INSTITUTO TECNOLOGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISION DE INGENIERIA Y ARQUITECTURA

PROGRAMA DE GRADUADOS EN CIENCIAS
DE LA INGENIERIA

REVERSE LOGISTICS MODELS AND ALGORITHMS: OPTIMIZING WASTE OF ELECTRIC AND ELECTRONIC EQUIPMENT RECOVERY SYSTEMS

THESIS
PRESENTED AS A REQUIREMENT
TO OBTAIN THE DEGREE OF:

DOCTOR OF PHILOSOPHY IN ENGINEERING
MAJOR IN INDUSTRIAL ENGINEERING

JULIO MAR ORTIZ

OCTOBER OF 2010

INSTITUTO TECNOLOGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISION DE INGENIERIA Y ARQUITECTURA

PROGRAMA DE GRADUADOS EN CIENCIAS
DE LA INGENIERIA

REVERSE LOGISTICS MODELS AND ALGORITHMS: OPTIMIZING WASTE OF ELECTRIC AND ELECTRONIC EQUIPMENT RECOVERY SYSTEMS

THESIS
PRESENTED AS A REQUIREMENT
TO OBTAIN THE DEGREE OF

DOCTOR OF PHILOSOPHY IN ENGINEERING
MAJOR IN INDUSTRIAL ENGINEERING

JULIO MAR ORTIZ

OCTOBER OF 2010

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY
CAMPUS MONTERREY

DIVISIÓN DE INGENIERÍA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN CIENCIAS DE LA INGENIERÍA

REVERSE LOGISTICS MODELS AND ALGORITHMS: OPTIMIZING WASTE OF ELECTRIC AND ELECTRONIC EQUIPMENT RECOVERY SYSTEMS

THESIS

PRESENTED AS A REQUIREMENT
TO OBTAIN THE DEGREE OF:

DOCTOR OF PHILOSOPHY IN ENGINEERING
MAJOR IN INDUSTRIAL ENGINEERING

JULIO MAR ORTIZ

OCTOBER OF 2010

Table of Contents

List of Tables

List of Figures

Dedicatory

To my lovely wife...

Lolis

... an amazing friend and partner.

To my dear daughter...

Alisson Itati

...for all the happiness and enthusiasm that provides to my life.

Acknowledgements

I wish to thank the Universidad Autónoma de Tamaulipas (UAT) and the Programa de Mejoramiento del Profesorado (PROMEP) for financial support through the UAT-265 scholarship. I wish to thank the UAT authorities for the support, especially to Dra. Teresa de Jesús Guzmán Acuña (institutional representative of PROMEP-UAT). Also I want to thank Dra. Carmen Zenia Nava Vera, and Dr. Julio César Barrientos Cisneros for their friendship and guidance.

I would like to thank my former Ph.D. advisors, Dr. José Luis Gónzalez-Velarde (Tecnológico de Monterrey) and Dr. Belarmino Adenso-Díaz (Universidad de Oviedo), for introducing me the problems, and for their invaluable guidance. I am also thankful to my dissertation committee members Dr. Neale Smith, Dra. Ada Álvarez, and Dr. Dagoberto Garza-Nuñez for many useful discussions, comments, and suggestions.

I would also like to express my special thanks to my wife, my daughter and my parents for their love and support. Without their understanding and encouragement, this dissertation would not have been possible. I want to express my gratitude to all my friends, especially to Mauricio Hincapié.

The third problem focuses on the study of robust and reconfigurable disassembly cell formation problems. This part of the research makes a significant contribution to the state of the art in Disassembly Systems Design and Cellular Manufacturing Systems for two reasons: first we consider two different approaches to deal with demand variability in disassembly systems: one reconfigurable and the other robust. In the reconfigurable approach the product demand varies form period to period in a deterministic manner; while in the robust approach the product demand varies in a random manner, however, this variation can be described in a number of probabilistic scenarios with a given occurrence probability. The problems are characterized and formulated as integer programming models. Second, given the complexity of the problems, we design a Variable Neighborhood Search (VNS) algorithm to efficiently solve them. The experimental analysis on a set of adapted (previously published) and randomly generated instances shows the good performance of the proposed algorithm.

Chapter 1

Introduction

1.1 Problem Background

This dissertation focuses on two topics: *Reverse Logistics* (RL) and *Waste of Electrical and Electronic Equipments* (WEEE). On one hand, during the last years RL has become a relevant topic not only for academics but also for practitioners. Companies are giving each day more and more importance to this field, because mainly of two reasons: the environmental issues and the impact of these issues on the public opinion, and the economic benefits that the company can obtain by the improvement of their return's processes. On the other hand, the strong increase of solid waste generation is a matter of concern in most industrialized countries. In Mexico the Ministry of Environment and Natural Resources (SEMARNAT) through the National Program for Prevention and Integrated Waste Management 2009 – 2012 has established as one of its main lines of action the design and implementation of Management Plans for the treatment of post consumer WEEE, which foresees the participation of all sectors involved under the principle of shared and differentiated responsibility. In Europe the new legislations based on the European Union Directive 2002/96/EC (EU, 2003) on WEEE have led to essential changes in the field of electronic scrap recycling. These situations make the interrelation between both subjects a wide and interesting field of research.

A reverse logistic system can be described as the logistic system that takes care of collection, selection, transportation, disassembly, recovery, disposal and re-distribution of discarded products. This offers a wide scope for the design and implementation of optimization models within the logistics – production – manufacturing interface. Thus in

literature are found contributions related to facility location problems, disassembly planning problems and inventory models that combine original and remanufactured products, just to mention a few. Also in previous research are found applications and case studies in various industries; for example in the automotive industry (Blanc *et al.*, 2006), the editorial industry (Soto, 2005), the recovery of construction wastes (Listes and Dekker, 2005), the recovery of WEEE (Shih, 2001), the tire remanufacturing industry (Strom, 1997), the recovery of returnable containers (Kroon and Vrijens, 1995) and the replacement of industrial equipment (Sharma, 2004).

In Europe, electronic waste is estimated to reach 12 million tons by 2015. This is the equivalent of approximately 14 kg per person per year (Goosey, 2004). In recent times, over 90% of the electronic waste still ends up in landfills, causing serious environmental problems. In response to this growing problem, the European Union Directive 2002/96/EC on WEEE has imposed to all member states the development of legislations based on the principle of extended producer responsibility to finance the treatment, recovery and recycling of all types of electronic waste. In Spain the directive came into effect on February 25 2005 by The Royal Decree 208/2005. The adoption of this legislation has led to essential changes in the field of electronic scrap recycling. Particularly, producers and importers of large and small electrical appliances are affected by the new normative, since they will have to manage over 75% of the total electronic waste. To cope with this new responsibility, companies have come together to create a number of *Collective Management Systems* (CMS) to be responsible for the operation of collecting and recycling systems.

Typically a CMS is a fund management committee created by industry associations representing the sector of manufacturers and importers of large and small appliances. Its responsibilities include: a) the establishment and management of the collection, transport, storage, recovery, treatment and control of WEEE and their packaging; b) to assist the maintenance and enhancement of natural and energy resources through the collection, treatment and management of WEEE, as well as improving health conditions of citizens and the environment, and c) the studies and research on the collection, transport, storage, recovery, treatment and control of such waste.

The collection process in which a CMS is involved can be described as follows (see Fig. 1.1): a CMS in particular establishes an agreement with a group of stores and collection points to collect the electronic waste from a given category (see Section 2.5 for the classification of WEEE categories provided by the European Directive), for example white goods like refrigerators and washing machines. Every store has a given capacity (directly

related to sales volume) to collect WEEE. In turn, every store has allotted in its warehouse a certain space for WEEE collected. Whenever this space is about to be saturated, the store manager calls to the coordination call center so that these items be taken off by the corresponding depot. Each depot holds a heterogeneous fleet of capacitated vehicles, which are used in the harvesting of WEEE. Their activities include the harvesting of WEEE from stores, consolidation at the depot, and in some cases value-adding activities such as sorting and packaging. The WEEE collected by the depots are then consolidated and shipped to the disassembly plants, where the returns are dismantled.

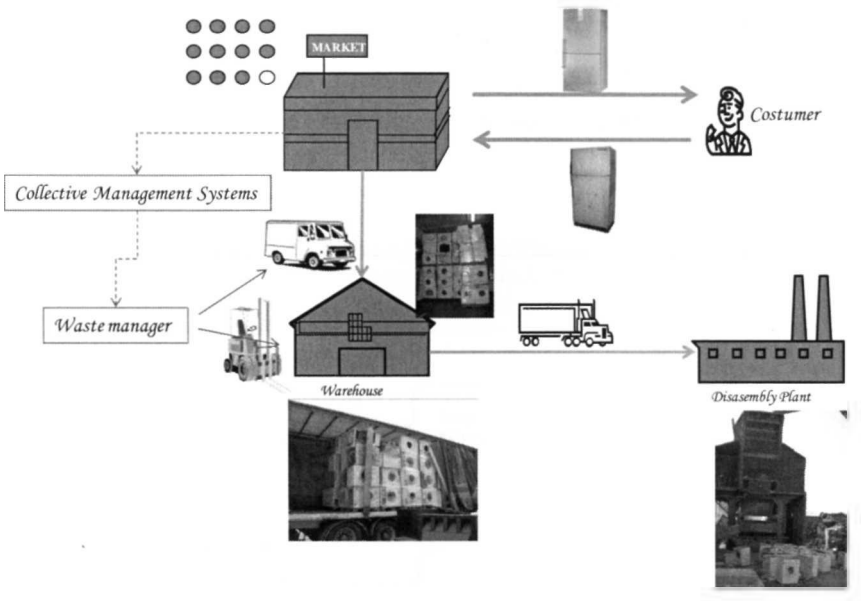


Figure 1.1 The WEEE collection process.

Many economic and environmental benefits can be achieved by electronic waste recycling, however to obtain these benefits an efficient reverse logistics system must be designed. From a logistical point of view, the design of a reverse logistic system has two main components: the location of collection depots and for each one of them, the design of the harvesting routes between its corresponding stores. Both problems fall into the reverse logistics management field. The first one aims to select which depots must be opened within a list of candidate sites, and to allocate collection points (stores) to depots. The second relates to the design of harvesting routes with capacitated vehicles. However, it should be noted that although the logistic function is an important activity, the product recovery aims to obtain useful components with economic value through reuse, remanufacturing or recycling, or to

remove hazardous components in compliance with safety regulations, and regardless the final destination for the collected products, disassembly operations are always required and must be optimized to improve the overall performance of the reverse logistics system.

1.2 Scope of the Research

The scope of this research is delineated in Fig 1.2 which provides a schematic view of the activities managed by a collective management system. This dissertation studies not only location and routing problems in WEEE reverse logistics, but also seeks to contribute to the development of the state of the art of disassembly systems, specifically to the design of disassembly cells. The aim is to embrace all the scope of incidence of the collective management systems.

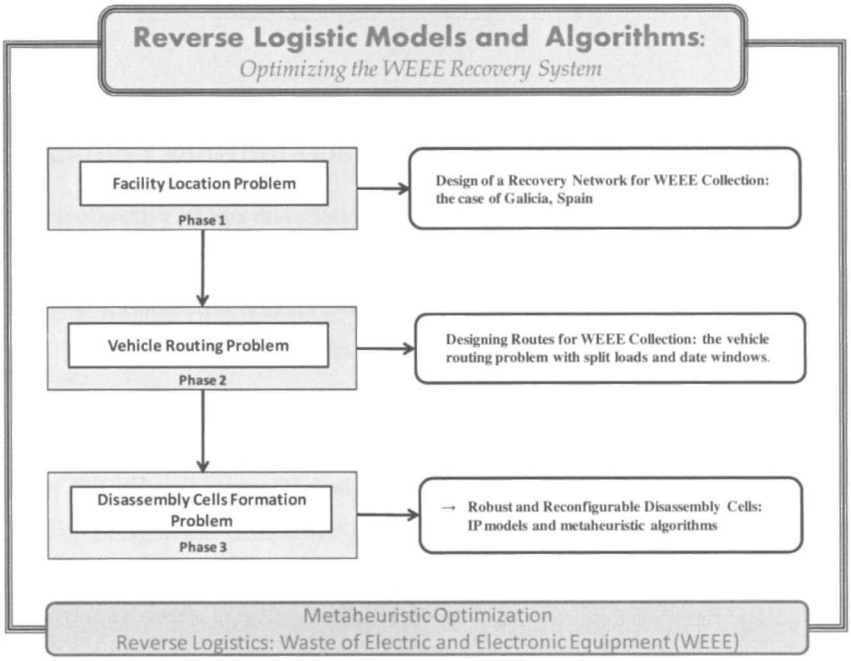


Figure 1.2 Scope of the dissertation.

Although both location and routing problems are interrelated, their analysis is often performed in a separate way, as we do in our research. Reverse logistics literature contains examples and case studies on each one individually, for example Shih (2001), Hu *et al.*

(2002), Listes and Dekker (2005), Lieckens and Vandaele (2007) and Aras *et al.* (2008) studied the location of collection depots, while Schultmann *et al.* (2005), Blanc *et al.* (2006), and Kim *et al.* (2009) analyzed the collection routing problem. On the other hand, though the cell formation problem has been one of the most studied in the manufacturing literature and there is a vast literature on disassembly systems (see for example Gungor and Gupta 2001, Tang *et al.* 2002), the contributions concerning disassembly cells are practically null.

In the literature we can found contributions dealing jointly with the location-routing problem (Min et al., 1998; and Nagy and Salhi, 2007), furthermore, we found a contribution which deals with the combined problem of jointly designing a cellular manufacturing into a supply chain network (Schaller, 2008). In fact, it is mathematically possible to design a model that combines location, routing and cellular manufacturing problems; however, coupled with the fact that its algorithmic solution would be highly complex, we do not see the usefulness of such approach in a WEEE recovery system. Therefore, the three problems are addressed independently. In later chapters we provide more arguments supporting our point of view.

1.3 Research Contributions

The research objective of this dissertation was:

"***To develop optimization models and solution algorithms for designing and improving reverse logistics systems for WEEE recovery***"

As stated above this research studies three problems named: 1) designing a recovery network for WEEE collection, 2) designing collection routes for WEEE, and 3) designing robust and reconfigurable disassembly cells. Therefore, there are particular objectives for each one, which are respectively stated below:

1. In the *network design problem* the objective is to redesign the logistics network of a Collective Management System which collects WEEE in the Spanish region of Galicia.
2. In the *WEEE collection routing problem* the objective is to solve a real-world problem of designing routes where a fixed and heterogeneous fleet of capacitated

vehicles with special features is used in the harvesting of WEEE from a set of customers.

3. The objective of the third problem is to design *robust and reconfigurable disassembly cells* to deal with two different kinds of demand variability. A Variable Neighbourhood Search (VNS) algorithm is proposed to solve them and we analyze for which one it is better suitable.

Given the complexity of the problems, the design of metaheuristic algorithms is required to solve real-world instances. For each problem a specific metaheuristic framework is selected based on its characteristics. The proposed metaheuristics can be used in real world to aid organizations improve their recovery systems. This was accomplished by demonstrating the validation of the algorithms with real world data provided by the ECOLEC foundation. The remaining of this section briefly discuses the contributions or each addressed problem.

1.3.1 Network Design for WEEE Collection

This part of the research aims to redesign the reverse logistic network of the Collective Management System –ECOLEC Foundation– for the collection of Waste of Electric and Electronic Equipment, in the Spanish region of Galicia. As a basis for our study a three-phase hierarchical approach is proposed. In the first phase a facility location problem is formulated and solved by means of a mixed integer linear programming; in the second phase a new integer programming formulation for the corresponding heterogeneous fleet vehicle routing problem is presented, and a savings-based heuristic algorithm is developed to efficiently solve the related collection routing problems; in the third phase a simulation study is performed on the collection routes in order to assess the overall performance of the recovery system. The results show a good performance of the proposed procedure, and an improved configuration of the recovery network compared to the one currently in use (particularly transportation costs are reduced by 29.2%).

The contribution of this case-oriented research focuses on the description of a methodological approach which combines three operational research techniques –integer programming, heuristic algorithms and simulation– to solve a real-world problem with substantial economic and environmental significance. The heuristic algorithm designed in this part of the research corresponds to a savings-based algorithm which was used to solve the corresponding vehicle routing problems in the recovering network of Galicia.

1.3.2 Vehicle Routing in WEEE Reverse Logistics

This part of the research focuses on the study of a Vehicle Routing Problem (VRP) for reverse logistics. It specifically addresses the problem of designing routes for the collection of WEEE. As will be shown in the literature review (see Chapter 2), though in recent years there has been an increasing interest towards so-called rich VRP models that include important issues arising from real-world applications, the literature on vehicle routing for reverse logistics is still very scarce. Most of the papers address specific problems and case studies, while theoretical contributions mainly deal with the VRP with simultaneous pick-up and delivery, or the VRP with backhauls.

This research makes a significant contribution to the state of the art in VRP and RL for two reasons: first we characterize and formulate as a mixed integer programming model a new variant of the VRP which includes important issues arising from real-world applications in WEEE recovery systems. Difficulty for this problem arises from the fact that it is characterized by four variants of the VRP that have been studied independently in the literature, but not together, including: (1) the use of a heterogeneous fleet of vehicles, (2) customers with high demand that may be split to fit the vehicle capacity, (3) the fact that a same vehicle may be assigned to more than one route, and (4) the agreement of a time interval for visiting customers. These characteristics make the problem complex and interesting, while complicating its classification within the standard VRP literature. We label this variant of the VRP as the Vehicle Routing Problem with Split Loads and Date Windows (VRP-SLDW). Second, given the complexity of the problem, we design a Greedy Randomized Adaptive Searching Procedure (GRASP) algorithm to solve it. The experimental analysis on a large set of randomly generated instances shows the good performance of the proposed algorithm. Moreover, computational results using real data show that the method outperforms real existing approaches to reverse logistics.

1.3.3 Robust and Reconfigurable Disassembly Cells

This part of the research focuses on the study of cell formation problems for disassembly operations in reverse logistics. It specifically addresses the problem of designing robust and reconfigurable disassembly cells to deal with demand variability typical of this problem. Even though the performance of a cellular manufacturing system and particularly of a disassembly system is highly dependent on the accuracy of the input data and it may rapidly deteriorate if demand and product quality changes, most of the current methods designed for

the manufacturing cell formation problem have been developed for a single-period planning horizon, which assume that problem data are constant for the entire planning horizon. Furthermore, given the practical differences between manufacturing systems and disassembly systems, typical cell formation problem formulations are not suitable for the Disassembly Cell Formation Problem (DCFP). In manufacturing systems, all operations belonging to a product are required by every product of the same type, while in the disassembly process not all operations for a given product type are required by every product. For instance, a computer unit could arrive to the disassembly shop without the hard disk drive. On the other hand, though the cell formation problem has been one of the most studied in the manufacturing literature and there is a vast literature on disassembly systems, the contributions concerning disassembly cells are practically null. Most studies in disassembly systems designs fit into three major areas: 1) modeling and representation of product disassembly sequences, 2) disassembly process planning, which includes the extent to which disassembly should be performed and how to decide the optimal disassembly sequence, and 3) disassembly system design and line balancing.

This research makes a significant contribution to the state of the art in Disassembly Systems Design and Cellular Manufacturing Systems for two reasons: first we consider two different approaches to deal with the DCFP with demand variability: one reconfigurable and the other robust. In the reconfigurable approach the product demand varies form period to period in a deterministic manner; while in the robust approach the product demand varies in a random manner, however, this variation can be described in a number of probabilistic scenarios with a given occurrence probability. The problems are characterized and formulated as integer programming models. Second, given the complexity of the problems, we design a Variable Neighborhood Search (VNS) algorithm to efficiently solve them. The experimental analysis on a set of adapted (previously published) and randomly generated instances shows the good performance of the proposed algorithm.

