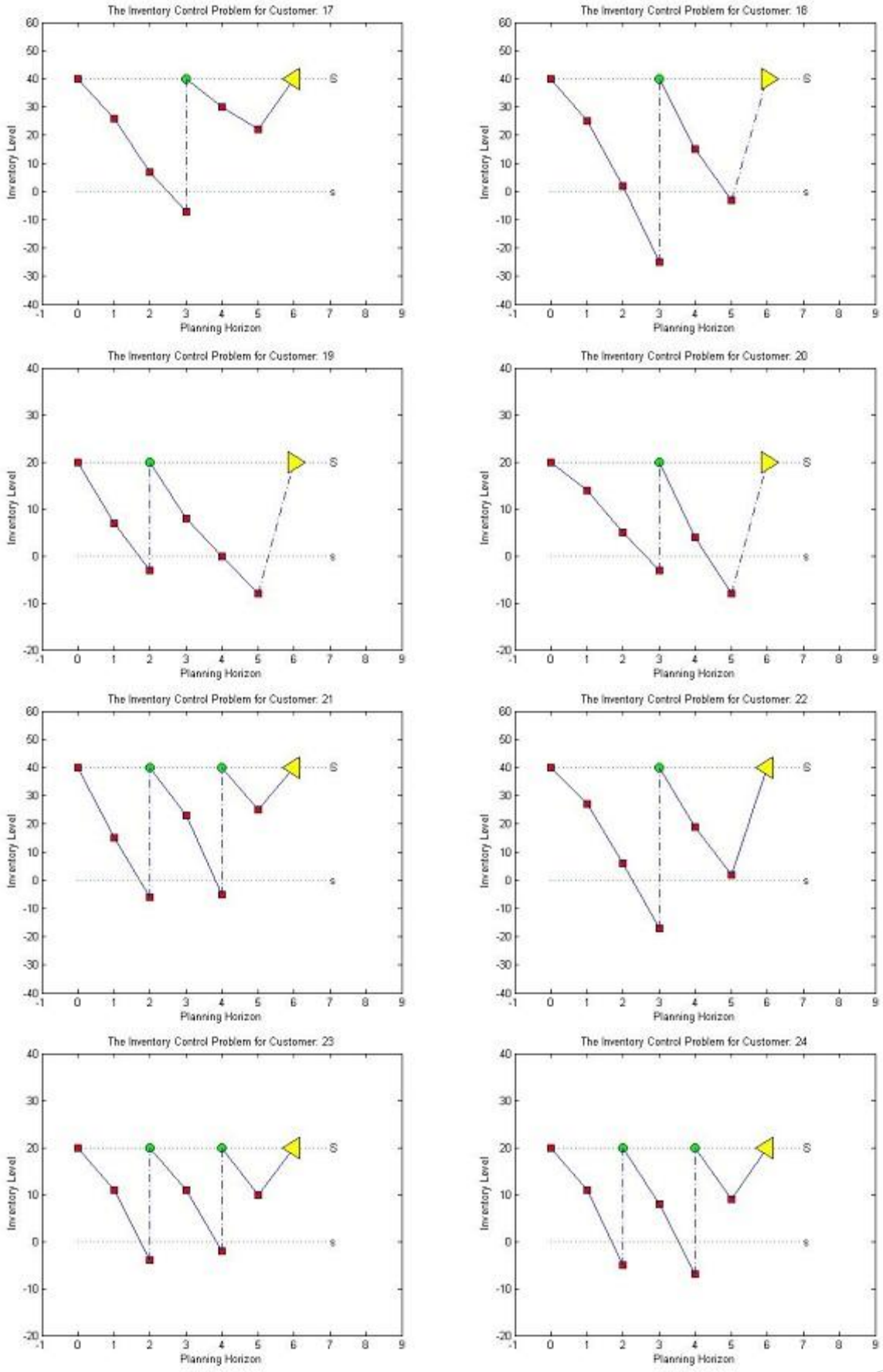


Fig. 22. Inventory simulation (Customer 17 – Customer 24)

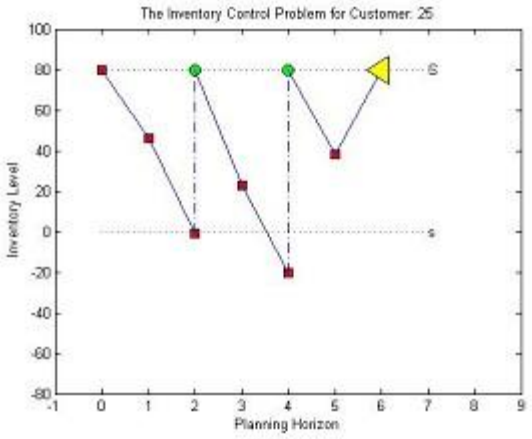


Fig. 23. Inventory simulation (Customer 25)

Since the simple simulation focuses only on the planning phase by determining the delivery times and quantities, the vehicle routes should be constructed. The routing phase (Phase II) is related to the usage of a Variable Neighborhood Search Algorithm (VNS) for solving a vehicle routing problem with time windows for each time period of the planning horizon where delivery quantities have been scheduled. The VNS is a single-point search meta-heuristic introduced by Mladenović and Hansen (1997). In the context of the algorithm, a set of neighborhood structures N_k where $k = 1,..,n$ are defined. The basic idea of the algorithm is to successively explore the set of pre-defined neighborhoods to provide a better solution. Each iteration of the algorithm is composed of three steps: *shaking*, *local search* and *move*. At each iteration, an initial solution is shaked from the current neighborhood N_k. For example, a solution x' is generated randomly in the current neighborhood $N_k(x)$. The representation of a VRPTW solution follows the representation presented in Chapter 3, in the context of the Simulated Annealing algorithm. A local search procedure is applied to the solution x' to generate the solution x''. The evaluation of the solution is based on the cost function related to the proposed vehicle routes. Therefore, the current solution is replaced by the new local optima x'' if and only if a better solution has been found (i.e., $f(x'') < f(x)$). The same search procedure is thus restarted from the solution x'' in the first neighborhood N_k. If no better solution is found, the algorithm moves to the next neighborhood N_{k+1}, randomly generates a new solution in this neighborhood, and attempts to improve it.

The generation of the initial solution is based on the Push Forward Insertion Heuristic (PFIH) (Solomon, 1987; Tan et al., 2001). The method tries to insert the customer between all the arcs in the current route. It selects the arc that has the lowest additional insertion cost. In addition, the feasibility check tests all the constraints related to time windows and vehicle capacity. When the current route is full of feasible insertions, PFIH will start a new route and repeat the procedure until all the customers are routed. As far as the first step of the VNS (shaking) is concerned, the 2-interchange neighborhood operator of Osman (1993) as well as the CROSS-exchange neighborhood operator of Taillard et al. (1997) are used randomly (rand2interchange

and randCrossExchange). Regarding the second step of the VNS (local search), nested neighborhoods are used based on the 2-interchange and CROSS-exchange mechanisms. These mechanisms (twoInterchange and crossExchange) are now used systematically (not randomly). In general terms, the 2-interchange mechanism is based on customer interchange between sets of vehicles routes. The 2 means that maximum two customer nodes may be interchanged between routes. The CROSS-exchange mechanism swaps sequences of consecutive customers between two routes. The detail information about PFIH, 2-interchange and CROSS-exchange can be obtained from papers of Solomon (1987), Osman (1993) and Taillard et al. (1997), respectively. Algorithm 13 presents the template of the proposed VNS algorithm.

Algorithm 13. *Variable neighborhood search algorithm (Phase II)*

Input: $deliveries, IRPTWdata$

$[H, vehicleCapacity] \leftarrow getPlanningHorizonAndVehCapacity(IRPTWdata), i \leftarrow 1$

while $i \leq H$ ***do***

$VRPTW \leftarrow createVRPTWproblem(deliveries, i)$

$x \leftarrow PFIH(VRPTW, vehicleCapacity)$

Repeat

$k \leftarrow 1$

while $k \leq 2$ ***do***

if $k = 1$ ***then***

$x' \leftarrow rand2interchange(x, vehicleCapacity)$

end − if

if $k = 2$ ***then***

$x' \leftarrow randCrossExchange(x, vehicleCapacity)$

end − if

$l \leftarrow 1, improvement \leftarrow false$

while $l \leq 2$ ***do***

if $l = 1$ ***then***

$x'' \leftarrow twoInterchange(x', vehicleCapacity)$

end − if

if $l = 2$ ***then***

$x'' \leftarrow crossExchange(x', vehicleCapacity)$

end − if

if $f(x'') < f(x')$ ***then***

$x' \leftarrow x'', l \leftarrow 1, improvement \leftarrow true$

else

$l \leftarrow l + 1$

end − if

end − while

if $improvement = true$ ***then***

$x \leftarrow x'', k \leftarrow 1$

else

$k \leftarrow k + 1$

end − if

end − while

Until *stoping criteria*

$i \leftarrow i + 1$

end − while

Output: *best found solution*

Based on the above example, the following figure (Fig. 24) illustrates the IRPTW solution.

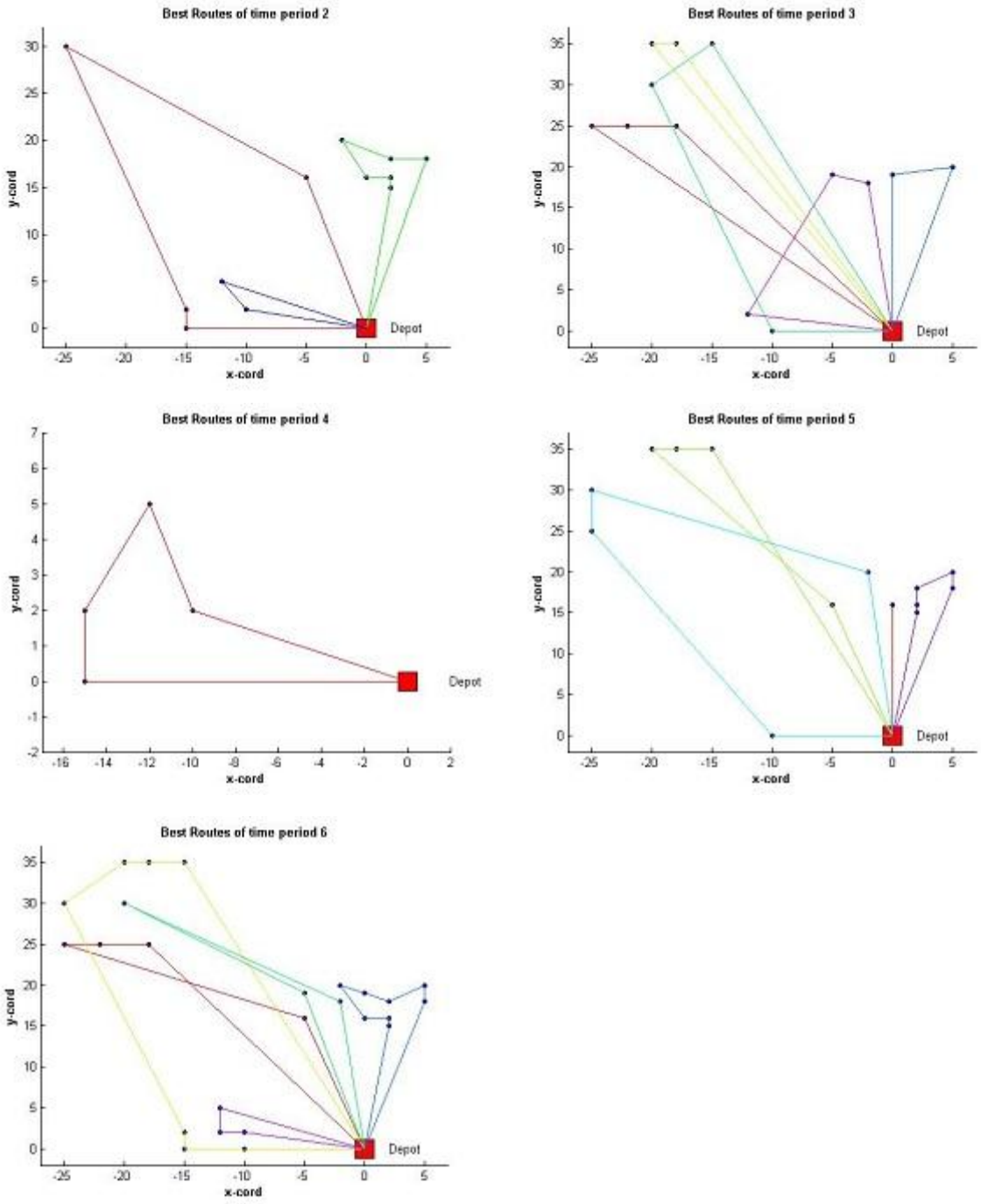


Fig. 24. IRPTW solution for the IRPTW sample problem

Table 10 presents the routes that take place in each time period of the planning horizon.