1.4 Outline

The remaining of this thesis is structured as follows: In Chapter 2 this dissertation reviews the essential literature published in order to acquaint the reader with the areas related to the research. It presents a literature review covering the various aspects of the studied problems. In particular this chapter provides a practical and concise introduction to reverse logistics and it summarizes current literature in topics such as operation research models in reverse

logistics, including network design problems, vehicle routing problems and cell formation problems. Chapter 3 states the research problems and outlines the solution approaches, while Chapter 4 formulates the mathematical models and shows its complexity. Chapter 5 describes the solution algorithms. Chapter 6 describes the experiments performed to validate the proposed algorithms, and discusses and analyzes its results. Finally, Chapter 7 summarizes and presents the conclusion of this dissertation, and discusses significant directions for future research.

Chapter 2

Literature Review

This chapter provides the background of the literature reviewed in order to acquaint the reader with the areas related to the research. In particular this chapter provides a practical and concise introduction to Reverse Logistics. It reviews past and current developments of Operations Research models in Reverse Logistics with special emphasis on location and network design issues, vehicle routing for reverse logistics and disassembly systems. Finally, a sample of Reverse Logistics related dissertations from 1995 to 2009 are described in a chronological order.

2.1 Introduction

Increased competition in the global market has forced companies to seek new ways to improve operations, increase profits, reduce costs, maximize productivity and attract more customers. This has led organizations to realize the need for agile, adaptable, and aligned supply chains to survive in a world where individual efficiency is not enough. An efficient Supply Chain Management (SCM) emphasize the importance of coordinating and integrating the key business processes of the involved organizations, thus avoiding local optimization and emphasizing integration (Krikke *et al.*, 2001a).

By the late eighties and early nineties, the organizational requirements were beyond their business process integration given the significance that the environmental issues got in business. This change motivated the managers of the organizations to face a paradox. On the one hand, the growing demand for new and better products to meet customer expectations for ever-lower prices, implied an increase of organizational productivity and an emphasis on economic aspects. On the other hand, the pressure of the consumers, who had a growing

environmental awareness, and the establishment of governmental laws demanded a reduction in their environmental impacts, even at the expense of productivity.

To address this paradox, managers began by considering that the task of logistics within the supply chain does not end with the delivery of finished products to final consumers. But from the standpoint of the product-life-cycle there should exist a parallel reverse flow perfectly managed to withdraw them at the time of becoming obsolete, damaged or for any other reason, in order to dispose of them in a place where a higher value could be obtained. This situation gives birth to the concept of *Reverse Logistics* (RL).

A RL network can be seen as a supply chain that is (re)designed to systematically manage the flow of parts and products destined for remanufacturing, recycling, or disposal activities. This enhanced supply chain is, therefore, capable of effectively using resources that were not previously considered or utilized (Dowlatshahi, 2000). Seitz (2004) stated that many product recovery strategies are based on regulations requiring companies to take responsibility for end-of-life or end-of-use products. The reality in many markets for remanufactured products, however, is based on potentially profitable business opportunities linked to the recovery of used products. This feature entails a competitive dimension to product recovery and RL activities.

Under a general classification, RL networks can be classified as: open loop supply chains or closed loop supply chains. If both markets coincide, we see a closed loop distribution-collection network; in contrast, if both markets are different, two open loop networks are formed (Fleischmann *et al.*, 2001). Typically, material recovery networks are open loop, while products or parts recovery networks are closed loops (Beamon and Fernandes, 2004).

While in USA the economic benefits of product recovery and recycling practices make Original Equipment Manufacturers (OEM) formally responsible for the recovery of their 'own' products after disposal, in Europe, new political policies aim at the closure of material flows, causing the same result: to reduce emissions and minimize residual waste. Particularly relevant for this dissertation is the European Union Directive 2002/96/EC (EU, 2003) on Waste of Electrical and Electronic Equipment (WEEE), whose general purpose is to prevent the creation of electrical and electronic waste and to promote reuse, recycling and the other forms of recovery in order to reduce final disposal. This directive has imposed to all member states the development of legislations based on the principle of extended producer responsibility to finance the treatment, recovery and recycling of all types of electronic waste. The adoption of this legislation has led to essential changes in the field of electronic scrap recycling. Particularly, producers and importers of large and small electrical appliances are

affected by the new normative, since they will have to manage over 75% of the total electronic waste. To cope with this new responsibility, companies have come together to create a number of Collective Management Systems (CMS) to be responsible for the operation of collecting and recycling systems.

One of the key problems CMS have to deal with as a result of WEEE directive is the set-up of an RL system for their discarded products. In the context of a WEEE collection system, an RL system can be described as the logistic system that takes care of collection, selection, transportation, disassembly, recuperation, disposal and re-distribution of discarded products. The aim is to recover maximal economical and ecological value at minimal economic costs while maintaining an accurate customer service level. Although this chapter reviews several optimization problems faced in the design of RL systems, the focus is on specific classes of logistics–manufacturing problems: *logistic network design*, *vehicle routing problems* and *cell formation problems*. These will be reviewed below.

This chapter has nine main sections. In Section 2.2 we review what literature says about reverse logistics. We will take a tour between the different existent definitions analyzing them, and pointing out the main findings in a historical perspective. In Section 2.3, we characterize RL systems, deriving a framework for both classifying RL networks and set a series of preliminary considerations to implement a recovery product program. In Section 2.4, we review published literature in RL and Operations Research (OR) with the aim to evaluate the actual state of development of the area. In Section 2.5, we review the EU directive on WEEE. The problems related to network design, vehicle routing problem and disassembly cell formation problems are reviewed in Sections 2.6 to 2.8. Finally, Section 2.9 review past dissertations on RL.

2.2 Reverse Logistics: definition and scope

There are several authors proposing a definition for reverse logistics. Even though RL was initially considered an extension of the traditional forward logistics, but in the reverse (opposite) direction, its concept and field of study have evolved over the last decade. Our purpose in this section is not to develop new concepts or theories about reverse logistics, but to provide a brief summary of the principal statements found in the literature.

In the early nineties, the first formal definition of RL was proposed by the Council of Logistics Management (Stock, 1992); stressing the role of all the logistics activities related to

recycling, waste disposal, and management of hazardous waste. In the same year, Phlen and Farris (1992) guided by a *marketing principles perspective* defined RL as "the movement of goods from the consumer to the producer in a distribution channel". By the mid-nineties Kroon *et al.* (1995) took up again a purely logistical guidance, defining RL as "all the logistics activities related to the management, reduction, reverse distribution and disposal of end-of-life items", which creates a flow in the opposite direction to forward logistic activities.

Considerably advancing toward the establishment of a formal understanding of the RL concept, Carter and Ellram (1998) offered a definition with a focus on business efficiency, defining RL as "the process by which organizations can become more environmentally efficient through the reuse and recycling of materials". This later definition allowed that the economic benefits of recycling and reuse were considered on designing RL systems. Despite the progress made in its conception, the studies of these authors in search of a comprehensive theory, led them to conclude that the available literature was not enough to establish a formal understanding of RL. However, Dowlatshahi (2000) closed the gap described by Carter and Ellarm to describe the strategic and operational factors in RL systems. Gradually the RL field was able to attract the attention from organizations, practitioners, academics and researchers who wanted build a formal understanding of RL.

Several definitions arose to the light. The emphasis was toward the systematic recovery of product or parts previously embarked that for some reason required to be reprocessed by a previous member in the supply chain (Dowlatshahi, 2002). One of the most representative contributions can be found in the work of Rogers *et al.* (1998) who not only emphasized the logistic process of RL, but also considered the concept of *value added activities*. They define RL as: "the process to plan, implement and control efficiently and cost effective the flow and storage of materials, in-process inventory, final goods and related information from the *point of consumption* to the *point of origin* in order to obtain value or to promote the proper disposal".

Summarizing this contribution, the Reverse Logistics Executive Council (RLEC), an emerging organization focused on the development of concepts and practices related to RL, defines RL as: "the process of moving goods from their typical final destination to another point with the aim of obtain value or to facilitate the proper disposal of products". On their own REVLOG, an European organization oriented to study the impact of RL practices in several industries, point out that RL refers to all the logistic activities related to the collection, disassembly and processing of used products, spare parts and/or materials to ensure a sustainable recovery (environmentally friendly). REVLOG defines RL as "the process of

planning, implementing and controlling flows of materials, in-process inventory and finished goods from the point of use to the point of collection or disposal". According to Brito *et al.* (2003a), the essence of the definition provided by Rogers *et al.* (1998) is maintained by the REVLOG, but the later do not relate to matters such as "point of use" or "point of origin", recognizing the existence of several types of returns by extending the scope for consideration of reverse flows of materials, and packaging of products whose life cycles are not finished.

In short, the definition of RL has changed over time, starting from an extension of forward logistics but in an opposite direction, to its emphasis on environmental and value added issues of the recovery process. Figure 2.1 describes a "rich picture" of the various activities involved in the planning and execution of the RL networks.

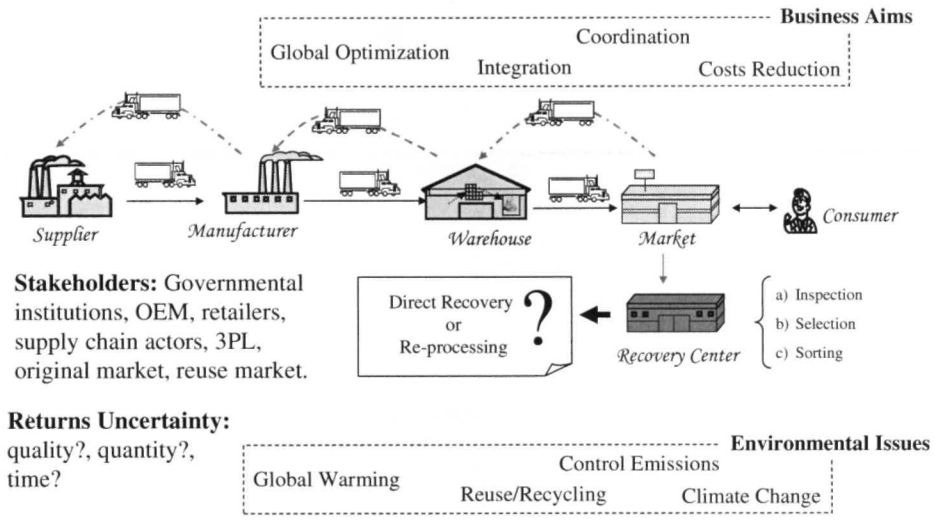


Figure 2.1 Rich picture of the strategic and operational activities of recovery networks.

RL appears to be a growing supply chain management topic for academic researchers, especially in the field of operations management and operation research. The available literature that provides in-depth treatments and analyses of sub-sets of RL is far less than short, practitioner-oriented, and conceptual-based literature in RL. In the literature, we can distinguish contributions mainly in quantitative models, case studies and theoretical developments underlying the theory in reverse logistics. Rubio *et al.* (2008) provided a description and analyses of the main characteristics of published literature in RL from 1995 to 2005 with more than 200 citations. The authors developed a database consisting of methodology and the techniques of analysis, as well as other relevant aspects of RL research.

In Guide and Van Wassenhove (2009) we can find a recount of the evolution of research in this growing area over the past 15 years, during which it developed from a narrow, technically focused niche area to a fully recognized subfield of supply chain management. An updated literature review of RL is found in Dowlatshahi (2010), which identified the present state of theory in RL regarding cost-benefit. Figure 2.2 describes some of the contributions that we consider of particular relevance to provide a historical perspective of the evolution of RL.

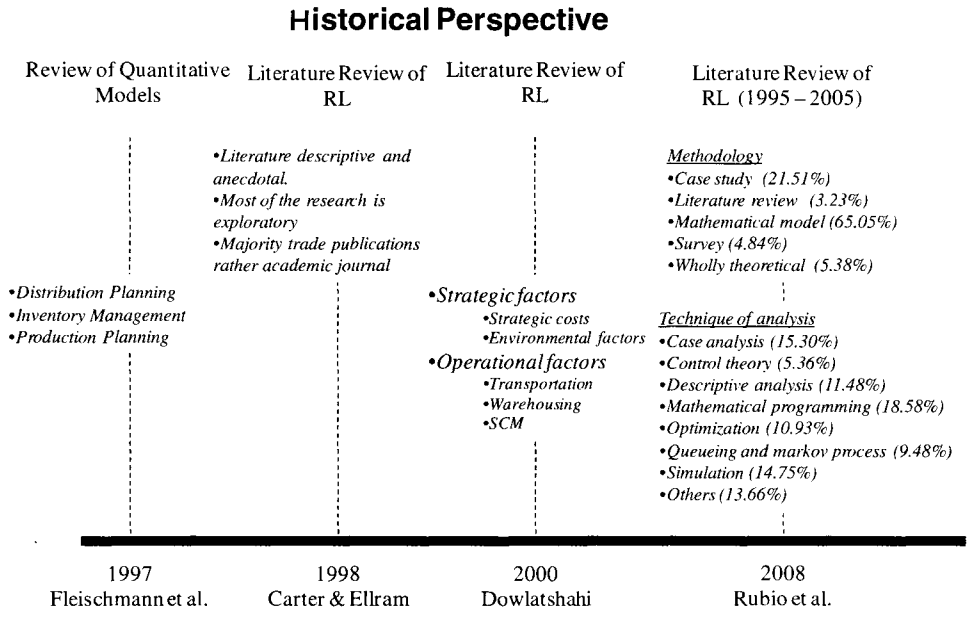


Figure 2.2 A historical perspective of reverse logistics.

The impact of logistic operations in the organizational performance is quite significant; in 2003 logistic costs were estimated about 8.5% of GDP in the USA economy. However, the inconsistent monitoring of RL costs makes them more difficult to estimate. Nevertheless, there are many estimates cited by various authors about the importance of RL and the economic impact of returned items. At the organizational level, Harrington (1994) reported that AT&T Network Systems Division has saved approximately $100 million in the nineteen months that has operated an RL program for the change of telephone equipment. Robbins-Gentry (1999) stated that customer returns are estimated at 6% of the overall sales and may be as high as 15% for mass merchandisers and up to 35% for catalogue and e-commerce retailers. Rogers *et al.* (2001) estimate that in 1999 the costs of RL in the USA were

approximately $37 billion dollars, representing 4% of total logistic costs in that year. Stock (2001) mentioned that RL costs in the USA are about $35 billion per year, or 4% of total logistic costs, which is consistent with the reported by Rogers *et al.* (2001). Daugherty *et al.* (2001) cited an estimate that returns for direct sales catalogue companies can be as high as 20% of sales. They also mentioned that RL accounts for 5 to 6% of total logistic costs in the retail and manufacturing sectors. Pogorelec (2000) estimated that 50% of online sales are potential returns, which makes managing RL a major priority. In 2004, the RLEC estimated that the costs of RL operations in the USA fluctuated between 0.5% and 1% of GDP.

A commonly found problem in the speech of logistic managers is the confusion between RL, green logistics, and related terms. This situation implicitly highlights the need to delimit the scope of RL, defining the differences with other concepts, such as: Waste Management, which refers to the management of those wastes for which there are not further uses (Brito *et al.*, 2003b); Green Logistics, which refers to the consideration of environmental aspects (such as green procurement) and other related efforts to measure and minimize the environmental impact of logistic activities (Brown *et al.*, 2003; Rodrigue *et al.*, 2001, Rogers *et al.*, 2001); the Green Supply Chain, which presents a value-search approach to prevent pollution, taking green initiatives as competitiveness resource (Van Hoek, 1999); Return Management, which is commonly used as a synonym concept of RL, however, both are at different decision levels. The RL is a set of processes that are performed on tactical and operational levels of a network recovery. On the other hand, return management is a tool to support all operations of RL to ensure that the recovery of products and packaging are cost effective while meeting all environmental obligations (Croxton *et al.*, 2001, Dekker, 2003).

2.3 Characterization of Recovery Networks

The development of RL systems has become an essential issue in the supply chain management agenda. Its development as a business unit has drawn the attention of researchers from various disciplines. If we review the number of articles published under the keyword of "*reverse logistics*", the number of papers presented at several international conferences, as well as the number of theses and dissertations at various universities more than 1000 references will appear. There are several papers presenting a review of the literature in the area of reverse logistics. These works include Fleishmann et al. (1997),

Carter and Ellram (1998), Dowlatshahi (2000), Kocabasoglu Prahinskia (2005), and more recently Dowlatshahi (2010) to mention a few. Drawing on these works and in some other papers in this section we discuss their contributions to derive a general characterization of RL systems in six dimensions. This structure will allow us on one hand to count with a framework for the classification of RL systems, and the other a set preliminary considerations for implementing a recovery program. Table 2.1 describes the dimensions and main questions to be considered, which are briefly discussed below.

Table 2.1 Reverse logistics network dimensions.

Dimension	Main question to be answered
Motivation and drivers	Why is a product recovery program needed?
Recovered items	In which industry the organization performs? Which are the main items to be recovered?
Recovery options	What are the valid forms of reuse?
Involved actors	How will this affect the current operational structure of the supply chain?
Types of returns	What kind of returns will be considered?
Advantages over traditional forward logistics models	What advantages and/or benefits will be obtained by implementing this program?

2.3.1 Motivation and Drivers

From a macro-perspective, the need to implement a product recovery program obeys to economical and environmental issues. Such issues are usually motivated by external factors such as suppliers, customers, government and competitors. However, some product recovery programs are encouraged proactively, given the strategic implications of partial or total product recovery benefits on business performance. The focus of product recovery programs is waste reduction, which is a major concern in industrialized countries and one of the major initiatives for the growing interest in the reuse and recycling. This goal is translated into company level through environmental laws, which force organizations to recover their end-of-life products and take care of their future treatment. In addition, a "green image" is becoming an important marketing factor, especially for customers with growing environmental awareness.

By comparing the case studies related to RL between the USA and Europe, Brito *et al.* (2003b) found that in the USA the main motivators for the development of RL were the economic benefits, while in Europe the primary motivator has been the environmental law. Naturally, the importance that each supply chain gives to each one of these factors at the

moment to extend its logistics network operation to a recovery network depends on its characteristics and internal motivators. In this sense, every supply chain faces its own obstacles or areas of improvement.

2.3.2 Recovered Items

The first examples of product recovery programs are found in both the remanufacturing of auto-parts and in the collection of returnable containers. However, in recent years its scope has expanded to various industries with characteristics and/or particular business environments that require different types of recovery networks. Therefore, each type of return requires an RL network appropriate to the characteristics of the returned products to optimize value recovery. Previous studies have shown that RL projects have been directed in many industries. Hence most of the published papers in the literature have been practitioner oriented, where real-world problems have been outlined as optimization problems within the logistics–production interface which falls within the RL management field. For example, Ammons (1997) studied the problem created by the recovery of carpets in the USA, Barros *et al.* (1998) dealt with the problem caused by the generation of construction wastes in the Netherlands, Sharma (2004) addressed the problem caused by the recovery of leasing equipment due to technological change, to name a few. In addition, some particular industries as the computer hardware industry has already begun to embrace RL programs by taking steps to streamline the way they deploy old systems; and in the process make it easier for the customers to refurbish existing computers or buy new parts.