Table 10

Cost information and routes for the IRPTW sample problem

IRPTW Solution		
Routes of Period 1	**Routes of Period 2**	**Routes of Period 3**
	Route 1: 0-24-25-19-10-0	Route 1: 0-13-17-18-0
	Route 2: 0-5-3-7-9-4-1-0	Route 2: 0-16-14-0
	Route 3: 0-23-21-0	Route 3: 0-20-15-12-0
		Route 4: 0-6-2-0
		Route 5: 0-8-11-22-0
VRPTW Cost = 0	VRPTW Cost = 163.4705	VRPTW Cost = 338.1404
Routes of Period 4	**Routes of Period 5**	**Routes of Period 6**
Route 1: 0-24-25-23-21-0	Route 1: 0-7-0	Route 1: 0-13-17-18-10-0
	Route 2: 0-10-16-14-12-0	Route 2: 0-20-24-25-19-16-14-12-0
	Route 3: 0-20-18-19-9-0	Route 3: 0-8-15-11-0
	Route 4: 0-5-3-4-2-1-0	Route 4: 0-5-3-7-9-6-4-2-1-0
		Route 5: 0-23-22-21-0
VRPTW Cost = 35.0462	VRPTW Cost = 247.8036	VRPTW Cost = 330.9369
Total VRPTW Cost		
1115.3975		

4.3. Computational Experiments and Results

This section presents the computational results of the proposed two-phase solution algorithm. The algorithm was developed in the MATLAB programming language and executed on a DELL personal computer with an Intel® Core™ i3-2120, clocked at 3.30 GHz, a microprocessor with 4 GB of RAM memory under the operating system Microsoft Windows 7 Professional. Since new benchmark instances were designed, the efficiency and the effectiveness of the proposed algorithm cannot be compared to other published IRPTW studies using benchmark instances previously introduced. This is due to the differentiated manner in which the proposed algorithm operates based on the assumptions presented in Section 4.1. However, this section validates the two-phase solution algorithm and then evaluates its performance by comparing the algorithm's solutions with solutions obtained by solving a VRPTW problem for each time period of the planning horizon based on the known demands (the planning phase is ignored). The algorithm has been tested on a newly introduced set of 18 IRPTW benchmark instances described in the following.

All benchmark instances and their computational results are available at http://www.msl.aueb.gr/files/SimVnsIRPTW.zip.

The new datasets have been developed by generalizing the well-known datasets C101, C201, R101, R201, RC101 and RC201 of Solomon (1987), http://web.cba.neu.edu/~msolomon/problems.htm. As a result, these datasets are divided into six classes. The datasets are named in the form of "IRPTW_**Z**_n**X**_p**Y**" strings, where "**Z**" stands for the class related to a specific dataset of Solomon (1987), i.e., C101, C201, R101, R201, RC101 and RC201, "**X**" stands for the number of customers and "**Y**" stands for the number of time periods. For instance, the problem IRPTW_C101_n25_p6 represents a test problem of the first class (i.e., dataset that was generated by the dataset C101 of Solomon (1987)) with 25 customers and a planning horizon of 6 days. Different problem sizes, based on the total number of customers, were designed, in each class. Specifically, each class contains problems with 25, 50 and 100 customers. Nodes coordinates are modified in such a way that the depot is located at the origin (i.e., coordinates $(0,0)$). The distance matrix is obtained by calculating the Euclidean distances. Time windows related to customers as well as the maximum operation time for each vehicle are kept the same as in the Solomon's datasets.

Demand exists for each customer at each time period of the planning horizon. Customer demand at each time period was generated according to the Poisson distribution, $Poisson(\lambda)$, where λ is the rate parameter. For each customer, the rate parameter is equal to his demand in the single-period VRPTW problem of Solomon (1987). In addition, for each customer $i \in C$, his maximum inventory capacity is defined as $U_i = 2\lambda_i$. As it usually happens in real life, customers with higher expected demands will have higher inventory capacities. Therefore, for each customer $i \in C$, inventory policy (s_i, S_i) is equal to $(0, U_i)$. An unlimited fleet of identical vehicles with capacity Q is available for the distribution of the product. The vehicle capacity is kept the same as in the Solomon's datasets. At the beginning of the planning horizon, each customer $i \in C$ has an initial inventory level up to his maximum inventory capacity, i.e., U_i. Finally, the supplier has a sufficient supply of products that can cover customers' demands throughout the planning horizon.

Since the algorithm cannot be compared to other published IRPTW studies, the best solution obtained from the proposed algorithm ($IRPTW$) is compared to the best solution obtained if the planning phase is ignored ($p-VRPTW$). In the aftermath of ignoring the planning phase, a VRPTW problem needs to be solved for each day of the planning horizon according to daily demand. The proposed Variable Neighborhood Search algorithm for the routing phase is then used to solve a daily VRPTW problem through the planning horizon. To compare the results, the following gap percentage formula is used:

$$Gap\ (\%) = (Sol_{IRPTW} - Sol_{p-VRPTW}) \times \frac{1}{Sol_{p-VRPTW}} \times 100$$

The $Sol_{p-VRPTW}$ corresponds to the solution obtained by solving the daily VRPTWs according to the known daily demands, while the Sol_{IRPTW} determines the solution

obtained by applying the proposed two-phase solution algorithm. Since the Sol_{IRPTW} is compared with the $Sol_{p-VRPTW}$, a positive gap means that the $Sol_{p-VRPTW}$ is outperformed. The computational results obtained are summarized in Table 11. For the $p-VRPTW$ and $IRPTW$ problems, the respective total vehicle routing cost is presented. In addition, for each $IRPTW$ a computation time (in seconds) needed to obtain a solution is presented, while the last column of the table shows the gap between the two problems reflecting the respective relative error.

Table 11

Experimental results

Instance	p-VRPTW	IRPTW		p-VRPTW – IRPTW
	Vehicle Routing Cost	Vehicle Routing Cost	Computation Time (seconds)	Gap (%)
IRPTW_C101_n25_p6	1150.8817	**1115.3975**	142.5091	-3.0832
IRPTW_C101_n50_p6	2559.8361	**2077.3767**	1.0827e+03	-18.8473
IRPTW_C101_n100_p6	5685.3165	**5067.128**	7.6578e+03	-10.8734
IRPTW_C201_n25_p6	1293.2554	**883.8036**	165.4300	-31.6606
IRPTW_C201_n50_p6	2753.3707	**1614.115**	2.1193e+03	-41.3768
IRPTW_C201_n100_p6	3883.0779	**2812.4137**	3.9252e+04	-27.5726
IRPTW_R101_n25_p6	3804.3657	**1880.0941**	68.9572	-50.5806
IRPTW_R101_n50_p6	6728.73	**3462.5182**	461.9215	-48.5413
IRPTW_R101_n100_p6	10388.3709	**5566.806**	4.6026e+03	-46.4131
IRPTW_R201_n25_p6	2795.4622	**1629.6064**	135.5557	-41.7053
IRPTW_R201_n50_p6	5106.1462	**2626.2472**	1.5377e+03	-48.5669
IRPTW_R201_n100_p6	7567.238	**3837.7514**	1.5402e+04	-49.2846
IRPTW_RC101_n25_p6	3257.2105	**1892.2402**	82.0641	-41.9061
IRPTW_RC101_n50_p6	5969.5287	**3907.7457**	556.1790	-34.5385
IRPTW_RC101_n100_p6	10783.6372	**6145.4136**	5.2140e+03	-43.0117
IRPTW_RC201_n25_p6	2549.0663	**1521.4596**	213.5986	-40.3131
IRPTW_RC201_n50_p6	4548.0065	**2878.2674**	2.2410e+03	-36.7136
IRPTW_RC201_n100_p6	8191.1021	**5053.3035**	1.2644e+04	-38.3074

Based on Table 11, it can be concluded that better solutions are obtained when the planning phase is considered. The ability of each customer to have storage enables a significant decrease in the vehicle routing cost, reducing the total number of routes during the planning horizon. As it can be observed, in all cases, the two-phase solution algorithm provides better solutions than the $p-VRPTW$, with gaps in the interval of -3.0832 percent to -50.5806 percent. The results indicate that if the inventory capacity of each customer is taken into account during the planning phase, better solutions can be obtained, significantly reducing the total vehicle routing cost and designating the importance of integrating supply chain activities.

To illustrate in more detail the behavior of the proposed algorithm, more information is presented about the vehicles (number of routes) used in each time period of the planning horizon in Table 12.

Table 12

Number of vehicles used during the planning horizon

Instance	p-VRPTW							IRPTW						
	P1	P2	P3	P4	P5	P6	No. of Routes	P1	P2	P3	P4	P5	P6	No. of Routes
IRPTW_C101_n25_p6	3	3	3	3	3	3	**18**	0	3	5	1	4	5	**18**
IRPTW_C101_n50_p6	7	6	5	6	6	6	36	0	7	6	4	6	9	**32**
IRPTW_C101_n100_p6	11	12	12	12	12	12	71	0	11	17	7	13	20	**68**
IRPTW_C201_n25_p6	2	2	2	2	2	2	12	0	1	2	1	1	2	**7**
IRPTW_C201_n50_p6	4	4	4	3	3	4	22	0	3	4	2	3	5	**17**
IRPTW_C201_n100_p6	5	5	4	5	4	6	29	0	5	4	3	5	7	**24**
IRPTW_R101_n25_p6	9	9	9	9	9	9	54	0	4	6	3	4	10	**27**
IRPTW_R101_n50_p6	14	14	13	14	14	14	83	1	8	7	5	9	13	**43**
IRPTW_R101_n100_p6	23	23	23	23	23	23	138	0	12	13	10	15	24	**74**
IRPTW_R201_n25_p6	4	4	4	4	4	4	24	0	3	2	2	4	4	**15**
IRPTW_R201_n50_p6	6	6	6	6	6	6	36	0	3	5	3	4	6	**21**
IRPTW_R201_n100_p6	9	9	8	10	9	9	54	1	6	6	3	4	9	**29**
IRPTW_RC101_n25_p6	6	6	6	6	6	6	36	0	3	6	1	6	6	**22**
IRPTW_RC101_n50_p6	10	10	10	10	10	10	60	0	7	10	5	8	12	**42**
IRPTW_RC101_n100_p6	20	18	19	20	17	19	113	0	11	16	5	14	20	**66**
IRPTW_RC201_n25_p6	4	4	4	4	4	4	24	0	3	2	2	2	4	**13**
IRPTW_RC201_n50_p6	6	6	5	5	5	5	32	0	4	2	4	4	5	**19**
IRPTW_RC201_n100_p6	10	10	11	10	11	10	62	0	7	7	4	6	9	**33**

Due to the fact that each customer has an initial inventory level equal to his maximum inventory capacity, in most cases no routes occur in period 1. However, for test problems “IRPTW_R101_n50_p6” and “IRPTW_R201_n100_p6” a single route takes place in order to satisfy the daily demand of some customers for whom their daily demands are greater than their maximum inventory capacity. Since stock-outs are not allowed, a route takes place to satisfy their demands. In addition, the number of routes is increased at the end of the planning horizon since the (s,S) inventory policy is applied for each customer. According to this policy, for each customer, the inventory level at the end of the planning horizon should be equal to the initial inventory level. On the other hand, in the context of the p-VRPTW, the number of vehicles is nearly the same, as a specific VRPTW problem should be solved on a daily basis.