The analysis of this practitioner oriented problems allows us to categorize the main types of recovery items in: packages (e.g. pallets, cans, containers), spare parts (e.g. tools, engines, engine components), waste of electric and electronic equipments (refrigerators, washing machines, microwave ovens), office equipment (e.g. copiers, phones, cameras), waste products (e.g. construction waste, carpet, tires, paper, steel), hazardous waste (e.g. batteries, pharmaceuticals), others (e.g. books, products in retail stores). The return percentages vary widely by product category, by season and across global markets, and are typically much higher for internet and catalogue sales. Return rates are also rising in Europe rapidly due to new European Union policies governing internet sales, and the entry of powerful USA based resellers.

2.3.3 Recovery Options

For every product type there is a candidate set of recovery options, which differ according to the product quality standards, the deterioration process, and its pattern of use (Brito and Dekker, 2003). In general, once the items have been recovered, a detailed inspection is needed for a proper selection and classification. After that, the decision maker must decide between a direct recovery (e.g. reuse, resale or redistribution), or indirect recovery (e.g. repair, remanufacturing, recycling, refurbishing, etc.). Some of them can be applied both to product and packaging (see Rogers *et al.*, 2001 p.133). In Table 2.2 we list some of them.

Table 2.2 Recovery alternatives for products and packaging.

For products	For packaging
• Return to the supplier	• Reuse
• Resell	• Renew
• Sell to an alternative market	• Recycling
• Reuse	• Land filling
• Reconditioning	
• Refurbishment	
• Remanufacturing	
• Recycling	
• Land filling	

2.3.4 Involved Actors

The actors involved in the recovery activities may include both original producers and third party logistics service providers (3PL). This will depend on the activities of collection, storage, sorting, disassembly, reprocessing and remarketing that the recovered items usually require. However, typical recovery network structures combine the forward supply chain actors with specialized actors for the movement and treatment of the returns. Usually, collective management systems are created by the original manufacturing enterprises to take their products back.

2.3.5 Types of Returns

The returns can be broadly classified as internal returns and external returns. The internal returns are known as *manufacturing returns*, i.e. all those products that are returned in the production phase, either for not complaining with quality standards or because they were not fully completed due to lack of material, usually these returns are referred to as *rework* or *reprocesses*. External returns are all those goods that have left the manufacturing plant, these include *guaranty returns*, i.e. products that are damaged, products whose quality was not able to meet customer needs or products that need to be repaired; *commercial returns*, i.e. products that were excess of inventory or whose season (off season) already happened and were not sold: *end-of-life returns*, i.e. products whose economic and/or functional life-cycle has been fully completed; *end-of-use returns*, i.e. those products that the customer has the opportunity to return within a certain period of time, but whose life-cycle is partially consumed, this is the case of used books. Identifying the type of return is of paramount importance to define its final destination. It should be noted that there is a vast literature for other types of reverse flows as the post-sales services and the guaranty returns.

2.3.6 Advantages over Traditional Logistics Models

Several authors have described the benefits obtained by the implementation of an effective product recovery system. These include the improvement of customer satisfaction, the reduction in the investment of raw materials, savings in acquisition, disposition, maintenance and transportation costs of inventories. In many organizations the recovery of packaging and products has become a practice capable of reducing the environmental impact of the supply chain, facilitating the elimination of waste while increasing profits (Rogers *et al.*, 2001, Autry *et al.*, 2001, Ritchie *et al.*, 2000). The return on investment is maximized by recovering value in the products, improving the collection process and improving the relationship with customers. Also better information management systems are reported by including end-customer feedback and facilitating the forecasting process (Hughes, 2003; Krikke *et al.*, 2001a; and Hillebersberg *et al.*, 2001). Furthermore, consolidating return volume form fewer locations improves vehicle fill, reduces transport/courier cost; eliminate costs for failed collections. Accurate stock returns leading to a reduction in disputes/credits. Faster cycle times means less product damages, fewer losses.

2.4 Operations Research Models for Reverse Logistics

The use of optimization models in the areas related to RL is not a new process. According to Guide *et al.* (2003) remanufacturing operations have been present in some industries, e.g. the automotive industry since 1920. However, the novelty of this field of study is the growing need to develop *viable business models* capable of adjusting the business aims with the environmental sustainability in the same plane. To define a framework for organizing the review is a difficult task, given the interrelationship between the various areas of research and analysis. However, for the purposes of this chapter the following areas of research are considered: 1) facility location and network design, 2) inventory control and production planning, 3) disassembly operations and recovery alternatives, 4) vehicle routing in reverse logistics, and 5) forecasting and information technologies.

These areas are very similar to those performed in the forward logistics process; therefore several authors (Fleischmann *et al.*, 1997; Soto, 2005, Dekker *et al.*, 1998) have proposed similar classification schemes. Each area corresponds to a decision and/or optimization problem, which are interrelated although their study is commonly dealt independently. Each area has a significant impact on the design and management of RL systems, and despite its apparent similarity with the determinants of performance of a forward logistics system, there are certain differences that justify the development of new theories.

2.4.1 Facility Location and Network Design

This area of research comprises all the design of the reverse network. It is concerned with the optimization of the location and capacity of the facilities as well as the flow of goods between them. In traditional location models, the demands and operational costs are considered inputs of the location models. In RL demands are located not only at one side of the chain but in both, since the secondary markets, disposal facilities, etc. also receive the product of the company.

There are a number of considerations which establish that the recovery networks differ from typical forward-only networks. Between the main differences reported in the literature are: a) the natural convergent structure of the network from several sources to few points of demand, versus the divergent structure of forward distribution, b) the combination of pull and push systems in recovery networks, versus the strong presence of pull systems on forward

logistics, c) the high degree of uncertainty of returns in terms of quantity and quality, that is translated into an undefined process of returns to be transformed in reusable products, d) the non uniform price of returns due to its variable quality (Fleischmann *et al.* 1997; Tibben-Lembke *et al.*, 2002). Furthermore, in a typical forward-only logistic network the target market is known in advance, whereas in a recovery network the final decision about the destiny of returns widely depends on the characteristics and quality of the goods that have been recovered, being the potential market for recoverable products another source of uncertainty (Rogers and Tibben-Lembke, 2001; Guide *et al.*, 2000; Fleischmann, 2000). As a direct consequence, coordination between the disposer market and the reuse market is required.

To determine when reverse flows should be integrated with forward flows Fleischmann *et al.* (2001) simulated the impacts of reverse flows in a logistic network. According to them influence of return flows is very much context dependent. The main reasons to create a separate infrastructure for reverse flows is when forward and reverse flows differ with respect to geographical distribution or cost structure and when return volumes are substantial. Tibben-Lembke and Rogers (2001) studied the efficiency of centralizing returns handling into a forward distribution centre. They claim that the return process is often neglected in favor of the "more important" forward process. Their study concentrated on commercial returns. With end-of-use returns there is often a need for expensive testing equipment and skilled labor. Then there is mainly a tradeoff between transportation and investment costs. In the case of end-of-use returns centralization is also limited by legislation, as waste cannot be transported across international borders.

During the last decade, a considerable number of articles that report facility location issues on recovery networks have been published. In each of the articles a quantitative model for the recovery network design problem has been developed. Facility location models presented in these articles range from mixed integer linear programming (MILP) optimization models (Kroon and Vrijens, 1995; Shih, 2001; Hu *et al.*, 2002), to continuous approximation models (Fleischmann, 2001), including life cycle models (Bloemhof *et al.*, 1996), stochastic models (Listes and Dekker, 2005) and mixed integer nonlinear programming models (Lieckens and Vandaele, 2007).

From the analysis of previous research works we point out the following observations: a) most of the location models presented in the literature preserve the same characteristics of the traditional location models minimizing the overall system linear cost and proposing slight modifications and extensions taking into account special characteristics such as the

convergent structure of the network from many sources to few demand points as well as technical rates of reuse feasibility, effective recovery and recovery; b) several papers in this area are practitioner related, in which a model is designed for a specific problem and/or company, nevertheless, given the complexity of the recovery networks derived from the uncertainty in both quality and quantity of the returns, it is difficult to create models for general purposes; c) except by Shih (2001) and Hu *et al.* (2002) which present cost minimization models in different industries; environmental parameters (e.g. environmental cost, waste, energy) have been excluded from the models, leading special attention to the economical structure of costs; d) except by Jeung-Ko and Evans (2006), dynamic distribution networks and nonlinear models are rarely studied as an integrated aspect for optimizing the forward and return network simultaneously; e) most of the papers discussed addressed uncertainty in product returns by examining a finite set of scenarios, however, Ammons *et al.* (2002) extend this idea by developing a solution methodology using an upper and lower-bounding scheme on a robust objective function, to 2004 Assavapokee (2004) extends it further to develop a methodology to solve problems where the parameters can take values from a compact real interval; finally, f) most of the models presented are solved using commercial mathematical optimization solvers like CPLEX, AIMMS, MPL, and LINDO.

2.4.2 Inventory Control and Production Planning

In the recovery networks, inventories take the form of original and recovered raw materials, work in process, finished goods, returns and disassembled parts. The variability in RL systems makes the sorting and the storage activities an important process. Therefore, it is important to design proper inventory control mechanisms to integrate reverse flows in the production process. In some cases traditional inventory management models may be appropriate for RL systems, specifically for specialized recycling companies, where used goods are purchased by third parties. However, in remanufacturing environments adapted policies are required to simultaneously control manufacturing and remanufacturing operations. It is not surprising that most research on production planning and inventory control for RL has been performed for this type of environment.

A comprehensive analysis of previous published literature of inventory control in remanufacturing is provided by Van der Laan (1997). Thereafter new models and approaches have been developed. In this section we only discuss some recent contributions to assess the progress in this area. There are two lines of research for this problem: some efforts have been

aimed at finding a methodology for the systematic analysis of inventories in the context of reusable items, while on the other hand there are some efforts towards the valuation of inventories and the calculation of the costs associated with such systems. The first line presents a purely operational guidance to develop models to examine the effect of return policies on inventory levels, pricing strategy and perceived risk. The second line presents a marketing orientation, to study the possibility of using the return policies as a tool for the coordination of sales channels.

The most relevant papers can be categorized into three groups: cash balance models, periodic and continuous review models. Basically the difference between them is that in continuous review models, lead and repair times can be modeled, while in periodic review and cash balance models they cannot. But in the last case the value of demands and returns are modeled in money and, some inventory control policies are considered (Van der Laan *et al.*, 1997).

Several quantitative papers have been made on production scheduling, inventory control and remanufacturing. Fleischmann *et al.* (2002) propose an inventory model for a single item with independent Poisson demand and returns, they show the optimality of an (s, Q) policy for the purchasing of new items. In Teunter and van der Laan (2004) we have a good starting point to address issues related to the valuation of inventories in RL systems. In Teunter *et al.* (2000) different methods for calculating the rates for the opportunity costs of remanufactured and manufactured items are discussed and compared. Minner (2001) shows the need for a balance between safety stock levels and the savings in purchasing costs when (re)designing a supply chain to embrace an RL system. Van der Laan *et al.* (1999) studied the production planning and inventory control in systems where manufacturing and remanufacturing operations occur simultaneously. They present a methodology to analyze a push control strategy (all returned products are remanufactured as early as possible) and a pull control strategy (where all returned products are remanufactured as late as it is convenient). They derive some managerial insights into the inventory-related effects of remanufacturing. Guide *et al.* (1997) evaluate some priority dispatching rules in combination with a modified drump-buffer-rope scheduling system under a variety of utilization levels (low, intermediate and high). They simulate those scenarios and evaluate the results obtained. Savaskan *et al.* (2004) propose a model to find the optimal mix of products for manufacturing and remanufacturing environment. Finally, Ferrer and Whybark (2001) describe an integrated material planning system to assist the administration of a remanufacturing plant.

2.4.3 Disassembly Operations

Disassembly is an activity which is not present in forward logistics. This activity consists in doing the inverse process of assembly. There are several issues to consider in this area, first decision is concerned with disassembling or not disassembling a product, since the quality of the parts is unknown. There is a probability that depends on the cause of return of the product of getting a part (or assembly) of good quality for remanufacturing. Second common decision in this area is on the level of disassembly. If the company decides to disassembly the product in several assemblies, and some of the assemblies don't work, the company decides whether disassembling again these assemblies or not. The decision is up to which level of disassembly is profitable for the company to disassembly. An additional problem arises from the upgrading or downgrading state of the product during its use by the customer. These changes in the product due to repair, upgrading or downgrading, increases the uncertainty of the quality state of the assemblies when the product is disassembled. There is also a damage risk in this disassembly process, since up to now; products have not been designed to support a disassembly process. To deal with these problems companies are working today in Design for Disassembly processes to minimize the impact in the disassembly process at the end of the life of the product.

Several authors illustrate the problem of disassembly. Zussman (1995) propose a disassembly system where the end of life value of the product is taken into account to evaluate the strategy to follow in the disassembly process. The process is complemented with a correct identification of the assemblies with adhesive labels, previously attached to the product when it was manufactured. Krikke (1998) proposes a model for the disassembly strategy for a single product. He proposes a two step optimization model to solve the problem and, complements the problem with the case of a television device. Gungor and Gupta (1998) propose a methodology for the Disassembly Sequence Plan. They propose to first generate a precedence disassembly matrix that represents the physically based precedence relationships between the assemblies of the products. Then an optimum disassembly sequence plan is proposed, and finally third step consists in performing the disassembly process, following the optimal disassembly sequence plan and, adjusting the process when an unexpected situation arises. Guide *et al.* (1999) examine the impact of the variability of the highly variable lead times on the control of parts released from the disassembly area to the remanufacturing area. They evaluate various disassembly release mechanisms for releasing the parts. Lead time variation is shown to have a significant effect on the choice of the disassembly release

mechanism. They only consider the lead time as a source of uncertainty, it is necessary to evaluate also the impact of other factors such as the variation in the number of returns received and the size and location of the inventory buffers. González and Adenso-Diaz (2004) propose a scatter search metaheuristic to deal with the optimum disassembly sequencing problem for the case of complex products when they should be reused or shredded at the end of its life. They use sequence-dependent disassembly costs, assuming that only one component can be released at each time. They evaluate the heuristic with a set of 48 products comparing it with the time spent by a feasible sequence consisting of disassembling the components in the reverse order of their assembly sequence.

2.4.4 Vehicle Routing in Reverse Logistics

The vehicle routing problem (VRP) is one of the most studied in the combinatorial optimization field and in the logistics literature. There are different variants of the VRP that can be characterized by the type of fleet, the number of depots, or the type of operations involved, among others. While in recent years, there has been an increasing interest towards so-called rich VRP models that include important issues arising from real-world applications (Battarra *et al.*, 2009), the literature on vehicle routing for reverse logistics is still very scarce. Even more, comparing the published literature between facility location issues and vehicle routing issues in RL, a considerable gap in vehicle routing for RL can be observed.

In reverse logistics most of the papers address specific problems and case studies. Schultmann *et al.* (2006) deal with the problem caused by the collection of plastic components of vehicles at the end-of-life phase in a closed-loop network in Germany. The problem is formulated as a symmetric VRP with backhauls, capacitated vehicles and maximum length tour constraints. The authors describe a tailored tabu search algorithm to generate a tour schedule at minimal cost. Blanc *et al.* (2006) describe the problem of optimizing a closed-loop collection network for container vehicles in the Netherlands. The problem is formulated as a multi-depot VRP with simultaneous pickup and delivery in alternate locations. Kim *et al.* (2009) discuss a case study in which the metaheuristic approach TABUROUTE applied for solving the design of WEEE collection routes in South Korea. On their own, the theoretical contributions founded under the keywords vehicle routing in reverse logistics or collection routing, mainly deal with the vehicle routing problem with simultaneous pickup and delivery (Dethloff, 2001; Dell'Amico *et al.*, 2006).

It could be due to the fact that, contrary to the facility location problem, the differences in the VRP between the forward logistics and the RL are not perceived at first sight. From a general standpoint, the vehicle routing problem makes no difference whether the items are new or used, or if they need to be delivered or collected. Therefore, some authors have considered that there are not differences between classical vehicle routing problem and routing collection problem. However, a more detailed analysis that considers the characteristics of RL systems and the collection context provides us with a better understanding of these differences. Beullens *et al.* (2004) identified five dimensions to characterize routing systems in RL: a) the collection infrastructure, 2) the collection policy, 3) the level of combination in the collection, 4) vehicle characteristics, and 5) degrees of freedom. The collection context refers to the type of items to be collected. From transportation point of view is not the same to collect batteries, auto parts, construction waste, returnable containers, or waste of electric and electronic equipments among others. Each return type has specific characteristics and the vehicles used in their collection are different. Furthermore, in the case of outsourcing, for the logistics service providers is not relevant whether a product will be delivered or collected.

2.5 Waste of Electric and Electronic Equipments (WEEE)

The production of electrical and electronic equipment (EEE) is one of the fastest growing markets in the world. At the same time this also means that the amount of waste electrical and electronic equipment (WEEE) will continue to increase in the coming decades. In its collection and management process several actors are involved; each one performs a specific function at each stage of the process. To understand how management is performed it is required to clarify which actors are involved and how are the interactions between them. Figure 2.3 shows a flow chart of the entire WEEE collecting and treatment process. As can be seen during the flow of WEEE from the moment they are generated until they reach the treatment plant several actors may be involved.

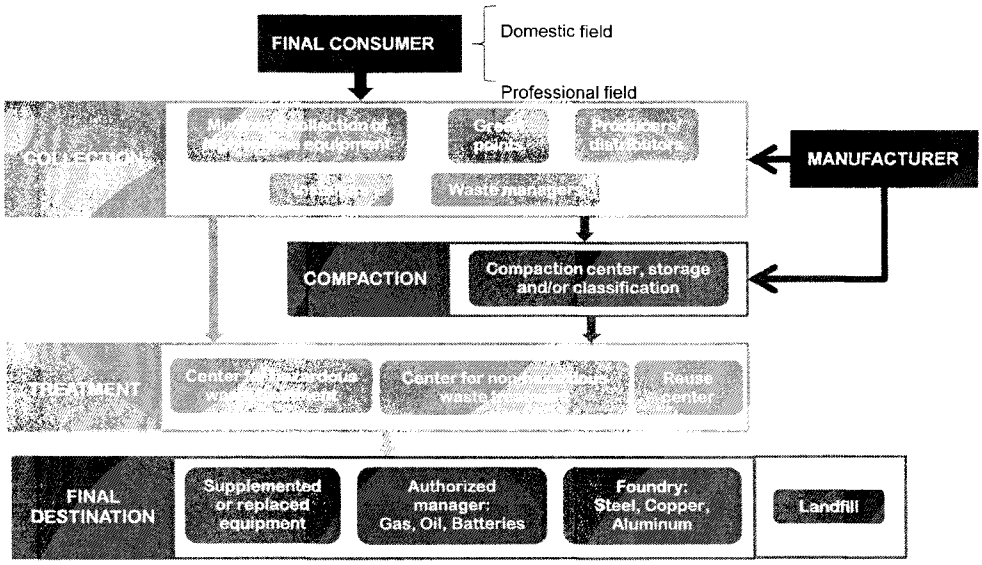


Figure 2.3 The WEEE collection and management process.

The WEEE is generated when the appliances (i.e. EEE) are replaced by new ones or when they have outlived their usefulness, even if they are not replaced. Its origin can be from the domestic (household products) or professional field. There is also a small fraction of WEEE originating from the manufacturers (producers), usually for being defective. The first stage of the management process is the collection. This can be accomplished in different ways such as municipal waste collection services for bulky equipment; in this case the city council of each municipality is responsible for the collection of WEEE and its transportation to treatment centers or intermediate facilities where there is storage pending to be recycled. Other options are the collection points, electric appliance distributors, installers and waste managers. In the second stage of the management process the treatment facilities are found. For the treatment of WEEE the authorized centers may be of several types depending on the characteristics of the waste to be treated. Sometimes, after collection, the WEEE goes through compaction process before arriving to the treatment centers. Within the process of treatment of WEEE it should be noted the role of scavenging as a special type of agents that also take part of this management system. The last stage of the process determines the final destination of the various fractions resulting from the treatment of WEEE.

2.5.1 The WEEE Directive

The Directive 2002/96/EC (EU, 2003) of the European Union on Waste Electrical and Electronic Equipment (WEEE) imposes on all EU member states to develop legislation based on Extended Producer Responsibility (EPR). The WEEE directive was approved by the European Parliament on 27 January 2003. EPR makes the Original Equipment Manufacturer responsible for the take-back and recovery of returned products. Its main aim is to promote reuse and recycling by imposing collection and recovery quota and to reduce e-waste by enhancing the eco-design of products. Also, certain product recycling information must be made public and product marking must be applied to products new on the market.

This so-called WEEE-directive distinguishes 10 product categories (see Table 2.3) both concerning B2B and B2C markets. For each category, three recovery options are allowed: component reuse, material recycling and incineration (or energy recovery). There are two types of quota defined: the consumer electronics market has to meet a number of targets that define minimum rates, such as minimum rate of recovery of 70–80%, which includes incineration with energy recovery, and a minimum rate of component, material and substance reuse and recycling (represented by so-called weight balances) of 50–70%. In addition, treatment of the collected products is required to remove fractions or groups that contain hazardous materials, such as batteries, printed circuit boards, cathode ray tubes, and external electric cables. The mandatory isolation of (mostly) hazardous contents fixes a minimum degree of disassembly for products containing these substances and also the recovery route is prescribed in detail; see Annex II of the Directive (EU, 2003).

Table 2.3 The 10 WEEE product categories.

1	Large household appliances
2	Small household appliances
3	IT and telecommunications equipment
4	Consumer equipment
5	Lighting equipment
6	Electrical and electronic tools
7	Toys, leisure and sport equipment
8	Medical devices
9	Monitoring and control instruments
10	Automatic dispensers

The purpose of the WEEE directive is to prevent the creation of electrical and electronic waste and to promote reuse, recycling and the other forms of recovery in order to reduce final disposal. Responsible are the currently acting producers and importers of electric equipment

who can fulfill their obligations individually or by joining a collective scheme. Member states must assure that the treatment is undertaken by competent, officially approved operators. In Spain the directive came into effect on February 25 2005 by The Royal Decree 208/2005. The adoption of this legislation has led to essential changes in the field of electronic scrap recycling. Particularly, producers and importers of large and small electrical appliances are affected by the new normative, since they will have to manage over 75% of the total electronic waste. An operational end-of-life consumer electronics recovery system should have been ready as of August 13, 2005 (Toffel, 2003). However, many member states have not met this deadline, there is little doubt that enforcement will become firmer, and that in the long run member states need to comply.

2.6 Designing Recovery Networks for WEEE Collection

Most of the previous work dealing with the design of reverse logistics networks for WEEE is based on facility location models. Queiruga *et al.* (2008) provide a method to rank alternatives for recycling plant installations in Spain. Walther et al. (2008) focus on facility location planning for the treatment of large household appliances (WEEE category 1). A capacity and facility adaptation planning model for remanufacturing mobile phones is presented by Franke *et al.* (2006). Other contributions consider the problem of locating collection and disposal areas for urban waste. Vuk and Kozelj (1991) apply the PROMETHEE methodology and the geometrical representation of multi-criteria analysis to select locations for disposal sites of municipal waste in Slovenia. Bautista and Pereira (2006) develop GRASP heuristics to solve the waste collection site location problem. Kara *et al.* (2003) and Kara *et al.* (2007) present a simulation model to calculate collection costs for EOL appliances in the Sidney Metropolitan area.

Few recent studies particularly deal with WEEE recovery optimization in networks. Walther and Spengler (2005) combine reverse logistics and disassembly planning models. Walther *et al.* (2008) explore the negotiation coordination mechanism in product recovery networks by using contract theory. Shih (2001) aims at optimizing the infrastructure design and the reverse flow in an integrated approach for the recycling of computers and electronic devices in Taiwan. Nagurney and Toyasaki (2005) develop an integrated framework for modeling the RL networks that include WEEE recycling. Brian and Lu (2006) analyze the localization options to determine the feasibility of carrying out an RL program for an

international manufacturer of electronic equipment in the Asia-Pacific region using the analytic hierarchical process. Aras *et al.* (2008) develop a tabu search algorithm for solving the problem of locating collection centers and to establish the optimal values of the incentives for different types of returns. It is assumed that users of the products have an inherent interest to make returns, and make the decision on the basis of financial incentives offered by the company. The incentives depend on the condition of the items returned. Of particular interest to our problem is the fact that harvesting is carried out with capacitated vehicles. Finally, Grunow and Gobbi (2009) present an optimization-based decision support tool for the design of WEEE networks. The approach was tested for the WEEE network in Denmark, where substantial improvements were obtained for all actors in the network.

2.7 Vehicle Routing in WEEE Reverse Logistics

Two of the main assumptions under which the classic VRP model is founded are the use of a homogeneous fleet of vehicles and the fact that the demand of each node is smaller than the capacity of the vehicles used. Note however that in the context of WEEE collection, both assumptions should be relaxed given the way in which these systems work, where the amount of WEEE generated by the collection point tends to be greater than the capacity of vehicles used in the harvesting, more over in WEEE collection systems, a heterogeneous fleet of vehicles with special characteristics is frequently used. Just one reference (Kim *et al.* 2009) was found to deal with the VRP in WEEE reverse logistics; however, the authors simplify the assumptions and apply the metaheuristic algorithm TABUROUTE to solve the WEEE collection problem in South Korea.

The vehicle routing models designed for WEEE collection problems in reverse logistics should consider that a fixed and heterogeneous fleet of capacitated vehicles with special features is used in the harvesting of WEEE from a set of customers. Each vehicle can perform at most a predefined number of tours, while the given maximum operation time is not exceeded. The amount of end-of-life items to be collected (hereafter referred to as the demand) of each customer can be greater than the capacity of the largest vehicle in the fleet; as a result the demand of a single customer may be split to be fulfilled by more than one route. Thus, each customer can be visited more than once on the same day. Collection orders to customers are triggered by a call from them. With each order there is an associated demand to be fulfilled and a range of dates within the collection should take place.

This problem is characterized by four variants of the VRP that have been independently studied in the logistics literature, including: (1) the use of a heterogeneous fleet of vehicles, (2) customers with high demand that may be split to fit the vehicle capacity, (3) the fact that a same vehicle may be assigned to more than one route, and (4) the agreement of a time interval for visiting customers. These characteristics make the problem complex and interesting, while complicating its classification within the standard VRP literature (Eksioglu *et al.*, 2009).

2.7.1 Variants Related to WEEE Collection Problems

The variants that characterize the problem addressed in this research have been previously cited. Feature (1) mentioned above is referred as the heterogeneous fleet vehicle routing problem (HF-VRP), of which there are three variants, two of them relating to the number of vehicles of each type: limited or unlimited; and the third one refers to the accessibility restrictions of vehicles to customers. Among the first authors undertaking the HF-VRP we refer to Golden *et al.* (1984). Recently Imran and Wassan (2009) propose an adaptation of the Variable Neighborhood Search metaheuristic embedded with two variants of the Dijkstra algorithm for the general problem. The third variant of the HF-VRP is referred to as the Site Dependent VRP (SD-VRP), and was introduced by Nag *et al.* (1988). Feature (2) is referred to as vehicle routing problem with split loads (VRP-SL), which was introduced by Dror and Trudeau (1989) detailing its heuristic properties. They showed that the partition of loads may result in savings, for both the total travelled distances as well as in the number of vehicles used. Recently, Mitra (2008) shows that by allowing the loads to be split, still better results may be obtained than those reached by the combined objective of minimizing the fixed costs of load and costs associated with the routes of vehicles.

Feature (3) refers to the contributions in the vehicle routing with multiple use of the same vehicles. Despite the standard VRP definition implicitly assumes that each vehicle is used only once over the planning horizon, in reality once the routes have been designed, it is possible to assign several of them to the same vehicle in such a way that the vehicles can operate feasible in the planning horizon. The problem of designing routes with multiple trips is known in the literature as the multi-trip VRP (MT-VRP) and was introduced by Taillard *et al.* (1996) and Brândao and Mercer (1998). The problem featured by (4) does not establish periodic visits as in the periodic VRP, nevertheless in each call the interval of days within the collection should take place is previously set, so this feature can be understood as a date

window that provides route flexibility. It should be noted the dissimilarity with the VRP with Time Windows (VRPTW), which is a generalization of the classical VRP wherein each customer i must be served within its corresponding time window $[e_i, l_i]$, hence the routes should be designed in such a way that each customer be visited only once by exactly one vehicle within the given interval of time. On the other hand, the VRP with Date Windows (VRPDW) is less restrictive than the VRPTW respecting to the time of visits, however the VRPDW is conducted at a broader level in a period T, without being a periodic VRP (each customer must be visited any time in a predefined interval of days). For example in the VRPTW two different customers $\{i,j\}$ cannot be served at the same time by the same vehicle, however in the VRPDW two different customers can be served by the same vehicle on the same day. The VRPDW is neither periodic given that what is collected on day t is dependent on what was collected on day $t - 1$.

The literature review shows that features (1) and (2) were studied in joint form by Belfiore and Yoshida Yoshizaki (2008), which propose a Scatter Search algorithm to solve a real-world problem with a heterogeneous fleet of vehicles with time windows and split loads. On their own, the research that more resembles the assumption described by the feature (4) is the one developed by Alvarez *et al.* (2009), who consider a VRP in which despite the existence of a delivery due date, there is certain flexibility to perform the delivery some days before. Features (1), (3) and (4) were addressed by Alonso *et al.* (2007) who introduced the Site-Dependent Multi-Trip Periodic VRP (SDMTVRP). Nevertheless, to the best of our knowledge these four features have not been studied all together.

2.8 Disassembly Cell Formation Problems

New trends in environmental conscious manufacturing and reverse logistics have motivated a significant effort to improve the performance of several recovery systems. Product recovery is an important activity either to obtain useful components with economic value through reuse, remanufacturing or recycling, or to remove hazardous components in compliance with safety regulations. Regardless the final destination for the collected products, disassembly operations are required and must be optimized to improve the overall performance of the recovery system.

The use of Cellular Manufacturing Systems (CMS) based in the Group Technology paradigm, have proved to give good results in typical manufacturing enterprises. Reported

benefits include reduced set-up times, material handling cost, in-process inventory, better production efficiency and quality as well as market response time (Jayaswal and Adil 2004; Nsakanda *et al.* 2006; Chakravorty and Hales 2008). Most of these benefits come from the fact that the cells have a certain amount of functional autonomy, facilitating better control on the shop floor. These benefits can be extended to recovery systems, specifically to the disassembly process if we consider the use of a cellular configuration to take into account the similarities in the disassembly tasks. Usually disassembly processes are characterized by the arrival of several, similar types of products to be dismantled each one with a low-medium volume. Hence, the use of a cellular configuration is justified.

This section reviews published literature on cell formation problems. It specifically reviews the problems of designing robust and reconfigurable disassembly cells to deal with demand variability. The motivation for this study is based on the fact that, contrary to the assembly process, in reverse logistic systems most of the recovered products arrive to the disassembly center in a variable quantity and quality. Therefore, there is a need to include such variability in a proactive manner to decision-making process.

2.8.1 Manufacturing Cells versus Disassembly Cells

Cellular manufacturing systems have been a profiler research area during the last 55 years; in fact the cellular manufacturing philosophy has emerged to cope with global competition, shorter life cycles and shift on product demand. Cellular manufacturing involves the processing of a collection of similar parts (part families) on dedicated clusters of dissimilar machines or manufacturing processes (cells). Group technology is an approach to manufacturing and engineering management that identify similar parts, and group them together into families to take advantage of their similarities in manufacturing and design (Selim *et al.*, 1998). The aim of CMS is to form perfect (i.e. disjoint) groups in which products do not have to move from one cell to the other for processing achieving benefits of economy. The benefits of CMS can only accrue to the company if strategic decisions are based upon results obtained from models that accurately describe its structural and operational features (Kioon *et al.*, 2009). Therefore it is important for companies that use CMS to invest enough time in the design and planning phase of any CMS implementation.

Among the main technical problems faced in the design of CMS, the cell formation problem has been one of the most studied problems in the manufacturing literature, and the first problem faced in practical applications. Cell formation means determination of the

assigned machines to each cell and determination of the manufacturing family corresponding to these machines. To address this problem, several and diverse solution procedures have been developed and published over the last years. Wemmerlöv and Hyer (1989), Singh (1993), Selim *et al.* (1998), and Papaioannou and Wilson (2009) provide extensive reviews of prior research. Singh (1993) has classified the approaches to cell formation into the categories of coding and classification systems, machine-components group analysis methods, graph theoretic methods, neural networks and heuristics, fuzzy clustering based methods, similarity coefficient based mathematical models, knowledge and pattern recognition methods, mathematical and heuristic methods. Selim *et al.* (1998) have categorized the approaches as cluster analysis, graph partitioning, descriptive procedures (part families identification, machine groups identification, part families/machine grouping), mathematical programming (linear, integer and quadratic programming, goal and dynamic programming), artificial intelligence approaches.

Most of the techniques available on literature for designing CMS consider the operation commonality by defining a measure of similarity between parts and then grouping parts into families. The similarity measure helps ensure that cells are focused if parts produced within a cell have a high level of similarity. However, a similarity measure approach generally does not consider part volumes, machine capacities and processing times. Part volumes are important because high volume parts may have a much larger impact than lower volume parts on material handling costs. Machine capacities are important for two reasons. First it may be that when processing loads are considered it may be required to have multiple machines of a given type and these machines can be allocated to smaller, more focused cells. Also, when considering processing loads it may happen that multiple machines of a type are required in a cell. Therefore, CMS designs are currently being researched with the emphasis on more integrated models and solution methodologies (Kioon *et al.*, 2009; Mansouri., 2000; Selim *et al.*, 1998; Singh, 1993). Some cell formation techniques have explicit or implicit objectives, such as the minimization of intercellular movements that do not necessarily produce the best overall cell performance or satisfy application-specific objectives.

Given the practical differences between manufacturing systems and disassembly systems, typical cell formation problem formulations are not suitable for the *Disassembly Cell Formation Problem* (DCFP). In manufacturing systems, all operations belonging to a product are required by every product of the same type, while in the disassembly process not all operations for a given product type are required by every product. For instance, a computer unit could arrive to the disassembly shop without the hard disk drive. Though the cell

formation problem has been one of the most studied in the manufacturing literature and there is a vast literature on disassembly systems (see for example Gungor and Gupta 2001, Tang *et al.* 2002), the contributions concerning disassembly cells are practically null.

Tang *et al.* (2002) groups research issues in the disassembly field into three major areas: 1) modeling and representation of product disassembly sequences (e.g. Moore, 1998) 2) disassembly process planning, which includes the extent to which disassembly should be performed (e.g. Adenso-Díaz *et al.*, 2007) and how to decide the optimal disassembly sequence (e.g. Lambert, 2003) and 3) disassembly system design and line balancing (e.g. Gungor and Gupta, 2002). Most studies in disassembly system design deal with disassembly lines (Gungor and Gupta 2001, 2002). When only one type of product is to be disassembled and its value and/or demand volume is high, a dedicated disassembly line may be justified. However, it is often the case that there are several similar types of products to disassemble, each one with a low-medium volume. Therefore, it may be better to use a cellular configuration that can take into account the similarities in the disassembly tasks. Kizilkaya and Gupta (1998) describe a typical disassembly scenario in which each family of products is disassembled in a separate cell with each disassembly cell being a succession of workstations in series. Das *et al.* (2000) also propose sorting the products into disassembly families each of which is processed in a different disassembly facility. They define a disassembly family as a group of products that have sufficient commonality, such that they can be efficiently disassembled in the same facility. To the best of our knowledge, the first reference regarding the DCFP is found in Adenso-Díaz *et al.* (2006), which present a mixed integer programming formulation that minimizes the total costs function of machinery depreciation and intercellular movement of products. The proposed model seeks to group the disassembly tasks and assign them to cells together with their required resources so that total costs are minimized. Finally, Andres *et al.* (2007) propose a two-phase approach for determining the optimal disassembly sequence when the disassembly system has a cellular configuration.

2.8.2 Variability Issues in Disassembly Cells

Uncertainty is an inherent element to all systems in reverse logistics, where variability in the quantity and quality of recovered products significantly affect the disassembly process. According to Kouvelis and Yu (1997) the best way to handle uncertainty is to accept it and make an effort to structure it, understand it and eventually make it part of the reasoning process in decision-makings. Therefore, in this section we identify the uncertainty and

variability elements in the disassembly process, allowing us to extend the deterministic DCFP model to formulations that incorporate the element of variability.

The performance of a cellular manufacturing system and particularly of a disassembly system is highly dependent on the accuracy of the input data. It may rapidly deteriorate if demand and product quality changes. However, most of the current methods designed for the manufacturing cells formation problem have been developed for a single-period planning horizon. These models assume that problem data (e.g., product mix and demand) are constant for the entire planning horizon. To address the problem of demand variability, created (among other factors) by the variability of the market and short life cycles, several approaches have been developed including dynamic, flexible and robust designs.

In the reconfigurable approach the product demand varies form period to period in a deterministic manner. Due to this dynamic condition of the demand the best cells configuration for one period may not be efficient or even feasible for subsequent periods, and some redesigns would be required; for instance to change the number and allocation of machines to cells from period to period, to adjust to demand variability. The main assumption that supports this approach states that although the demand for each period is known a priori, finding an optimal configuration for each period could be prohibitively expensive. In the robust approach the product demand varies in a random manner. However, this variation can be described in a number of probabilistic scenarios with a given occurrence probability. The aim of this approach is to obtain a cellular design in which the machines remain constant and do not move, only the flow of products changed once the demand have been observed. In both approaches the robustness and the ability to reconfigure respectively are indicators of the flexibility that has a cellular manufacturing system to manage demand variability. The election of the approach to use hardly depends of the assumptions that most reflect the manufacturing system reality. We refer to Balakrishnan and Cheng (2007) for a detailed literature review on robust and reconfigurable cell formation problems.

2.8.3 Reconfigurable Cell Formation Problems

Even though most of the models consider static conditions, dynamic situations are more realistic when a multi-period planning horizon is considered. In such environments the product mix and demand in each period is different. This occurs in seasonally or monthly production systems. As a result, the cell configuration in one period may not be optimal in another period. To address this problem, several authors recently proposed models and solution procedures by considering dynamic cell reconfigurations over multiple time periods

(e.g. Chen, 1998; Wicks and Reasor, 1999; Mungwattana, 2000; Tavakkoli-Moghaddam et al., 2005a,b; Balakrishnan and Cheng, 2005; Defersha and Chen, 2006; Saidi-Mehrabad and Safaei, 2006; Schaller ,2007).

Allowing system reconfiguration enhances the flexibility of disassembly systems to respond to this dynamic condition of demand. In fact, by rearranging the cells, the disassembly system can continue operating efficiently as the product mix and demand change. Reconfiguration consists of swapping the existing machines between cells (machine relocation); adding/removing machines to/from cells, machine duplication and changing the process plan of the part-operation. Also, the reconfiguration may consist of increasing or decreasing the number of formed cells in each period. The purpose of the reconfigurable approach in our disassembly cell formation problem is to design a cell configuration within each period that establishes a perfect balance among operating and reconfiguration costs between successive periods to adjust to long-term demand changes.