4.4. Conclusions and Future Work

In this chapter, a two-phase solution algorithm was introduced to handle the IRPTW. The chapter gives more emphasis to how a simple simulation can be used in hybrid synthesis with a Variable Neighborhood Search algorithm (single-point search meta-heuristic) for the solution of the IRPTW. Particularly, the simple simulation is related to the planning phase of the IRPTW to determine the delivery times and quantities, while the Variable Neighborhood Search algorithm is associated with the routing phase to determine the routes. The algorithm has been tested on a newly introduced set of 18 IRPTW benchmark instances by comparing the algorithm's solutions with the solutions obtained by solving a VRPTW problem for each time period of the planning horizon based on known demand (the planning phase is ignored). The computational results show that the proposed algorithm is outperformed, simultaneously verifying the benefits obtained by the integration of the inventory and the vehicle routing decisions. Due to the myopic nature of the proposed algorithm, it is worth noting that the two-phase solution algorithm should be even further improved. To begin with, both simulation and VNS should be dealt with in an iterative way to define a re-optimization phase. In this case, (s,S) inventory policy can be initialized randomly and recalculated at each iteration of the solution algorithm. This can be obtained by applying a Discrete Event Monte Carlo Simulation for the planning phase of the problem. In terms of future research, the goals are to (a) extend and improve the proposed algorithm, (b) explore the algorithm behavior in other problems (instances), (c) take into account inventory holding costs of customers in the objective function and (d) focus on the development of other meta-heuristic approaches for the solution of the IRPTW.

Chapter 5

Conclusions

In this thesis, two solution algorithms were introduced to handle the IRP and the IRPTW, respectively. As far as the first solution approach is concerned, the thesis gives more emphasis to how a Genetic Algorithm (population-based search meta-heuristic) can be used in hybrid synthesis with a Simulated Annealing Algorithm (single-point search meta-heuristic) for the solution of the IRP. Particularly, the Genetic Algorithm is related to the planning phase of the hybrid approach to determine the delivery times and quantities, while the Simulated Annealing algorithm is associated with the routing phase to determine the routes of each individual of the population. Regarding the second solution approach, a two-phase solution algorithm was introduced to handle the IRPTW. The proposed approach combines a simple simulation for the planning phase with a Variable Neighborhood Search algorithm for the routing phase to solve the IRPTW. In both studies, stock-outs or lost sales are not allowed, and therefore no shortage costs or costs related to lost sales are included in the objective function. This is a characteristic that differentiates the proposed algorithms from other works most closely related to this thesis.

Both algorithms have been tested on a newly introduced set of IRP and IRPTW benchmark instances. The computational results show that the proposed algorithms are outperformed, simultaneously verifying the benefits obtained by the integration of the inventory and the vehicle routing decisions. However, both algorithms can be even further improved. In terms of future research, in the context of the hybrid evolutionary optimization algorithm, the goals are to (a) explore more deeply the parameters of the Genetic Algorithm and the Simulated Annealing Algorithm, (b) explore the algorithm behavior in other problems (instances) and (c) focus on the development of other meta-heuristic approaches for the solution of the IRP. Regarding the two-phase solution algorithm, both simulation and Variable Neighborhood Search algorithm should be dealt with in an iterative way to define a re-optimization phase. The future goals are to (a) extend and improve the proposed algorithm, (b) explore the algorithm behavior in other problems (instances), (c) take into account inventory holding costs of customers in the objective function and (d) focus on the development of other meta-heuristic approaches for the solution of the IRPTW (e.g., combining the Genetic Algorithm presented in Chapter 3 with the Variable Neighborhood Search Algorithm presented in Chapter 4). Finally, the proposed algorithms can be extended to complicated problems such as the Inventory Routing Problem with Soft Time Windows (IRPTW) as well as the Production Routing Problem (PRP) and its variations.

References

Abdelmaguid, T., & Dessouky, M. (2006). A genetic algorithm approach to the integrated inventory-distribution problem. *International Journal of Production Research 44*(21), 4445-4464.

Abdelmaguid, T., Dessouky, M., & Ordóñez, F. (2009). Heuristic approaches for the inventory-routing problem with backlogging. *Computers & Industrial Engineering 56*(4), 1519-1534.

Adulyasak, Y., Cordeau, J. F., & Jans, R. (2015). The production routing problem: a review of formulations and solution algorithms. *Computers & Operations Research 55*, 141-152.

Aghezzaf, E. H., Raa, B., & Van Landeghem, H. (2006). Modeling inventory routing problems in supply chains of high consumption products. *European Journal of Operational Research 169*(3), 1048-1063.

Agra, A., Christiansen, M., & Delgado, A. (2016a). Discrete time and continuous time formulations for a short sea inventory routing problem. *Optimization & Engineering*, doi:10.1007/s11081-016-9319-0.

Agra, A., Christiansen, M., Delgado, A., & Hvattum, L. M. (2015). A maritime inventory routing problem with stochastic sailing and port times. *Computers & Operations Research 61*, 18-30.

Agra, A., Christiansen, M., Delgado, A., & Simonetti, L. (2014). Hybrid heuristics for a short sea inventory routing problem. *European Journal of Operational Research 236*(3), 924-935.

Agra, A., Christiansen, M., Ivarsøy, K., Solhaug, I. E., & Tomasgard A. (2016b). Combined ship routing and inventory management in the salmon farming industry. *Annals of Operations Research*, doi:10.1007/s10479-015-2088-x.

Agra, A., Andersson, H., Christiansen, M., & Wolsey, L. (2013). A maritime inventory routing problem: discrete time formulations and valid inequalities. *Networks 62*(4), 297-314.

Aksen, D., Kaya, O., Sibel Salman , F., & Tüncel, O. (2014) An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem. *European Journal of Operational Research 239*(2), 413-426.

Andersson, H., Christiansen, M., & Desaulniers, G. (2016). A new decomposition algorithm for a liquefied natural gas inventory routing problem. *International Journal of Production Research 54*(2), 564-578.

Andersson, H., Hoff, A., Christiansen, M., Hasle, G., & Løkketangen, A. (2010). Industrial aspects and literature survey: combined inventory management and routing. *Computers & Operations Research 37*(9), 1515-1536.

Archetti, C., Bertazzi, L., Hertz, A., & Speranza, M. G. (2012). A hybrid heuristic for an inventory routing problem. *INFORMS Journal on Computing 24*(1), 101-116.

Archetti, C., Boland, N., & Speranza, M. G. (2014). A matheuristic for the multi-vehicle inventory routing problem. Department of Economics and Management, University of Brescia, Italy. Working paper number: WPDEM 2014/3.

Archetti, C., Speranza, M. G. (2016). The inventory routing problem: the value of integration. *International Transactions in Operational Research 23*(3), 393-407.