2.8.4 Robust Cell Formation Problems

2.8.4.1 Robust Optimization

Robust optimization is a framework for modeling optimization problems that involve uncertainty. It has been applied to several areas of research and practice, such as production planning, machine programming, international supplies, and logistics. We refer to Cao and Chen (2005) for an example in the manufacturing cell formation problem field. The robust optimization methodology uses scenario-based approaches to deal with the uncertainty, where the scenarios correspond to the possible realizations of the uncertainty. The specification of a set of scenarios provides a way to incorporate different attitudes toward risk, relaxing the assumptions of knowing the probability distribution of uncertain data, thereby making a difference to *stochastic programming*. Defining the set of scenarios is a difficult task. It requires for the decision maker the dual task of identifying the main factors causing uncertainty and describes the relationship between them. This process helps reduce the number of scenarios, eliminating those that are unrealistic, favoring its analysis.

The objective of the robust optimization approach in our disassembly cell formation problem is to design a cell configuration that performs well across several potential realizations of uncertainty over a specified planning horizon. Since the definition of good performance is context-dependent, there exist several definitions of robustness. Particularly

relevant to our research are those provided by Kouvelis and Yu (1997) and Mulvey *et al.* (1995). The first describes the goal of robust optimization as finding a solution whose objective function remains close to the optimal solution for each of scenario, while the latter defines it as finding a solution that minimizes the largest deviation from optimality. Both approaches propose their respective *robust optimization modeling frameworks* (ROMF) which are summarized and adapted to our particular problem below.

2.8.4.2 Robust Probabilistic Models

Mulvey *et al.* (1995) propose an ROMF subject to noise with uncertain or incomplete data, integrating goal programming formulations with a scenario-based description of the problem data. Its framework was developed using the theory of stochastic programming for mathematical programming problems with continuous variables, in which they use appropriately defined penalty terms in the objective function to penalize decisions that are deviated from optimality in any scenario. From a modeling point of view, the definition of a robust optimization problem requires a set of future possible scenarios $S = \{1, 2, 3, \ldots, \Omega\}$, each with a probability of occurrence ρ_s such that $\sum_{s=1}^{\Omega} \rho_s = 1$. Also, for each scenario $s \in S$ let $\{B_s, C_s, g_s\}$ be the set of realizations for the coefficients of the control constraints. The ROMF distinguishes between a robust solution (with respect to optimality, if it remains 'close' to optimal for any realization of the scenario s), and a robust model (with respect to feasibility, if it remains 'almost' feasible for any realization of s). Since it is unlikely that a solution remains both optimal and feasible for any realization s, the ROMF allows the measurement of a tradeoff between model solution robustness. The following model formalizes a way to control (and measure) this tradeoff.

$$
\begin{array}{lll}
\text{Minimize} & f(x,y_1,y_2,\ldots,y_\Omega) + \lambda\, h(e_1,e_2,\ldots,e_\Omega) & \\
\text{Subject to} & A\,x = b & \\
 & B_s x + C_s y_s + e_s = g_s & \text{for all } s \in S \\
 & x \geq 0,\ y_s \geq 0 & \text{for all } s \in S
\end{array}
$$

The first term of the objective function measures solution-robustness, while the second term measures model-robustness. The λ is a weighting factor that measures the relative importance of obtaining a model-robust solution versus a solution-robust solution. For instance, if $\lambda = 0$, the objective function minimize the term $f(\cdot)$ and the solution may be infeasible for some scenarios; whereas, if λ is assigned to be sufficiently large, the term $h(\cdot)$

dominates the objective function and results in a higher cost. Note that there is a variety of choice for the cost function $f(x,y_1,y_2,\ldots,y_\Omega)$ and the feasibility penalty function $h(e_1,e_2,\ldots,e_\Omega)$ used to penalize violations of the control constraints. It is, however, necessary that the chosen functions lead to consistent preferences between alternative decisions, and that the weight assigned to λ reflects the preferences of the decision-maker. Typically, robust optimization formulations include a set of robust variables or design variables, whose values must remain the same across all scenarios being considered, and a set of control variables, which are subjected to adjustment once the uncertain parameters are observed. Additionally, the recourse variables are often included in the objective function to penalize infeasibility in complying with constraints in some scenarios. These variables represent the gaps (shortages or excesses) in the constraints.

2.8.4.3 Minimax Regret Models

Kouvelis and Yu (1997) propose a general robust optimization framework that applies the *minimax regret criterion* to differentiate the performance of different solutions on a given set of possible scenarios, which is mainly developed for models with discrete decision variables. The framework focuses on the problem of making a strong decision in a decision making environment with significant uncertainty. A *Robust Decision* is defined as one that performs well in all scenarios, and that offsets the worst case scenario. According to the authors, different approaches can be used to select a robust decision. By using the minimax regret criterion, the authors distinguish between two variations depending on how the regret is defined. In a minimization model, this can be measured as the difference between the cost of a decision that a decision-maker would make with the anticipated knowledge that a scenario occurs, and the cost of a decision made without such knowledge. The authors refer to this resulting decision as the *robust deviation decision.* Alternatively, the regret can be defined as the ratio of both quantities, serving as a surrogate measure of the percentage deviation of the robust solution regarding the optimal solution for any scenario. The authors refer to this resulting decision as the *relative robust decision.*

The choice of the robustness approach to be used crucially depends on the process used to generate the scenarios, and as such require a sharp intuition about the environment from the decision-maker. It should be noted that the concept of robustness adopted by Kouvelis and Yu (1997) corresponds to the concept of robust solution in the terminology of Mulvey *et al.* (1995). However, for both approaches, the robust model accuracy is as good as the estimated

values for each scenario undertaken by the decision maker. If the range of parameters does not cover all potential values or it includes unrealistic ranges, the robust solution obtained could be of poor quality.

In a specific situation, one or all of robustness criteria can be applied, however, the relative robustness criterion should be used in environments where both the performance of a simple optimal scenario fluctuates in a wide range of values or the performance of a decision through the scenarios is highly variable. Thus, to obtain a robust formulation for the disassembly cell formation problem under the minimax regret criterion, we may consider obtaining a relative robust decision, which exhibits *the best worst-case percentage deviation from optimality* among all feasible decisions on all scenarios.

2.8.4.4 Prior Cell Formation Problems with Uncertainty

Considering uncertainty of production demands and product mix, Dahel and Smith (1993) proposed a bi-criteria optimization model with design flexibility as one of the criteria to accommodate possible production demand changes. Based on the random nature of product mix, Seiffodini (1990) and Harhalakis *et al.* (1994) addressed the issue of robustness of cellular manufacturing systems. By assigning probabilities to discrete product mix and to the associated machine-component incidence matrix, Seiffodini (1990) developed a probabilistic cell clustering model to address possible changes of product mix. Harhalakis *et al.* (1994) discussed cell formation problems under random product demand and developed a mathematical programming model to minimize the expected inter-cell material handling cost. Cao and Chen (2005) consider a system configuration problem with product demands expressed in a number of probabilistic scenarios. They develop an optimization model integrating cell formation and part allocation is developed to generate a robust system configuration to minimize machine cost and expected inter-cell material handling cost, and propose a two-stage Tabu search based heuristic algorithm to solve it.

2.8.5 Other Approaches to Handle Variability in CFP

Safaei *et al.* (2007) propose a fuzzy programming based approach to design a cellular manufacturing system under dynamic and uncertain conditions. Virtual cellular manufacturing (VCM) has been proposed as an alternative to CMs, for functional layout

settings where a conversion to CMs is not feasible from a technical or financial perspective. Instead of a physical reallocation of machines – as in CMS – VCM aims to reduce set-up times by grouping similar jobs in production planning and control.

2.9 Past Doctoral Dissertations on Reverse Logistics

There have been a number of PhD theses on the field of recovery networks and reverse logistics leading the current trends of research in the area. A recent study performed by Stock and Broadus (2006) reveals that research in reverse logistics were among the ten main topics for doctoral research in supply chain management and logistics related areas from 1999 to 2004. In this section a sample of reverse logistics related dissertations from 1995 to 2009 are described in a chronological order. As expected, each of the 15 reviewed PhD dissertations dealt with a set of particular issues within reverse logistics. A diversity of subjects can be found as recycling systems, inventory control of remanufacturing, recovery strategies, network design and vehicle routing. Furthermore, most of the theses made a link with practice. The theses included studies on the electronics and communication industries (IBM, Ericsson), automobile industry (BMW and Volkswagen), and packaging industry (Tekpak), among others. Most of the reviewed thesis corresponds mainly to European universities; however, a good number of recent dissertations are from USA universities.

(1995) Thierry studied the impact of product recovery management in the electronics and car industries. Special attention was given to the impact on product design, logistics, and relations between actors. The author made use of two real life cases: a copy machine remanufacturing company and product recovery by the car manufacturer BMW.

(1995) Jahre investigated the performance of collection and recycling systems of household waste, with specific emphasizes on packaging materials. Two main aspects were taken into account: the degree of separation at the source and co-collection. The author employed data provided by the European Recovery and Recycling Association (ERRA).

(1997) Van der Laan analyzed the effects of remanufacturing on inventory control. Special attention was given to coordination mechanisms between the manufacturing and remanufacturing operations and to which actions deal to cost-efficiency. The findings were tested with real data from Volkswagen on car parts remanufacturing.

(1998) Krikke addressed the determination of recovery strategies and the logistics network design. For each of the issues, a case study was discussed in detail. On the recovery strategies, for the recycling of computer monitors, the author considered Roteb, the municipal waste company of the city of Rotterdam, in the Netherlands. On logistics network design, the author dealt with the copier reverse logistics network of an international company, with headquarters in Venlo, the Netherlands.

(2000) Fleischmann dealt with quantitative models for reverse logistics network design and for inventory management with returns. In particular, the thesis inquired under which conditions a network can be split into two separate networks, the forward and the reverse. In addition, the practical issues arising from reverse logistics were illustrated with a case study at IBM.

(2000) Newton presents robust approach for planning the strategic infrastructure of a reverse production system (RPS). The proposed algorithm generates a strategic design of the infrastructure which includes the location of collection and production facilities, the production equipments to be installed at the production facilities and the collection routing of raw materials. The author considers the robust optimization and the bi-level programming.

(2001) Beullens focused on the use of Operations Research (OR) tools for supporting the facility location decision, process planning and vehicle routing in reverse logistics. All the issues were illustrated with real life examples.

(2001) Kobeissi considered the evaluation of options for recovery of end of-life products. The thesis described the required resources and the different activities of the evaluation process. The focus was on the collection strategies, taking into account the product, or the geographic zone of collection. The measures of performance were maximization of profit and the satisfaction of the customer. Besides this, particular informational issues were considered as follows: how to integrate information in the collection schemes and what is the impact on efficiency.

(2002) Brodin analyzed the influence of both the product and the relationship between actors on the efficiency of logistics systems for recycling. To do so, the author made use of 1) a case study of the electronics recycling in the Netherlands; 2) a survey of the Swedish recycling market; and 3) interviews with Ericsson, Telia (large customer of telecommunications equipment), Stena Technoworld (recycling company) and Tekpak (a US package producer).

(2002) French (2002) explores reuse practices in plants within the process industry, focusing on those that produce homogeneous products. The sources of returned products included both internal sources, such as obsolete materials, and external sources, such as returns due to shipping errors. The reuse options were also studied. Both the sources and options were analyzed against a number of descriptor variables to understand why a given firm might have certain sources or reuse options. The results indicated that for homogeneous product producers, a larger percentage of returns are from internal rather than external sources. Common internal returns include out-of shelf-life products, obsolete products, partial container returns and rework. Common reuse options are: dispose or destroy, repackage, blend into the same or a similar product, and find a customer or market.

(2003) Alshamrani study the integration of forward and reverse movement of products at the routing level for a system consisting of a depot and a number of geographical dispersed customers who order quantities of a single product every period that need to be recovered in the following period (e.g. returnable containers, blood for hospitals). The problems considered under a multi-period planning horizon where future customer demands are uncertain.

(2003) Brito presents a research thesis designed to provide a better understanding of reverse logistics. She brought insights to reverse logistics decision making and laid down a theoretical ground for the development of reverse logistics as a research field. This thesis aimed at a) structuring reverse logistics as a research field, b) a better understanding of reverse logistics practices, c) structuring and supporting reverse logistics decision-making, and d) conjecturing about the future development of reverse logistics as a research field.

(2005) Soto proposes a collaborative production model for strategic and tactical planning, within a remanufacturing environment for the Spanish editorial sector. He identified a lack of research in business planning combining a collaborative approach for multiple products and factories with the introduction of the returned parts and assemblies in production process. The current competitive situation of business requires the use of models where these factors are combined. The author propose a mathematical model to solve the production collaborative planning taking into account the following elements: a) a supply chain that consists on several factories, a recovery center and a distribution center, b) multiple periods, products, production processes and materials, c) a Modified Bill of Materials (taking into account the

returned parts and assemblies). The benefits of this model are to help the companies to do the strategic and tactical production planning in the presence of returns and collaborating between them. Finally, he presents two small examples that represent the potential of these benefits.

(2006) Serrato-Garcia present a quantitative analysis of an *outsourcing* system in RL based on a Markov model.

(2006) Du addresses the analysis of reverse network design that deal with the returns that require post-sale service. An RL facility location problem with two objectives is proposed: minimization of the overall costs and minimization of the total tardiness of cycle time. The solution approach designed for solving this bi-objective optimization model consists of a combination of three methods: scatter search, the constraint method and the dual simplex method.

(2006) Sastry addresses the special reverse supply chain performance measurement problem faced by electronic companies. The research objective of this dissertation was: "To develop a quantitative methodology for evaluating the reverse supply chain performance in the consumer electronics industry, to maximize revenue within given technical and environmental constraints". The methodology was based on a composite reverse logistics overall performance index (RLOPI) to benchmark organizational performance across industry.

(2009) Topcu studied decision-making and modeling issues that arise in facility and warehouse designs in a remanufacturing context. The models include uncertainty in yield rates and demand, reconfigurable and flexible designs, interdependencies between returned products, and type of inventory control system. A mixed-integer, multi-period and multi-component stochastic programming recourse optimization model has been developed which identifies optimal schedules of internal, external, and reconfigured amounts of inventory space for a given time period. In order to better emulate a generalized remanufacturing facility with random receiving patterns, component yields, and refurbished demand over multiple time periods, a Monte Carlo simulation model has been developed. Finally, a heuristic approach based on a multi-dimensional golden section search algorithm is implemented to identify the optimal storage capacities and reconfiguration decisions that minimize long-term expected total costs.

Chapter 3

Problems Description

The aim of this chapter is to describe in more detail the three problems that are addressed in this thesis. Section 3.1 describes the WEEE collection problem in Galicia and discusses the need for its redesign. Section 3.2 describes the vehicle routing problem with split loads and date windows and discusses its importance in a real-world problem like the collection of WEEE. Finally, Section 3.3 describes why the use of a cellular configuration could improve the performance of a disassembly system.

3.1 Design of a Recovery Network for WEEE Collection

The strong increase of solid waste generation is a matter of concern in most industrialized countries. In Europe, electronic waste is estimated to reach 12 million tons by 2015. This is the equivalent of approximately 14 kg per person per year (Goosey, 2004). In recent times, over 90% of the electronic waste still ends up in landfills, causing serious environmental problems. In response to this growing problem, the European Union Directive 2002/96/EC (EU, 2003) on Waste of Electrical and Electronic Equipment (WEEE) has imposed to all member states the development of legislations based on the principle of *extended producer responsibility* to finance the treatment, recovery and recycling of all types of electronic waste.

In Spain the directive came into effect on February 25 2005 by The Royal Decree 208/2005. The adoption of this legislation has led to essential changes in the field of electronic scrap recycling. Particularly, producers and importers of large and small electrical appliances are affected by the new normative, since they will have to manage over 75% of the total electronic waste. To cope with this new responsibility, companies have come together to create a number of *Collective Management Systems* (CMS) to be responsible for the operation of collecting and recycling systems. The ECOLEC Foundation (hereafter

referred as the CMS) is one of the collective management systems with a higher percentage of WEEE collection: from category I electronic waste, which includes large electrical appliances, the CMS collects more than half of those generated in all Spain.

3.1.1 The WEEE Collection Problem in Galicia

In the region of Galicia (settled in northwestern Spain, with a population close to 3 million inhabitants) more than 3,239 tons of WEEE were collected in 2008. Currently, the management of the recovery network is in charge of third part logistics service providers (3PL) recruited by the central coordination of the CMS, which are responsible for the logistics activities in each district. Their activities include the harvesting of WEEE from stores to a depot managed by each 3PL, consolidation at the depot, and in some cases value-adding activities such as sorting and packaging.

The collection process can be described as follows: every store has a given capacity (directly related to sales volume) to collect WEEE. In turn, every store has allotted in its warehouse a certain space for WEEE collected. Whenever this space is about to be saturated, the store manager calls to the coordination call centre so that these items be taken off by the corresponding depot. Based on historical records, it is assumed that the average number of items that a given store has available for its delivery whenever it is visited is known and constant over the planning horizon. Each depot holds a heterogeneous fleet of capacitated vehicles, which are used in the harvesting of WEEE. The collection should be completed within the period of one week after receiving the call from the store.

To complete the collection on the 707 stores and collection points that the CMS currently serves in Galicia, a network consisting of seven depots is currently used (Fig. 3.1); such depots have been located and sized as a result of a historical evolution of the CMS relationships with their 3PL. Consequently, the recovery network has been developed on an ad hoc basis, where collection and storage activities are mainly based on the limited capabilities and available resources of the 3PLs without any detailed assessment of the environmental and economical benefits of the recovery activities. Therefore, the current management of the recovery network provides to the CMS just with a superficial understanding of the products they are recovering, isolating the collection and recycling activities. Furthermore, the CMS is now faced with the challenge to improve their collection activities to recover a larger proportion of WEEE at a reasonable cost and at the same time to meet the ever-increasing number of legislative requirements. Motivated by these facts, the

central coordination of the CMS has considered redesigning the current network configuration to be more involved in the collection process.

The potential locations of candidate depots (which would replace the current ones in leasing) are to be selected from a list of 20 candidate sites that have been predetermined based on subjective factors such as local laws, community attitudes, availability of resources, and proximity of transportation networks, among others. From there, the items collected will be directly shipped to one of the two recovery plants, where they will be disassembled in its basic components to remove the hazardous and polluting elements from those that can be recycled or reused.

3.1.2 Problem Remarks

Many economic and environmental benefits are achieved by electronic waste recycling. A recent study on the environmental impact of European regulations on WEEE recycling (Barba-Gutiérrez *et al.*, 2008) shows that inefficient recovery networks could cause even greater environmental impacts in transport harvesting than the ones saved by recycling practices. From a logistical point of view, the design of a recovery network has two main components: the location of collection depots and for each one of them, the design of the harvesting routes between its corresponding stores. Both problems fall into the reverse logistics management field. The first one aims to select which depots must be opened within a list of candidate sites, and to allocate collection points (stores) to depots. The second relates to the design of harvesting routes with capacitated vehicles. Although these two problems are interrelated, their analysis is often performed in a separate way, as we do in our research. Reverse logistics literature contains examples and case studies on each one individually, for example Shih (2001), Hu *et al.* (2002), Listes and Dekker (2005), Lieckens and Vandaele (2007) and Aras *et al.* (2008) studied the location of collection depots, while Schultmann *et al.* (2005), Blanc *et al.* (2006), and Kim *et al.* (2009) analyzed the collection routing problem.