Archetti, C., & Speranza, M. G. (2013). A survey on matheuristics for routing problems. Department of Economics and Management, University of Brescia, Italy. Working paper number: WPDEM 2013/11.

Arram, A., Ayob, M., & Zakree, M. (2014). Comparative study of meta-heuristic approaches for solving traveling salesman problem. *Asian Journal of Applied Sciences 7*, 662-670.

Augerat, P., Belenguer, J. M., Benavent, E., Corberán, A., Naddef, D., & Rinaldi, G. (1998). Computational results with a branch-and-cut code for the capacitated vehicle routing problem. Institute for Systems Analysis and Computer Science, Roma, Italy. Research Report number: R.495.

Axsäter, S. (2006). *Inventory control*. (2nd ed.). New York: Springer.

Aziz, N. A. B., & Moin, N. H. (2007). Genetic algorithm based approach for the multi product multi period inventory routing problem. In *Proceedings of 2007 IEEE international conference on industrial engineering and engineering management*, Singapore (pp. 1619-1623).

Ballou, R. H. (1989). Heuristics: rules of thumb for logistics decision making. *Journal of Business Logistics 10*(1), 122-132.

Bektas, T. (2006). The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega 34*(3), 209-219.

Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., Mack R. G, & Prutzman, P. J. (1983). Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces 13*(6), 4-23.

Bertazzi, L., Bosco, A., Guerriero, F., & Laganà, D. (2013). A stochastic inventory routing problem with stock-out. *Transportation Research Part C: Emerging Technologies 27*, 89-107.

Bertazzi, L., Bosco, A., & Laganà, D. (2015). Managing stochastic demand in an inventory routing problem with transportation procurement. *Omega 56*, 112-121.

Bertazzi, L., Savelsbergh, M., & Speranza, M. G. (2008). Inventory routing. In B. Golden, S. Raghavan, & E. Wasil (Eds.), *The vehicle routing problem: latest advances and new challenges* (pp. 49-72). New York: Springer.

Bertazzi, L., & Speranza, M. G. (2013). Inventory routing problems with multiple customers. *EURO Journal on Transportation and Logistics 2*(3), 255-275.

Bertazzi, L., & Speranza, M. G. (2012). Inventory routing problems: an introduction. *EURO Journal on Transportation and Logistics 1*(4), 307-326.

Bräysy, O., & Gendreau, M. (2005a). Vehicle routing problem with time windows, Part I: route construction and local search algorithms. *Transportation Science 39*(1), 104-118.

Bräysy, O., & Gendreau, M. (2005b). Vehicle routing problem with time windows, Part II: metaheuristics. *Transportation Science 39*(1), 119-139.

Campbell, A. M., & Savelsbergh, M. (2004). A decomposition approach for the inventory-routing problem. *Transportation Science 38*(4), 488-502.

Cho, W. D., Lee, Y. H., Lee, Y. T., & Gen, M. (2013). An adaptive genetic algorithm for the time dependent inventory routing problem. *Journal of Intelligent Manufacturing 25*(5), 1025-1042.

Coelho, L. C., Cordeau, J. F., & Laporte, G. (2013). Thirty years of inventory routing. *Transportation Science 48*(1), 1-19.

Coelho, L., Cordeau, J. F., & Laporte G. (2012a). The inventory-routing problem with transhipment. *Computers & Operations Research 39*(11), 2537-2548.

Coelho, L., Cordeau, J. F., & Laporte, G. (2012b). Consistency in multi-vehicle inventory-routing. *Transportation Research Part C: Emerging Technologies 24*, 270-287.

Coelho, L. C., & Laporte G. (2013). The exact solution of several classes of inventory-routing problems. *Computers & Operations Research 40*(2), 558-565.

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J. Y., & Semet, F. (2002). A guide to vehicle routing heuristics. *The Journal of the Operational Research Society 53*(5), 512-522.

Croom, S., Romano, P., & Giannakis, M. (2000). Supply chain management: an analytical framework for critical literature review. *European Journal of Purchasing & Supply Management 6*(1), 67-83.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science 6*(1), 80-91.

Diabat, A., Abdallah, T., & Le, T. (2016). A hybrid tabu search based heuristic for the periodic distribution inventory problem with perishable goods. *Annals of Operations Research 242*(2); 373-398.

Díaz-Parra, O., Ruiz-Vanoye, J., Loranca, B. B., Fuentes-Penna, A., & Barrera-Cámara, R. (2014). A survey of transportation problems. *Journal of Applied Mathematics*, http://dx.doi.org/10.1155/2014/848129.

Eksioglu, B., Vural, A. V., & Reisman, A. (2009). The vehicle routing problem: a taxonomic review. *Computers & Industrial Engineering 57*(4), 1472-1483.

El-Sherbeny, N. (2010). Vehicle routing with time windows: an overview of exact, heuristic and metaheuristic methods. *Journal of King Saud University 22*(3), 123-131.

Federgruen, A., Prastacos, G., & Zipkin, P. H. (1986). An allocation and distribution model for perishable products. *Operations Research 34*(1), 75-82.

Federgruen, A., & Zipkin, P. H. (1984). A combined vehicle-routing and inventory allocation problem. *Operations Research 32*(5), 1019-1037.

Flood, M. (1956). The traveling-salesman problem. *Operations Research 4*(1), 61-75.

Gaur, V., & Fisher, M. L. (2004). A periodic inventory routing problem at a supermarket chain. *Operations Research 52*(6), 813-822.

Ghiami, Y., Van Woensel, T., Christiansen, M., & Laporte G. (2015). A combined liquefied natural gas routing and deteriorating inventory management problem. In F. Corman, S. Voβ, & R. Negenborn (Eds.), *Computational Logistics* (pp. 91-104). Switzerland: Springer.

Goel, V., Furman, K., Song, J. H., & El-Bakry, A. (2012). Large neighborhood search for LNG inventory routing. *Journal of Heuristics 18*(6), 821-848.

Goel, V., Slusky, M., Van Hoeve, W. J., Furman, K. C., & Shao, Y. (2015). Constraint programming for LNG ship scheduling and inventory management. *European Journal of Operational Research 241*(3), 662-673.

Goetschalckx, M. (2011). *Supply chain engineering*. New York: Springer.