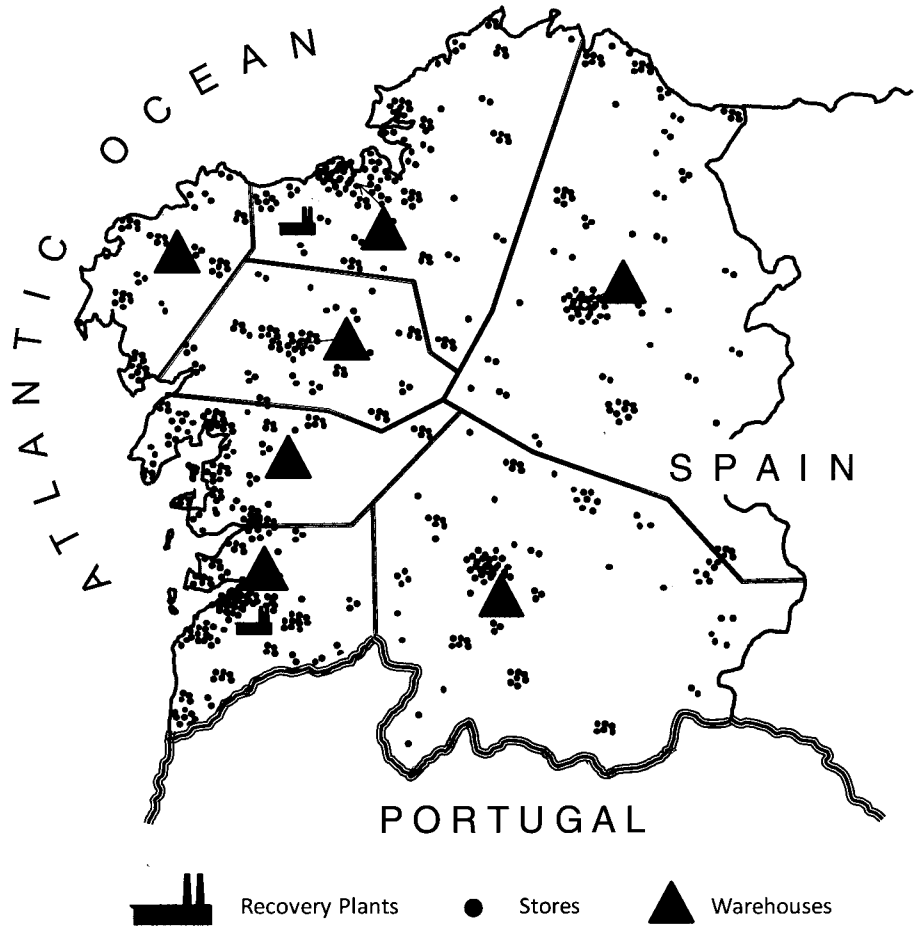


Figure 3.1 Geographical location of Galicia, current depots and collection points.

3.2 The Vehicle Routing Problem with Split Loads and Date Windows

The vehicle routing problem (VRP) is one of the most studied in the combinatorial optimization field and in the logistics literature. There are different variants of the VRP that can be characterized by the type of fleet, the number of depots, or the type of operations involved, among others. While in recent years, there has been an increasing interest towards so-called rich VRP models that include important issues arising from real-world applications, the literature on vehicle routing for reverse logistics is still very scarce. Most of the papers

address specific problems and case studies, while theoretical contributions mainly deal with the VRP with simultaneous pick-up and delivery, or the VRP with backhauls.

This part of the research focuses on the study of a VRP for reverse logistics. It specifically addresses the problem of designing routes for the collection of Waste of Electric and Electronic Equipment (WEEE). The motivation for this study is twofold:

1. First, the fact that the majority of European countries have recently implemented the European Union Directive 2002/96/EC (EU, 2003) on WEEE, whose general purpose is to prevent the creation of electrical and electronic waste and to promote reuse, recycling and the other forms of recovery in order to reduce final disposal.
2. Second, to generalize and extend the vehicle routing problem faced in the case of study performed in the northwestern Spain, which was depicted in Section 3.1. There the collection reports (i.e. number of visits that each truck made to the customers and the number of items collected in each visit) were taken as the input data to the routing problem. Thus based on historical records, it was assumed that the average number of items that a given store has available for its delivery whenever it is visited is known. However, given that every store has allotted in its warehouse a certain space for WEEE collected, whenever this space is about to be saturated the store manager calls to the coordination call center so that these items be taken off by the corresponding depot. The depot, that holds a heterogeneous fleet of capacitated vehicles used in the harvesting of WEEE, must collect all the items from the store within the period of one week after receiving the call from the store. Usually, the amount of end-of-life items to be collected (hereafter referred to as the demand) of each customer can be greater than the capacity of the largest vehicle in the fleet; as a result the demand of a single customer may be split to be fulfilled by more than one route. Thus, each customer can be visited more than once on the same day. Therefore the operational decision includes how the demand of each customer should be split to be collected by the capacitated vehicles.

Given that this is a real-world problem practical restrictions are imposed, like the availability of a fixed and heterogeneous fleet of capacitated vehicles with special features used in the harvesting of WEEE from a set of customers, the existence of a maximum operation time for each vehicle, which limit the number of tours that each vehicle can perform. It should be noted that collection orders to customers are triggered by a call from

them, and that with each order there is an associated demand to be fulfilled and a range of dates within the collection should take place. In Fig. 3.2 the problem structure is depicted.

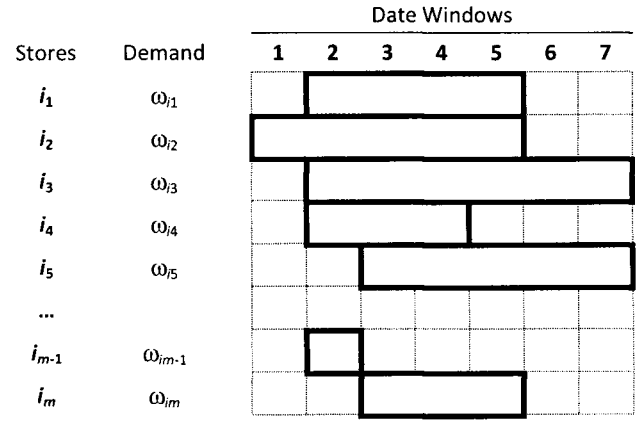


Figure 3.2 Problem illustration of the VRP-SLDW. Suppose that there are m stores that need to be visited, we known the amount of demand to be collected and the interval of days within the collection should take place.

This problem is characterized by four variants of the VRP that have been independently studied in the logistics literature, including: (1) the use of a heterogeneous fleet of vehicles, (2) customers with high demand that may be split to fit the vehicle capacity, (3) the fact that a same vehicle may be assigned to more than one route, and (4) the agreement of a time interval for visiting customers. These characteristics make the problem complex and interesting, while complicating its classification within the standard VRP literature. We label this variant of the VRP as the *Vehicle Routing Problem with Split Loads and Date Windows* (VRP-SLDW).

3.3 Disassembly Cell Formation Problems

New trends in environmental conscious manufacturing and reverse logistics have motivated a significant effort to improve the performance of several recovery systems. Product recovery is an important activity either to obtain useful components with economic value through reuse, remanufacturing or recycling, or to remove hazardous components in compliance with safety regulations. Regardless the final destination for the collected products, disassembly

operations are required and must be optimized to improve the overall performance of a recovery system.

The use of Cellular Manufacturing Systems (CMS) based in the Group Technology paradigm, have proved to give good results in typical manufacturing enterprises. Reported benefits include reduced set-up times, material handling cost, in-process inventory, better production efficiency and quality as well as market response time (Jayaswal and Adil 2004; Nsakanda *et al.* 2006; Chakravorty and Hales 2008). Most of these benefits come from the fact that the cells have a certain amount of functional autonomy, facilitating better control on the shop floor. These benefits can be extended to recovery systems, specifically to the disassembly process if we consider the use of a cellular configuration to take into account the similarities in the disassembly tasks. As is the case in manufacturing, when only one type of product is to be processed and the volume is high, a dedicated disassembly line may be justified. However, disassembly processes are characterized by the arrival of several similar types of products to be dismantled, each one with a low-medium volume. Hence, the use of a cellular configuration is justified.

3.3.1 Robust and Reconfigurable Disassembly Cells

This part of the research focuses on the study of cell formation problems for disassembly operations in reverse logistics. It specifically addresses the problem of designing robust and reconfigurable disassembly cells to deal with two different kinds of demand variability: probabilistic and deterministic. In the reconfigurable approach the product demand varies form period to period in a *deterministic manner.* Due to this dynamic condition of the demand the best cells configuration for one period may not be efficient or even feasible for subsequent periods, and some redesigns would be required; for instance to change the number and allocation of machines to cells from period to period, to adjust to demand variability. The main assumption that supports this approach states that although the demand for each period is known a priori, finding an optimal configuration for each period could be prohibitively expensive. On the other hand, in the robust approach the product demand varies in a *random manner.* However, this variation can be described in a number of probabilistic scenarios with a given occurrence probability. The aim of this approach is to obtain a cellular design in which the machines remain constant and do not move, only the flow of products changes once the demand has been observed.

The motivation for this study is based on the fact that, contrary to the assembly process, in reverse logistic systems most of the recovered products arrive to the disassembly center in a variable quantity and quality. Therefore, there is a need to include such variability in a proactive manner to decision-making process.

It should be noted that the aim of this part of the research is to describe and formulate integer programming models and metaheuristic algorithms for both approaches (robust and reconfigurable); however we do not pretend to design a cellular manufacturing system under both dynamic and uncertain conditions as for example Safaei *et al.* (2007) did. We consider that although in both approaches the robustness and the ability to reconfigure respectively are indicators of the flexibility that has a cellular manufacturing system to manage demand variability, the election of the approach to use hardly depends of the assumptions that most reflect the manufacturing system reality. We refer the reader to Balakrishnan and Cheng (2007) for a detailed literature review on robust and reconfigurable cell formation problems.

Chapter 4

Mathematical Formulations

In this chapter we provide the mathematical formulations of the studied problems. Section 4.1 presents the mathematical formulations of the related depot location and collection routing problems. Section 4.2 presents the mathematical formulation of the vehicle routing problem with split loads and date windows and comments on its computational complexity. Finally, in Section 4.3 four integer programming formulations for the disassembly cell formation problem and its variations are presented.

4.1 Modeling the Recovery Network for WEEE Collection

Despite of the fact that facility location (Melo *et al.*, 2009) and vehicle routing (Toth and Vigo, 2002) problems are closely interrelated, given the algorithmic complexity they tend to be addressed independently in literature. Currently there is a general discussion about the advantages and disadvantages of dealing jointly with the location-routing problem. On one hand, some authors point out that when the routes are stable for long periods, solving the combined location-routing problem leads to better solutions than solving them separately without considering their relationship, thus avoiding the risk of sub-optimization. On the other hand, different authors claim inappropriate to combine the location and routing problems in the same planning framework due to the difference in their planning horizons. We refer to (Min *et al.*, 1998; and Nagy and Salhi, 2007) for a detailed literature review on location-routing problems.

In reverse logistics, there are certain features that complicate the treatment of the problem as a combined location-routing problem, such as the inherent variability and uncertainty in the amount and time in which WEEE is returned by the final consumer. Therefore, this

research aims to solve the problem in a hierarchical and sequential manner, solving and analyzing the related problems of location and routing as well as their interrelationships between different nodal configurations and routes.

4.1.1 Modeling the Facility Location Problem

A considerable number of articles reporting reverse logistics facility location issues have been published (e.g. Hu *et al.*, 2002; Listes and Dekker, 2005; Lee *et al.*, 2009). Most of these location models preserve the same characteristics of the forward logistics models, minimizing the overall system linear cost, and proposing slight modifications and extensions to take into account special features such as the convergent structure of the network as well as technical rates for reuse feasibility or effective recovery.

The facility location model for our particular problem must be able to indicate the nodal configuration of points on a WEEE recovery network. It means to determine the number, size and location of regional depots, where the WEEE is temporarily stored on its way to the disassembly plants. For a formal definition of the problem, let $I = \{i_1, i_2, ..., i_I\}$ be the set of collecting points (stores) whose location and demand (i.e. the amount of WEEE collected in a given store) are well known, and $P = \{p_1, p_2, ..., p_P\}$ stands for the set of disassembly plants. Due to technical limitations or capacity conditions, not every plant can receive all the waste generated in the region; so it must be ensured that no more items than its capacity will be sent to plant p; similarly, there are conditions on the contract, indicating that any plant should receive at least some amount of return for processing, so we will ensure that plant p, will be sent at least some percentage L_p of WEEE generated. A set of $W = \{w_1, w_2, ..., w_W\}$ potential locations for regional depots, and a set of $K = \{k_1, k_2, ..., k_K\}$ vehicle types that may be assigned to the depots are also considered. The capacity to be installed at each depot is decided by the model considering both the amount of WEEE moving through the depot, and the turnover γ_w in the depot. For every candidate depot to be opened, the annual fixed cost per unit of capacity to be installed (leasing cost) is known. For each vehicle k we know its maximum load capacity, the set of stores that vehicle k may visit (because of street access restrictions), as well as its fixed annual cost and its variable travelling cost. The problem consists of finding the best combination of (1) the set of depots to open, within the list of candidate sites, (2) their capacities, (3) the types of vehicles to be allotted to the depots, and (4) the stores allocation to the depots, in order to minimize the total cost. To facilitate the

description of the mathematical model proposed here, the notation used is described in Table 4.1.

Table 4.1 Notation used for the location model.

– Sets:	
I	Set of customers.
K	Set of vehicles types.
P	Set of recovery plants.
W	Set of depots.
– Parameters:	
O_i	Total amount (kilograms) of WEEE per year available in store i for its collection.
τ_w	Average number of days (defined by the decision maker as a target goal) that at most collected items must remain in depot w.
DA	Days per year that the system works.
ρ	Average weight of collected items.
f_w	Fixed cost per year of installed capacity (€/kilograms) in depot w.
q^1_k	Capacity (in kilograms) of vehicle type k.
q^2_k	Capacity (in items) of vehicle type k.
π_k	Fixed annual cost (€/year) for vehicle type k (e.g. driver, amortization, maintenance).
λ_k	Variable travelling cost (€/km) for vehicle type k.
t_k	Average working hours per day for vehicle type k.
υ_k	Average speed for vehicle type k (km/h).
μ	Average loading/unloading time per tour.
U_p	Upper processing rate in plant p.
L_p	Lower processing rate for plant p.
ω^1_i	Average amount (kilograms) of WEEE available in store i for its collection whenever it calls to facility r.
ω^2_i	Average amount (items) of WEEE available in store i for its collection whenever it calls to facility r.
B	Binary matrix where $b_{ik} = 1$ means that vehicle type k is able to access to the location of store i, and 0 otherwise.
d_{ab}	Distance between a and b, where $ab \in [iw, ip, ww', wp]$.
γ_w	Turnover in depot w, DA/τ_w.
Q_{ik}	Number of trips per call that each vehicle type k should travel in the harvesting to store i, if such store were served by vehicles type k, where $Q_{ik} = max(\lceil \omega^1_i / q^1_k \rceil, \lceil \omega^2_i / q^2_k \rceil)$.
M_{ik}	Number of trips per year required for the harvesting of WEEE for store i if vehicle type k is used, where $M_{ik} = (O_i \cdot Q_{ik})/\omega^1_i$
– Binary decision variables $O(IK(W+P))$:	
Z_{irk}	1 if facility $r \in W \cup P$ collects the items from store i in vehicle k, and 0 otherwise.
– Integer decision variables $O(K(W+1)(P+W))$:	
N_{wrk}	Number of trips per year to travel from depot w to facility $r \in W \cup P$, $r \neq w$, in vehicle type k, to transport X_{wrk}.
V_{rk}	Number of vehicles type k to allot to facility $r \in W \cup P$.
– Continuous decision variables $O(W(K(W+P)+1))$:	
X_{wrk}	Annual amount of kilograms to be shipped from depot w to facility $r \in W \cup P$, $r \neq w$, in vehicle type k.
S_w	Capacity to install at depot w.

The objective of the proposed model is to minimize a linear cost function of the logistics network. The total reverse logistics cost involved in the objective function includes four

major cost items: (4.1.1) total leasing cost, (4.1.2) fixed cost of vehicles, (4.1.3) total transportation cost for harvesting at stores, and (4.1.4) total transportation cost between depots and recovery plants, as shown in the following objective function equations:

$$\textit{Minimize}\quad f(x) = \sum_{w\in W} f_w \cdot S_w \qquad (4.1.1)$$

$$+ \sum_{r\in W\cup P}\sum_{k\in K} \pi_k \cdot V_{rk} \qquad (4.1.2)$$

$$+ \sum_{i\in I}\sum_{r\in W\cup P}\sum_{k\in K} (2d_{ir}\lambda_k)\cdot M_{ik}\cdot Z_{irk} \qquad (4.1.3)$$

$$+ \sum_{w\in W}\sum_{r\in W\cup P-\{w\}}\sum_{k\in K} (2d_{wr}\lambda_k)\cdot N_{wrk} \qquad (4.1.4)$$

Considering the required conditions or specifications for WEEE collection and vehicles capacities, and other requirements limited by operating capacities, nine groups of constraints were defined:

$$\sum_{r\in W\cup P}\sum_{k\in K} Z_{irk}\cdot b_{ik} = 1 \qquad \forall i\in I \qquad (4.2)$$

$$\sum_{w\in W}\sum_{p\in P}\sum_{k\in K} X_{wpk} + \sum_{i\in I}\sum_{p\in P}\sum_{k\in K} O_i\cdot Z_{ipk} = \sum_{i\in I} O_i \qquad (4.3)$$

$$\sum_{i\in I}\sum_{k\in K} O_i\cdot Z_{iwk} + \sum_{w'\in W-\{w\}}\sum_{k\in K} X_{w'wk} = \sum_{k\in K}\sum_{p\in P} X_{wpk} + \sum_{w'\in W-\{w\}}\sum_{k\in K} X_{ww'k} \qquad \forall w\in W \qquad (4.4)$$

$$\sum_{i\in I}\sum_{k\in K} O_i\cdot Z_{iwk} + \sum_{w'\in W-\{w\}}\sum_{k\in K} X_{w'wk} \le S_w\cdot\gamma_w \qquad \forall w\in W \qquad (4.5)$$

$$\sum_{k\in K} X_{wpk} \le U_p\cdot\sum_{i\in I}\sum_{k\in K} O_i\cdot Z_{iwk} \qquad \forall w\in W, p\in P \qquad (4.6)$$

$$\sum_{w\in W}\sum_{k\in K} X_{wpk} + \sum_{i\in I}\sum_{k\in K} O_i\cdot Z_{ipk} \ge L_p\cdot\sum_{i\in I} O_i \qquad \forall p\in P \qquad (4.7)$$

$$t_k\cdot DA\cdot V_{rk} \ge \sum_{i\in I}\left(\mu + \frac{2d_{ir}}{\upsilon_k}\right)\cdot M_{ik}\cdot Z_{irk} + \sum_{w\in W-\{r\}}\left(\mu + \frac{2d_{wr}}{\upsilon_k}\right)\cdot N_{w,r,k} \qquad \forall r\in W\cup P, k\in K \qquad (4.8)$$

$$V_{pk} = 0 \qquad \forall p\cup P, k\in K : k < 3 \qquad (4.9)$$

$$N_{wrk} \ge \max\left(\frac{X_{wrk}}{q_k^1}, \frac{X_{wrk}}{\rho\cdot q_k^2}\right) \qquad \forall w\in W, r\in W\cup P, k\in K \qquad (4.10)$$

$$Z_{irk}\in\{0,1\};\quad N_{wrk}, V_{rk}\in Z_+;\quad X_{wrk}, S_w \ge 0 \qquad (4.11)$$

Constraint (4.2) ensures the harvesting of all kilograms of WEEE collected by store *i* with a single vehicle type *k*, able to access to store *i*, belonging either to a depot *w* or a recovery plant *p*. Constraint (4.3) ensures that the total amount of kilograms generated among all the

stores arrive to any recovery plant p, either directly from the stores i or by means of a depot w. Constraint (4.4) ensures that all returns collected by the depot w have been sent to a recovery plant p or to another depot w'. Constraint (4.5) determines the capacity to be installed at each depot w. Constraints (4.6) and (4.7) guarantee the capacity conditions for disassembly plants. Constraint (4.8) determines an upper bound for the number of vehicles k required by facility $r \in W \cup P$. Constraint (4.9) avoids allotting type 1 and 2 vehicles (usually used for collection at stores) to any plant p. Constraint (4.10) sets a lower bound for N_{wrk} variables, where $X_{wrk}/q^1{}_k$ is the number of trips to perform from w to r in k if weight capacity is considered, and $X_{wrk}/\rho q^2{}_k$ accounts for the number of trips to complete from w to r in vehicle type k if unit capacity is considered. Finally, constraint (4.11) deals with the nature of the decision variables. The problem is formulated as a mixed integer linear programming with $\{(W(2 + P + 2K(W + P) + K) + P(2K + 1) + I + 1\}$ constraints.

4.1.2 Modeling the Collection Routing Problem

Contrary to the reverse logistics facility location problem, literature on collection routing problem is merely scarce. Most of the references address specific problems and case studies (Schultmann *et al.*, 2006; Blanc *et al.*, 2006; Kim *et al.*, 2009), while theoretical contributions deals mainly with Vehicle Routing Problems (VRP) with simultaneous pickup and delivery (Dethloff, 2001; Dell'Amico *et al.*, 2006).

Two of the main assumptions under which the classic VRP model is founded are the use of a homogeneous fleet of vehicles and the fact that the demand of each node is smaller than the capacity of the vehicles used. Note however that in the context of WEEE collection, both assumptions should be relaxed given the way in which these systems work, where the amount of WEEE generated by the collection point tends to be greater than the capacity of vehicles used in the harvesting, more over in WEEE collection systems a heterogeneous fleet of vehicles with special characteristics is frequently used.

Formally speaking a customized model of the WEEE collection routing problem requires the definition of a directed graph $G = (V, A)$ where $V = \{v_0, v_1, ..., v_m\}$ is a set of vertices, and $A = \{(v_i, v_j): i \neq j\}$ is a set of arcs. Vertex v_0 denotes the depot at which a heterogeneous fleet of capacitated vehicle types has been allotted, and the remaining m vertices of V represent the cities to be visited. With every city there is associated a positive demand ω_i. For every pair of vertices $i,j \in \{V: i \neq j\}$ the distance d_{ij} between them is known. For each vehicle type k we

know the maximum load capacity, the set of stores that vehicle type k may visit, and the variable travelling cost of vehicle type k. The problem consist of designing a set of least cost vehicle routes in such a way that every route starts and ends at the depot, and every store $V \backslash \{v_0\}$ is visited exactly once by exactly one vehicle. Apart from these general limitations, the following side constraints must be satisfied: *i*) the total demand of every route should not exceed the vehicle capacity q_k used in such route; *ii*) the location of some stores restrict the type of vehicle to be used in routes, and therefore some vehicles cannot be used to serve a given set of stores. Notation used in our mathematical model is depicted in Table 4.2.

Table 4.2 Notation used in the routing model.

– Sets:	
V	Set of vertex, representing the union of all stores allotted to the depot plus the depot itself, where card(V) = m + 1.
A	Set of directed arcs (v_i, v_j), card(A) = n.
K	Set of vehicles type allotted to the depot.
– Parameters:	
ω_i	Amount of items deliver by store i (demand) every time the store is visited.
q_k	Capacity of vehicle type k.
λ_k	Travelling cost per kilometer for vehicle type k (includes fixed and variable cost per kilometer travelled).
d_{ij}	Distance associated with every arc $(i,j) \in A$, representing the travelling time required to move from vertex i to vertex j.
B	Binary matrix, where b_{ik} = 1 means that the vehicle type k can access to the location of store i, and 0 otherwise.
– Binary decision variables $O(Kn^2)$:	
x_{ijk}	1 if arc $(i,j) \in A$ is transverse within the optimal solution by vehicle type k, and 0 otherwise.
– Continuous decision variables $O(Kn)$:	
u_{jk}	Cumulative load in vehicle k after visit store j.

The objective of the problem is the transportation of the WEEE collected from the origins (stores) to the destinations (depots) through the most economic routes. Based on the transportation requirements several routing problems may emerge. The objective function (12) minimizes the total harvesting cost, composed of routing costs between visited vertices by vehicle type k in all routes.

$$Minimize \quad f(x) = \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} \lambda_k \cdot d_{ij} \cdot x_{ijk} \tag{4.12}$$

The model constraints are:

$$\sum_{k \in K} \sum_{i \in V \backslash \{j\}} b_{jk} \cdot x_{ijk} = 1 \qquad \forall j \in V \backslash \{0\} \tag{4.13}$$

$$\sum_{k\in K}\sum_{i\in V\setminus\{0\}} x_{0ik} = \sum_{k\in K}\sum_{i\in V\setminus\{0\}} x_{i0k} \tag{4.14}$$

$$\sum_{i\in V: i\neq j} x_{ijk} = \sum_{i\in V: i\neq j} x_{jik} \qquad \forall j \in V, k \in K \tag{4.15}$$

$$u_{jk} \geq M + \omega_j + M\cdot\left(x_{0jk} - 1\right) + \sum_{i\in \mathbf{V}\setminus\{0\}: i\neq j} \omega_i \cdot x_{ijk} \qquad \forall j \in V\setminus\{0\}, k \in K \tag{4.16}$$

$$u_{jk} \geq u_{ik} + \omega_j + \hat{M}\cdot\left(x_{ijk} - 1\right) + \left(\hat{M} - \omega_i - \omega_j\right)\cdot x_{jik} \qquad \forall i, j \in V\setminus\{0\}: i \neq j, k \in K \tag{4.17}$$

$$u_{ik} \geq M + \sum_{j\in V\setminus\{0\}: j\neq i} \omega_j \cdot x_{jik} \qquad \forall i \in V\setminus\{0\}, k \in K \tag{4.18}$$

$$u_{ik} \leq M + q_k - \sum_{j\in V\setminus\{0\}: j\neq i} \omega_j \cdot x_{ijk} \qquad \forall i \in V\setminus\{0\}, k \in K \tag{4.19}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i, j \in V, k \in K \tag{4.20}$$

Constraint (4.13) ensures that vertex j can be reached directly just from a single vertex $i \neq j$, with a single vehicle k capable of accessing vertex j. Constraint (4.14) guarantee that the number of tours leaving the depot is equal to the number of tours arriving to depot. Constraint (4.15) establishes the continuity conditions and ensures that the vehicle arriving to a given vertex is the same vehicle leaving such vertex. It should be noted that inequalities (4.16) to (4.19) are variations of the sub-tour elimination constraints, which were lifted following the procedure described by Desrochers and Laporte (1991) and Kara *et al.* (2004) to replace weaker requirements. Constraints (4.16) and (4.17) establish the heterogeneous fleet capacity conditions and sub-tour elimination constraints, where $M = 1 + \sum_{i\in V} \omega_i$ and $\hat{M} = M \times 1000$. Constraint (4.16) was lifted to replace the weaker requirement $u_{jk} \geq M + \omega_j$ if $x_{0jk} = 1$ for all $j \in V$, $k \in K$. Constraint (4.17) is a modification of the capacity cut constraints allowing the existence of stores with a greater demand than the capacity of the smaller vehicle type used. This constraint was lifted to replace the weaker requirement $u_{jk} \geq u_{ik} + \omega_j$ if $x_{ijk} = 1$ for all $j \in V$, $k \in K$. Constraints (4.18) and (4.19) provide the lower and upper bounds for the variable u_{ik}, which together with constraint (4.15) enforces the tour to end at the depot when the inclusion of some other vertex in the tour exceeds the capacity of vehicle k. Constraints (4.18) and (4.19) replace the weaker requirements $M \leq u_{ik} \leq M + q_k$ for all $i \in V$, $k \in K$. This formulation differs from the heterogeneous vehicle routing problem formulations presented by Yaman (2006), and enclose $\{2K(n^2+n-1) + n\}$ constraints.

4.2 Modeling the VRP-SLDW

Formally speaking the mathematical formulation of the Vehicle Routing Problem with Split Loads and Date Windows (VRP-SLDW) requires the definition of a directed graph $G = (V, E)$ where $V = \{v_0, v_1, \ldots, v_m\}$ is a set of vertices, and $E = \{(v_i, v_j): i \neq j\}$ is a set of arcs. Vertex v_0 denotes the depot at which a heterogeneous fleet of K capacitated vehicles has been allotted, while the remaining m vertices of V represent the customers to be visited. For every pair of vertices $i,j \in \{V: i \neq j\}$ the distance d_{ij} between them is known. For each vehicle k we know the maximum load capacity, the set of customers that vehicle k may visit (because street access restrictions), and the variable travelling cost of vehicle k. For the collection, the independent customers make use of a call system with a timely collection guarantee. Each collection order predefines the quantity of items w_i to be collected from customer i and the range of dates within which the collection should take place. The time interval to carry out the collection from a single customer is specified by its date window $[s_i, f_i]$, where s_i and f_i are respectively the first and last days within which the collection must take place, such that $s_i \leq f_i$. For all customers the service time μ_i is assumed constant, since most of the times each customer is served in a full truck load, being visited when it has at hand a sufficient number of items to be collected. For the sake of real practice, when the number of items to be collected from any customer i is less than a constant factor ε, those items are not collected and labeled as backorders to be considered in a following request. The notation used in the mathematical formulation is described in Table 4.3.

The mathematical model for the VRP-SLDW is given by:

$$Minimize \sum_{i \in \mathbf{V}} \sum_{j \in \mathbf{V}} \sum_{t \in \mathbf{T}} \sum_{k \in \mathbf{K}} \sum_{r=1}^{R} \lambda_k \cdot d_{ij} \cdot y_{ij}^{(tkr)} \tag{4.21}$$

subject to:

$$\sum_{i \in \mathbf{V} \setminus \{j\}} \sum_{r=1}^{R} y_{ij}^{(tkr)} \leq a_{jt} \cdot b_{jk} \qquad \forall j \in V \setminus \{0\}, t \in T, k \in K \tag{4.22}$$

$$\sum_{j \in \mathbf{V} \setminus \{0\}} y_{0j}^{(tkr)} \leq 1 \qquad \forall t \in T, k \in K, r = 1, \ldots, R \tag{4.23}$$

$$\sum_{i \in \mathbf{V} \setminus \{j\}} y_{ij}^{(tkr)} = \sum_{i \in \mathbf{V} \setminus \{j\}} y_{ji}^{(tkr)} \qquad \forall j \in V, t \in T, k \in K, r = 1, \ldots, R \tag{4.24}$$

$$x_j^{(tkr)} \leq w_j \cdot \sum_{i \in \mathbf{V} \setminus \{j\}} y_{ij}^{(tkr)} \qquad \forall j \in V \setminus \{0\}, t \in T, k \in K, r = 1, \ldots, R \tag{4.25}$$

$$x_j^{(tkr)} \geq \varepsilon \cdot \sum_{i \in \mathbf{V} \setminus \{j\}} y_{ij}^{(tkr)} \qquad \forall j \in V \setminus \{0\}, t \in T, k \in K, r = 1, \ldots, R \tag{4.26}$$

$$y_{ij}^{(t,k,r)} \geq y_{ij}^{(t,k,r+1)} \qquad \forall i,j \in V\backslash\{0\}, t \in T, k \in K, \; r = 1,\ldots,R-1 \tag{2.27}$$

$$\sum_{t\in\mathbf{T}}\sum_{k\in\mathbf{K}}\sum_{r=1}^{R} x_i^{(tkr)} + \sigma_i = w_i \qquad \forall i \in V\backslash\{0\} \tag{4.28}$$

$$\sum_{i\in\mathbf{V}}\sum_{j\in\mathbf{V}}\sum_{r=1}^{R}\left(t_{ij}+\mu_j\right)\cdot y_{ij}^{(tkr)} \leq L_k \qquad \forall t \in T, k \in K \tag{4.29}$$

$$u_j^{(tkr)} \geq u_i^{(tkr)} + x_j^{(tkr)} + Q_k \cdot \left(y_{ij}^{(tkr)} - 1\right) \qquad \forall i,j \in V\backslash\{0\}: i \neq j, t \in T, k \in K, \; r = 1,\ldots,R \tag{4.30}$$

$$x_i^{(tkr)} \leq u_i^{(tkr)} \leq Q_k \qquad \forall i \in V\backslash\{0\}, t \in T, k \in K, \; r = 1,\ldots,R \tag{4.31}$$

$$x_i^{(tkr)} \geq 0, u_i^{(tkr)} \geq 0, 0 \leq \sigma_i \leq \varepsilon \qquad \forall i \in V, t \in T, k \in K, r = 1,\ldots,R \tag{4.32}$$

$$y_{ij}^{(tkr)} \in \{0,1\} \qquad \forall i,j \in V, t \in T, k \in K, \; r = 1,\ldots,R \tag{4.33}$$

Table 4.3 Notation used in the mathematical model for the VRP-SLDW.

– Sets:	
V	Set of vertex, representing the union of all customers allotted to the depot plus the depot itself, where card(V) = m + 1.
E	Set of arcs (v_i, v_j), card(E) = n.
T	Set of days in the planning horizon.
K	Set of available vehicles.
– Parameters:	
w_i	Amount of items to be collected (demand) at customer $i \in V$.
ε	Minimum load to be collected at any customer i, to justify its visit.
d_{ij}	Travelling distance (km) from vertex i to vertex j.
t_{ij}	Travelling time (hrs) from vertex i to vertex j.
μ_i	Service time (hrs) at vertex $i \in V$.
Q_k	Capacity of vehicle k.
L_k	Maximum operation time for vehicle k per day.
λ_k	Travelling cost per kilometer for vehicle k.
R	Maximum number of tours that any vehicle can perform per day.
A	Binary matrix, where a_{it} = 1 means that customer i must be visited within the set of days defined by its date windows $\{t \in T: s_i \leq t \leq f_i\}$.
B	Binary matrix, where b_{ik} = 1 means that vehicle k is feasible for customer i, and 0 otherwise. It should be note that b_{0k} =1 for all $k \in K$.
– Binary decision variables O($KRTn$):	
$y_{ij}^{(tkr)}$	1 if arc $(i,j) \in E$ is transverse within the optimal solution by the tour r of vehicle k on day t, and 0 otherwise. In this sense, a route is defined by a set of three attributes (tkr).
– Integer decision variables O($m(KRT$ + 1)):	
$x_i^{(tkr)}$	Amount of items collected at customer i on route (tkr).
σ_i	Amount of items not collected from customer i, due to its economical infeasibility.
– Continuous decision variables O($KRTm$):	
$u_i^{(tkr)}$	Cumulative load in route (tkr) after visiting customer i.

The objective function (4.21) minimizes the total routing cost. Equations (4.22), (4.23) and (4.24) provide the connectivity and continuity constraints. Equation (4.25) and (4.26) specify collection bounds for each customer on any route. Equation (4.27) guarantees the proper use of tours in every route. Moreover, constraints (4.26) and (4.27) strength the formulation by restricting duplicate solutions, which reduce the number of nodes in the branch-and-bound tree. Equation (4.28) ensures the collection of customer's demand in one or more routes, or its consideration as backorder. Equation (4.29) sets the maximum service time for each vehicle, whereas equations (4.30) and (4.31) provide both the capacity restrictions and the sub-tour elimination constraints. Finally, equations (4.32) and (4.33) deal with the nature of the decision variables. Note that although $x_i^{(tkr)}$ and σ_i are declared as continuous, the formulation forces them to take integer values.

4.2.1 Problem Complexity

The problem modeled above can be characterized as a new variant of the VRP that incorporates several elements that have been independently studied in the literature. This problem is NP-Hard since it is a variant of the VRP that incorporates a larger number of decision elements. To show the difficulty of the problem, a trial example with just 10 customers totalizing 607 units of demand was solved: a set of 3 vehicles with capacities of 25, 30 and 50 units respectively, was allotted to the depot; each vehicle can carry at most 3 trips per day, without exceeding a 7 working hours per day on a period of the week. This sample problem results in a mixed integer linear programming model with 7,570 variables (6,300 of which are binary) and 21,333 constraints. The problem was modeled on AMPL and solved by CPLEX v.11.1 using an Acer PC with an AMD Athlon X2 Processor running at 1.90 GHz and 2 GB RAM. After 3 hours of computing time, the best solution found was 1,229.66, with a *relative mipgap* of 0.3008, and an *absolute mipgap* of 369.981.

4.3 Disassembly Cell Formation Problems Formulations

In Adenso-Díaz *et al.* (2006), after performing a literature review, they concluded that although most researchers recognize the need to identify disassembly families and design the system to benefit from existing similarities in products structure and tooling, no mathematical model for disassembly cell formation had been proposed. In this section we formulate and detail not only its proposed mathematical model, but also we propose three integer programming formulations for the corresponding reconfigurable and robust variations.

The notation that will be used in the mathematical formulations is described in Table 4.4.

Table 4.4 Notation used in mathematical formulations of disassembly cells.

– Sets:	
P	Set of product types.
Q	Set of disassembly operations.
Q_p	Subset of disassembly operations required by product p.
C	Set of cells to be formed.
M	Set of machine/tools.
T	Set of periods in the planning horizon.
S	Set of scenarios.
– Parameters:	
D^t_p	Demand for product type p in period t.
D^s_p	Demand for product type p under scenario s.
α_p	Cost to move one unit of product p between any two cells (intercellular material handling cost).
O_{pj}	Set of machine/tools required by j-th operation of product p.
θ_{pj}	Duration of j-th operation of product p.
f_{pj}	Relative frequency of j-th operation of product p. It should be noted that $0 \le f_{pj} \le 1$.
$\pi_{pjj'}$	Probability that the j'-th of product p be performed immediately after of its j-th operation ($j < j$'), where: $\pi_{pjj'} = f_{pj'} \prod_{j<l<j'} (1 - f_{pl})$.
H_m	Capacity of machine type m in time units.
β_m	Amortization cost of machine/tool type m.
γ_m	Relocation cost of machine type m.
ρ_s	Occurrence probability of scenario s.
U	Maximum number of disassembly tasks per cell.
L	Minimum number of disassembly tasks per cell.
– Binary decision variables:	
x^t_{pjc}	1 if the j-th operation of product p is assigned to cell c in period t, and 0 otherwise.
x^s_{pjc}	1 if the j-th operation of product p is assigned to cell c under scenario s, and 0 otherwise.
y^t_c	1 cell c is formed in period t, and 0 otherwise.
y_k	1 cell c is formed, and 0 otherwise.
– Integer decision variables:	
z_{mc}	Number of machine/tools of type m used in cell c.
z_{mct}	Number of machine/tools of type m used in cell c during period t.
z^+_{mct}	Number of machine/tools of type m added in cell c during period t.
z^-_{mct}	Number of machine/tools of type m removed in cell c during period t.