Griffis, S. E., Bell, J. E., & Closs, D. J. (2012). Metaheuristics in logistics and supply chain management. *Journal of Business Logistics 33*(2), 90-106.

Guemri, O., Bektar, A., Beldjilali, B., & Trentesaux, D. (2016). GRASP-based heuristic algorithm for the multi-vehicle inventory routing problem. *4OR 14*(4); 377-404.

Guerrero, W. J., Prodhon, C., Velasco, N., & Amaya, C. A. (2013). Hybrid heuristic for the inventory location-routing problem with deterministic demand. *International Journal of Production Economics 146*(1), 359-370.

Hemmati, A., Hvattum, L. M., Christiansen, M., & Laporte, G. (2016). An iterative two-phase hybrid matheuristic for a multi-product short sea inventory-routing problem. *European Journal of Operational Research 252*(3), 775-788.

Hemmati, A., Stålhane, M., Hvattum, L. M., & Andersson, H. (2015). An effective heuristic for solving a combined cargo and inventory routing problem in tramp shipping. *Computers & Operations Research 64*, 274-282.

Hewitt, M., Nemhauser, G., Savelsbergh, M., & Song, J. H. (2013). A branch-and-price guided search approach to maritime inventory routing. *Computers & Operations Research 40*(5), 1410-1419.

Holland, J. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Huang, S. H., & Lin, P. C. (2010). A modified ant colony optimization algorithm for multi-item inventory routing problems with demand uncertainty. *Transportation Research Part E: Logistics and Transportation Review 46*(5), 598-611.

Jiang, Y., & Grossmann, I. (2015). Alternative mixed-integer linear programming models of a maritime inventory routing problem. *Computers & Chemical Engineering 77*, 147-161.

Juan, A., Grasman, S, Caceres-Cruz J., & Bektaş T. (2014). A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs. *Simulation Modelling Practice and Theory 46*, 40-52.

Kirkpatrick, S., Gellat, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science 220*(4598), 671-680.

Laporte, G. (2010). A concise guide to the traveling salesman problem. *The Journal of the Operational Research Society 61*(1), 35-40.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science 43*(4), 408-416.

Lappas, P., Kritikos, M., & Ioannou, G. (2015a). A genetic algorithm for the inventory routing problem with time windows. *27th European conference on operational research*, Glasgow, United Kingdom.

Lappas, P., Kritikos, M., & Ioannou, G. (2015b). Classic metaheuristics and evolutionary optimization algorithms for routing problems: a computational study. *4th International Eurasian Conference on Mathematical Sciences and Applications*, Athens, Greece.

Lappas, P., Kritikos, M., & Ioannou, G. (2015c). Metaheuristic algorithms for routing problems: from the travelling salesman problem to the inventory routing problem. *MASSEE International Congress on Mathematics*, Athens, Greece.

Le, T., Diabat, A., Richard, J. P., & Yih Y. (2013). A column generation-based heuristic algorithm for an inventory routing problem with perishable goods. *Optimization Letters 7*(7), 1481-1502.

Li, K., Chen, B., Lyer Sivakumar, A., & Wu, Y. (2014). An inventory-routing problem with the objective of travel time minimization. *European Journal of Operational Research 236*(3), 936-945.

Li, Z., Jiang, C., & Jiang, L. (2015). An inventory routing problem with soft time windows. *12th International Symposium on Operations Research and its Application in Engineering, Technology and Management*, Luoyang, China.

Liao, L., Li, J., & Wu, Y. (2013). Modeling and optimization of inventory-distribution routing problem for agriculture products supply chain. *Discrete Dynamics in Nature and Society*, http://dx.doi.org/10.1155/2013/409869.

Liu, S. C., & Lee, W. T. (2011). A Heuristic method for the inventory routing problem with time windows. *Expert Systems with Applications 38*(10), 13223-13231.

Liu, S. C., Lu, M. C., & Chung, C. H. (2015). A hybrid heuristic method for the periodic inventory routing problem. *The International Journal of Advanced Manufacturing Technology 85*(9), 2345-2352.

Maniezzo, V., Stützle, T., & Voβ, S. (2009). *Matheuristics: hybridizing metaheuristics and mathematical programming*. New York: Springer.

Mercer, A., & Tao, X. (1996). Alternative inventory and distribution policies of a food manufacturer. *The Journal of the Operational Research Society 47*(6), 755-765.

Min, H., & Zhou, G. (2002). Supply chain modeling: past, present and future. *Computers & Industrial Engineering 43*(1-2), 231-249

Mirzaei, S., & Seifi, A. (2015). Considering lost sale in inventory routing problems for perishable goods. *Computers & Industrial Engineering 87*, 213-227.

Mjirda, A., Jarboui, B., Macedo, R., & Hanafi, S. (2012). A variable neighborhood search for the multi-product inventory routing problem. *Electronic Notes in Discrete Mathematics 39*, 91-98.

Mjirda, A., Jarboui, B., Macedo, R., Hanafi, S., & Mladenović , N. (2014). A two phase variable neighborhood search for the multi-product inventory routing problem. *Computers & Operations Research 52*, 291-299.

Mjirda, A., Jarboui, B., Mladenović, J., Wilbaut, C., & Hanafi, S. (2016). A general variable neighborhood search for the multi-product inventory routing problem. *IMA Journal of Management Mathematics 27*, 39-54.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research 24*(11): 1097-1100.

Moin, N. H., & Salhi, S. (2007). Inventory routing problems: a logistical overview. *The Journal of the Operational Research Society 58*(9), 1185-1194.

Moin, N. H., Salhi, S., & Aziz, N. A. B. (2011). An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem. *International Journal of Production Economics 133*(1), 334-343.

Nambirajan, R., Mendoza, A., Pazhani, S., Narendran, T. T., & Ganesh, K. (2016). CARE: heuristics for two-stage multi-product inventory routing problems with replenishments. *Computers & Industrial Engineering 97*, 41-57.

Niakan, F., & Rahimi, M. (2015). A multi-objective healthcare inventory routing problem; a fuzzy possibilistic approach. *Transportation Research Part E: Logistics and Transportation Review 80*, 74-94.

Nolz, P., Absi, N., & Feillet, D. (2014a). A stochastic inventory routing problem for infectious medical waste collection. *Networks 63*(1), 82-95.

Nolz, P., Absi, N., & Feillet, D. (2014b). A bi-objective inventory routing problem for sustainable waste management under uncertainty. *Journal of Multi-criteria Decision Analysis 21*(5-6), 299-314.

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems. *Annals of Operations Research 41*(4), 421-451.

Papageorgiou, D., Nemhauser, G., Sokol, J., Cheon, M. S., & Keha, A. (2014). MIRPLib – a library of maritime inventory routing problem instances: survey, core model, and benchmark results. *European Journal of Operational Research 235*(2), 350-366.

Park, Y. B., Yoo J. S., & Park, H. S. (2016). A genetic algorithm for the vendor-managed inventory routing problem with lost sales. *Expert Systems with Applications 53*, 149-159.

Popović, D., Vidović, M., & Radivojević, G. (2012). Variable neighborhood search heuristic for the inventory routing problem in fuel delivery. *Expert Systems with Applications 39*(18), 13390-13398.

Potvin, J. Y. (2009). Evolutionary algorithms for vehicle routing. *INFORMS Journal on Computing 21*(4), 518-548.

Qin, L., Miao, L., Ruan, Q., & Zhang, Y. (2014). A local search method for periodic inventory routing problem. *Expert Systems with Applications 41*(2), 765-778.

Raa B. (2015). Fleet optimization for cyclic inventory routing problems. *International Journal of Production Economics 160*, 172-181.

Rego, C., Gamboa, D., Glover, F., & Osterman, C. (2011). Traveling salesman problem heuristics: leading methods, implementations and latest advances. *European Journal of Operational Research 211*(3), 427-441.

Ronen, D. (1993). Ship scheduling: the last decade. *European Journal of Operational Research 71*(3), 325-333.

Santos, E., Satoru Ochi, L., Simonetti, L., & Henrique Gonsález, P. (2016). A hybrid heuristic based on iterated local search for multivehicle inventory routing problem. *Electronic Notes in Discrete Mathematics 52*, 197-204.

Schmid, V., Doerner, K., & Laporte, G. (2013). Rich routing problems arising in supply chain management. *European Journal of Operational Research 224*(3), 435-448.

Shao, Y., Furman, K., Goel, V., & Hoda, S. (2015). A hybrid heuristic strategy for liquefied natural gas inventory routing. *Transportation Research Part C: Emerging Technologies 53*, 151-171.

Shirokikh, V., & Zakharov, V. (2015). Dynamic adaptive large neighborhood search for inventory routing problem. In H. A. L. Thi, T. P. Dinh, & N. T. Nguyen (Eds.), *Modeling, computation and optimization in information systems and management sciences* (pp. 231-241). New York: Springer.

Shukla, N., Tiwari, M. K., & Ceglarek, D. (2013). Genetic-algorithms-based algorithm portfolio for inventory routing problem with stochastic demand. *International Journal of Production Research 51*(1), 118-137.

Simić, D., & Simić, S. (2013). Evolutionary approach in inventory routing problem. In I. Rojas, G. Joya, & J. Cabestany (Eds.), *Advances in computational intelligence* (pp. 395-403). New York: Springer.

Singh, T., Arbogast, J., & Neagu, N. (2015). An incremental approach using local-search heuristic for inventory routing problem in industrial gases. *Computers & Chemical Engineering 80*, 199-210.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research 35*(2), 254-265.

Song, J. H., & Furman, K. (2013). A maritime inventory routing problem: practical approach. *Computers & Operations Research 40*(3), 657-665.

Soysal, M., Bloemhof-Ruwaard, J., Haijema, R., & Van der Vorst, J. (2016). Modeling a green inventory routing problem for perishable products with horizontal collaboration. *Computers & Operations Research*, http://dx.doi.org/10.1016/j.cor.2016.02.003.

Soysal, M., Bloemhof-Ruwaard, J., Haijema, R., Van der Vorst, J. (2015). Modeling an inventory routing problem for perishable products with environmental considerations and demand uncertainty. *International Journal of Production Economics 164*, 118-133.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J. Y. (1997). A tabu search heuristic for the vehicle routing problem with time windows. *Transportation Science 31*(2), 170-186.

Talbi, E. G. (2009). *Metaheuristics: from design to implementation*. New Jersey: Willey.

Tan, K. C. (2001). A framework of supply chain management literature. *European Journal of Purchasing and Supply Management 7*(1), 39-48.

Tan, K. C., Lee, L. H., Zhu, Q. L., & Ou, K. (2001). Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering 15*, 281-295.

Tatsis, V., Parsopoulos, K., Skouri, K., & Konstantaras, I. (2013). An ant-based optimization approach for inventory routing. In M. Emmerich, A. Deutz, O. Schütze, T. Bäck, E. Tantar, A. Tantar, P. Del Moral, P. Legrand, P. Bouvry, & C. Coello (Eds.), *EVOLVE – A bridge between probability, set oriented numerics, and evolutionary computation IV* (pp. 107-121). New York: Springer.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. Philadephia: SIAM.

Vansteenwegen, P., & Mateo, M. (2014). An iterated local search algorithm for the single-vehicle cyclic inventory routing problem. *European Journal of Operational Research 237*(3), 802-813.

Yang, Z., Emmerich, M., Bäck, T., & Kok, J. (2015). Multicriteria inventory routing by cooperative swarms and evolutionary algorithms. In J. M. Ferrández Vicente, J. R. Álvarez-Sánchez, F. de la Paz López, J. Toledo-Moreo, & H. Adeli (Eds.), *Bioinspired computation in artificial systems* (pp. 127-137). New York: Springer.

Zeng, W., & Zhao, Q. (2010). Study of stochastic demand inventory routing problem with soft time windows based on MDP. In Z. Zeng, & J. Wang (Eds.), *Advances in neural network research and applications* (pp. 193-200). Shanghai: Springer.

Zhang, C., Nemhauser, G., Sokol, J., Cheon, M. S., & Keha, A. (2013). Flexible solutions to maritime inventory routing problems with delivery time windows. Georgia Institute of Technology, Technical Report.

Zhang, C., Nemhauser, G., Sokol, J., Cheon, M. S., & Papageorgiou, D. (2015). Robust inventory routing with flexible time window allocation. Georgia Institute of Technology, Technical Report.