4.3.1 The Basic Disassembly Cell Formation Problem

For a formal definition of the Disassembly Cell Formation Problem (DCFP) let $M = \{m_1, m_2, \ldots, m_M\}$ and $P = \{p_1, p_2, \ldots, p_P\}$ be, respectively, the set of machine/tools (such as screwdrivers, shears for cutting material, mechanical multi-devices for grinding, separation and treatment of gases...) and the set of products (such as washing machines, dryers, ovens, refrigerators...) that need to be dismantled with the aim of obtain parts or materials of economic value and/or remove toxic elements to promote the proper disposition. Let $Q = \{q_1, q_2, \ldots, q_Q\}$ be the set of disassembly operations (such as removal of the printed circuit board and the power supply, or removal of condensers, mercury switches and batteries), and let $Q_p \subseteq Q$ be the set of disassembly operations required to dismantle product p. To this end, a product p and an operation q are not independent, and each pair $\{p, q\}$ represent a *disassembly task*. Let $N = \Sigma_{p \in P} \text{card}(Q_p)$ be the total number of disassembly tasks, which have to be grouped into a subset of cells $C = \{c_1, c_2, \ldots, c_C\}$. Both the amount of returned products (hereafter referred as the demand) and the physical conditions (hereafter referred as the quality) in which the product arrive to the disassembly center are known with certainty. For each product it is known the sequence in which the disassembly operations need to be performed. The optimal disassembly sequence can be obtained for example trough Adenso-Díaz *et al.* (2007). Two cost elements are considered: machine/tools acquisition (i.e. amortization) costs and material handling costs. Assuming that cell sizes are not large and that inter-cell distances are small, intercellular transportation costs are considered as surrogate of material handling costs. To keep cell sizes bounded a maximum number of operations per cell is imposed. Thus the objective of the DCFP consist on grouping the disassembly task and assign them to their corresponding cells with their resources so that: 1) total costs are minimized, 2) all recovered items are disassembled, and 3) the following restrictions are met: a) the total number of disassembly tasks assigned to a given cell must be between the corresponding lower and upper bounds, b) the operating time assigned to a machine must not exceed its capacity, and c) any task should be assigned to a single cell.

The following assumptions are also considered: for each machine/tool m both their capacities and amortization costs are known; for each disassembly operation q the set of machines required to carry it out, and the time required to perform such operation in a given product p are known. The uncertain physical state of the units to disassemble is considered through a relative frequency factor that represents the probability of a given task being required. The number of machines is determined by the solution to the model. However, it is

assumed that multiple units of the same machine type may be required in a single cell, thus machine duplication is allowed. The production capacity of all machines in the disassembly system should be sufficient to dismantle all products. The maximum (U) and minimum (L) number of tasks allowed per cell result in the maximum and minimum number of cells to form respectively: $C_{max} = \lfloor N/L \rfloor$ and $C_{min} = \lceil N/U \rceil$.

The model DCFP model proposed by Adenso-Díaz *et al.* (2006) is given by:

DCFP (Disassembly Cell Formation Problem)

$$\text{Minimize } TC = \sum_{m\in \mathbf{M}} \sum_{c\in \mathbf{C}} \beta_m \cdot z_{mc} + \sum_{t\in \mathbf{T}} \sum_{p\in \mathbf{P}} \sum_{j\in Q_p} \sum_{j<j'} \sum_{c\in \mathbf{C}} \alpha_p \cdot D_p \cdot f_{pj} \cdot \pi_{pjj'} \cdot x_{pjc} \cdot \left(1 - x_{pj'c}\right) \quad (4.34)$$

Subject to:

$$\sum_{c\in C} x_{pjc} = 1 \qquad \forall p \in P, j \in Q_p \quad (4.35)$$

$$L \cdot y_c \le \sum_{p\in P} \sum_{j\in Q_p} x_{pjc} \le U \cdot y_c \qquad \forall c \in C \quad (4.36)$$

$$\sum_{p\in P} \sum_{\{j:m\in O_{pj}\}} D_p \cdot \theta_{pj} \cdot f_{pj} \cdot x_{pjc} \le H_m \cdot z_{mc} \qquad \forall m \in M, c \in C \quad (4.37)$$

$$x_{pjc} \in \{0,1\} \qquad \forall p \in P, j \in Q_p, c \in C \quad (4.38)$$

$$y_c \in \{0,1\} \qquad \forall c \in C \quad (4.39)$$

$$z_{mc} \ge 0 \text{ and integer} \qquad \forall m \in M, c \in C \quad (4.40)$$

The objective function in equation (4.34) minimizes the non-linear total costs function of: *i*) machine/tool acquisition (i.e. depreciation) costs and *ii*) material handling costs. Assuming that cell sizes are not large and that within-cell distances are small, intercellular transportation costs are considered as surrogate of material handling costs. Constraint (4.35) establishes a feasibility condition, ensuring that each disassembly task $\{p, q\}$ is assigned to a single cell. Constraint (4.36) provides bounds conditions and ensures that the total number of disassembly task assigned to a cell does not exceed predefined levels. The capacity conditions are given by constraint (4.37) which ensures that the capacity of the machine type m is not exceeded, while ensuring that all recovered products are disassembled. Finally the constraints (4.38) to (4.40) deal with the nature of the decision variables.

The above is a quadratic integer program with linear constraints. The objective function is a non-linear equation due to the quadratic terms of intercellular movement. In Adenso-Díaz *et al.* (2006) the objective function was linearized using the following transformation:

$$x_{pjc} \cdot (1 - x_{pj'c}) = \frac{1}{2}(v_{pjj'c} + u_{pjj'c}) \tag{4.41}$$

under the following set of constraints:

$$x_{pjc} - x_{pj'c} = v_{pjj'c} - u_{pjj'c} \qquad \forall p \in P,\ j \in Q_p,\ c \in C,\ j' > j \tag{4.42}$$

where $v_{pjj'c}$ and $u_{pjj'c}$ (with $j' > j$) are two nonnegative continuous variables. Finally the objective function is given by:

$$Minimize\ TC' = \sum_{m \in M} \sum_{c \in C} \beta_m \cdot z_{mc} + \sum_{p \in P} \sum_{j \in Q_p} \sum_{j < j'} \sum_{c \in C} \alpha_p \cdot D_p \cdot f_{pj} \cdot \pi_{pjj'} \cdot \frac{1}{2} \cdot (v_{pjj'c} + u_{pjj'c}) \tag{4.43}$$

4.3.2 The Reconfigurable Disassembly Cell Formation Problem

In a multi-period context, the variability can be observed in the change of demand from period to period. Allowing system reconfiguration enhances the flexibility of disassembly systems to respond to this dynamic condition of demand. In fact, by rearranging the cells, the disassembly system can continue operating efficiently as the product mix and demand change. Reconfiguration consists of swapping the existing machines between cells (machine relocation); adding/removing machines to/from cells, machine duplication and changing the process plan of the product-operation. Also, the reconfiguration may consist of increasing or decreasing the number of formed cells in each period. The aim of the reconfigurable approach in our disassembly cell formation problem consist on designing a cell configuration within each period that establish a perfect costs balance among operating and reconfiguration costs between successive periods to adjust to long-term demand changes. We refer to Tavakkoli-Moghaddam *et al.* (2005) and Schaller (2007) for two examples of reconfigurable approaches in the manufacturing cell formation problem.

Formally speaking the formulation of the *Reconfigurable Disassembly Cell Formation Problem* (DCFP-D) requires the definition of a set of $T = \{t_1, t_2, \ldots, t_T\}$ periods for the planning horizon. The product demand varies from period to period in a deterministic manner over the entire planning horizon. The capability and capacity of each machine type are known and constant over time. The machine relocation is allowed and it is performed between periods and it requires zero time, also the machine relocation cost of each machine type is known and it is independent of where the machines are actually being relocated. Finally, the time value of money is not considered. The objective of the DCFP-D consist on group

disassembly tasks and assign them to cells together with their required resources so that total costs are minimized, while allowing system reconfiguration between successive periods. The mathematical formulation for the DCFP-D is formulated as follows:

DCFP-D (Reconfigurable approach for the Disassembly Cell Formation Problem)

$$\text{Minimize } TC_D = \sum_{t\in T}\sum_{m\in M}\sum_{c\in C}\beta_m \cdot z_{mct} + \sum_{t\in T}\sum_{m\in M}\sum_{c\in C}\gamma_m \cdot \left(z^{+}_{mct} + z^{-}_{mct}\right) + \sum_{t\in T}\sum_{p\in P}\sum_{j\in Q_p, j<j'}\sum_{c\in C}\alpha_p \cdot D^t_p \cdot f_{pj} \cdot \pi_{pjj'} \cdot \frac{1}{2} \cdot \left(v^t_{pjj'c} + u^t_{pjj'c}\right) \quad (4.44)$$

Subject to:

$$\sum_{c\in C} x^t_{pjc} = 1 \qquad \forall p \in P,\ j \in Q_p, t \in T \quad (4.45)$$

$$L \cdot y^t_c \le \sum_{p\in P}\sum_{j\in Q_p} x^t_{pjc} \le U \cdot y^t_c \qquad \forall c \in C, t \in T \quad (4.46)$$

$$\sum_{p\in P}\sum_{\{j:m\in O_{pj}\}} D^t_p \cdot \theta_{pj} \cdot f_{pj} \cdot x^t_{pjc} \le H_m \cdot z_{mct} \qquad \forall m \in M, c \in C, t \in T \quad (4.47)$$

$$z_{mc(t-1)} + z^{+}_{mct} - z^{-}_{mct} = z_{mct} \qquad \forall m \in M, c \in C, t \in T \quad (4.48)$$

$$x^t_{pjc} - x^t_{pj'c} = v^t_{pjj'c} - u^t_{pjj'c} \qquad \forall p \in \mathrm{P}, j \in Q_p, c \in C,\ j' > j, t \in \mathrm{T} \quad (4.49)$$

$$x^t_{pjc} \in \{0,1\} \qquad \forall p \in P,\ j \in Q_p, c \in C, t \in T \quad (4.50)$$

$$y^t_c \in \{0,1\} \qquad \forall c \in C, t \in T \quad (4.51)$$

$$z_{mct}, z^{+}_{mct}, z^{-}_{mct} \ge 0 \text{ and integer} \qquad \forall m \in M, c \in C, t \in T \quad (4.52)$$

$$v^t_{pjj'c}, u^t_{pjj'c} \ge 0 \qquad \forall p \in \mathrm{P}, j \in Q_p, c \in C,\ j' > j, t \in \mathrm{T} \quad (4.53)$$

The objective function in equation (4.44) has been linearized according to equations (4.41 – 4.42) to remove the nonlinearity caused by the quadratic terms of intercellular movement. It minimizes the total sum of the machine amortization cost, the machine relocation cost and the intercellular material handling cost over the planning horizon. The first term represents the cost of all required machines. The machine amortization or investment cost is obtained by the product of the number of machine type m in cell c in period t and the respective costs. The second term is the machine relocation cost. It is the sum of the product of the number of machines relocated and the respective costs. The number of machines relocated is obtained by the sum of the number of machines to be added plus the number of machines to be removed in all cells in all periods. The third term is the intercellular material handling costs. Constraint (4.45) sets a feasibility condition. It ensures that in period t, the j-th operation of product p, will be assigned to one and only one cell. Constraint (4.46) provides the lower and

upper bounds conditions of cells. It ensures that the total number of tasks assigned to one cell does not exceed the predefined bounds. Capacity conditions are given by constraint (4.47). It guarantees that machine capacities are not exceeded, while ensuring that all recovered products are dismantled during each period. The parameter θ_{pj} implies that during all the time required by the j-th operation of product p, the machine/tool type m cannot be used in any other operation; all that time corresponds to the machine type m, not matter whether the use of other machines were also considered in the total duration time of the operation. Constraint (4.48) determines how many machines of each type are relocated into or out of each cell during each period. It states that the number of machine type m assigned to cell c in the current period (t) equals the number of machine type m assigned to cell c in the previous period ($t - 1$) plus the number of machine type m added to cell c at the beginning of period t, minus the number of machine type m removed from cell c at the beginning of period t. It should be noted that the number of machine type m assigned to cell c in period zero are zero, i.e. at the beginning of the planning period no machine has been installed in any cell $z_{mc0} = 0$, $\forall m \in M$, $c \in C$. Constraint (4.49) provides the conditions under which the objective function is linearized, where $v^t_{pjj'c}$ and $u^t_{pjj'c}$ (with $j' > j$) are two nonnegative continuous variables. Finally, constraints form (4.50) to (4.53) deals with the nature of the decision variables.

4.3.3 The Robust Disassembly Cell Formation Problem

The aim of robust optimization approach in our disassembly cell formation problem is to design a cell configuration that performs well across several potential realizations of uncertainty over a specified planning horizon. Since the definition of good performance is context-dependent, there exist several definitions of robustness. Particularly relevant to our research are those provided by Kouvelis and Yu (1997) and Mulvey *et al.* (1995). The first describes the goal of robust optimization as finding a solution whose objective function remains close to optimal solution for each scenario, while the latter defines it as finding a solution that minimizes the largest deviation from optimality.

4.3.3.1 Robust Probabilistic Formulation of the DCFP

To formulate a robust DCFP with demand uncertainty under the concept of robust optimization described by Mulvey *et al.* (1995) we assume that the decision variables related to the formation of cells (y_c) and the decision variables related to the number of machines allotted to cells (z_{mc}) are the *design variables* valid for all scenarios, and we let the decision to allocate the j-th operation of product p to cell c form the *control variables*, which can be adjusted once the demand is observed. Thus, the cells are formed and the machines are assigned to them, and then decisions regarding disassembly task and flow patterns are made for each scenario realization. Consequently, we will have a variable number of tasks processed in each cell c, in relation to each scenario $s \in S$, which is expressed in the decision variable $x^s_{pjc} \in \{0,1\}$. We introduce the error or *recourse variables* el^s_c and eu^s_c to indicate any shortages or excesses in the size of cells respectively, and em^s_c to indicate any capacity surplus in the machines. Note that although these variables are declared as continue, the formulation forces them to take integer values. We also introduce five extra parameters: parameters λ_1, λ_2 and λ_3 that the decision maker can adjust to give varying importance to each component of the function that penalizes infeasibility in the objective function; and parameters σ_1 and σ_2 representing the percentage of the maximum permissible deviation for each error variable el^s_c and eu^s_c, respectively.

To measure the *solution-robustness* we make use of a function commonly used in stochastic lineal programming formulations (expected value) $f(\cdot) = \sum_{s \in S} \rho_s \cdot \xi_s + \sum_{m \in M} \sum_{c \in C} \beta_m \cdot z_{mc}$, where ξ_s is the objective function (previously linearized) to be minimized under scenario s, i.e.:

$$\xi_s = \sum_{p \in P} \sum_{j \in Q_p} \sum_{j < j'} \sum_{c \in C} \alpha_p \cdot D^s_p \cdot f_{pj} \cdot \pi_{pjj'} \cdot \frac{1}{2} \cdot \left(v^s_{pjj'c} + u^s_{pjj'c} \right) \tag{4.54}$$

On the other hand, to measure the *model-robustness* we make use of the following function to minimize the expected value of the associated error $\lambda \cdot h(\cdot) = \sum_{s \in S} \rho_s \cdot \sum_{c \in C} (\lambda_1 \cdot el^s_c + \lambda_2 \cdot eu^s_c + \lambda_3 \cdot em^s_c)$. Thus the robust disassembly cell formation problem is formulated as follows:

DCFP-R (Robust Model for the Disassembly Cell Formation Problem)

$$\text{Minimize } TC_R = \sum_{m\in M}\sum_{c\in C}\beta_m \cdot z_{mc} + \sum_{s\in S}\rho_s \cdot \xi_s + \sum_{s\in S}\rho_s \cdot \sum_{c\in C}\left(\lambda_1 \cdot el_c^s + \lambda_2 \cdot eu_c^s + \lambda_3 \cdot em_c^s\right) \quad (4.55)$$

Subject to:

$$\sum_{c\in C} x_{pjk}^s = 1 \qquad \forall p\in P, j\in Q_P, s\in S \quad (4.56)$$

$$\sum_{p\in P}\sum_{j\in Q_P} x_{pjc}^s + el_c^s \geq L\cdot y_c \qquad \forall c\in C, s\in S \quad (4.57)$$

$$\sum_{p\in P}\sum_{j\in Q_P} x_{pjc}^s - eu_c^s \leq U\cdot y_c \qquad \forall c\in C, s\in S \quad (4.58)$$

$$\sum_{p\in P}\sum_{\{j:m\in O_{pj}\}} D_p^s \cdot \theta_{pj}\cdot f_{pj}\cdot x_{pjc}^s - em_c^s \leq H_m \cdot z_{mc} \qquad \forall m\in M, c\in C \quad (4.59)$$

$$x_{pjc}^s - x_{pj'c}^s = v_{pjj'c}^s - u_{pjj'c}^s \qquad \forall p\in \mathrm{P}, j\in Q_p, c\in C,\ j'>j, s\in \mathrm{S} \quad (4.60)$$

$$0\leq el_c^s \leq \lceil \sigma_1 \cdot L\rceil \qquad \forall c\in C, s\in S \quad (4.61)$$

$$0\leq eu_c^s \leq \lceil \sigma_2 \cdot U\rceil \qquad \forall c\in C, s\in S \quad (4.62)$$

$$x_{pjc}^s \in \{0,1\} \qquad \forall p\in P, j\in Q_p, c\in C, s\in S \quad (4.63)$$

$$y_c \in \{0,1\} \qquad \forall c\in C \quad (4.64)$$

$$z_{mc}\geq 0 \text{ and integer} \qquad \forall m\in M, c\in C \quad (4.65)$$

$$v_{pjj'c}^s, u_{pjj'c}^s \geq 0 \qquad \forall p\in \mathrm{P}, j\in Q_p, c\in C,\ j'>j, s\in \mathrm{S} \quad (4.66)$$

$$em_c^s \geq 0 \qquad \forall c\in C, s\in S \quad (4.67)$$

Constraint (4.56) sets that for each scenario s, the j-th operation of product p must be assigned to one and only one cell. Constraints (4.57) and (4.58) are control constraints, the error variables el^s_c and eu^s_c determine whether to a cell c has been allotted less than L tasks or more than U task respectively under a given scenario s. For example in (4.57), if for a given scenario s, the total number of disassembly task assigned to cell c is greater than L, then the lower bound is met and the deviation will be $el^s_c = 0$, however, if less than L disassembly task are assigned then $el_c^s = L - \sum_{p\in P}\sum_{j\in Q_p} x_{pjc}^s$ when the objective function is minimized, which is similar in (4.58). Constraint (4.59) is also a control constraint, where the error variable em^s_c measures the lack of capability in cell c. Constraint (4.60) is auxiliary to keep the model linearized. Constraints (4.61) and (4.62) define the maximum permissible deviation for error variables el^s_c and eu^s_c respectively. Finally constraints from (4.63) to (4.67) define the nature of the decision variables.

4.3.3.2 Minimax Regret Formulation of the DCFP

To obtain a robust formulation for the disassembly cell formation problem under the minimax regret criterion, we will consider to obtain a relative robust decision, which exhibits *the best worst-case percentage deviation from optimality* among all feasible decisions on all scenarios, i.e.:

$$z_R = \max_{s\in S} \frac{f(X_R, D^s) - f(X_s^*, D^s)}{f(X_s^*, D^s)} = \min_{X\in\bigcap_{s\in S} F_s} \max_{s\in S} \frac{f(X, D^s) - f(X_s^*, D^s)}{f(X_s^*, D^s)} \tag{4.68}$$

With regard to the framework of robust optimization discussed in this section, the relative robust model for the design of disassembly cells with demand uncertain is formulated as follows:

DCFP-REG (Minimax Regret Model for the Disassembly Cell Formation Problem)

Minimize δ (4.69)

Subject to:

$$\sum_{m\in\mathbf{M}}\sum_{c\in\mathbf{C}} \beta_m \cdot z_{mc} + \sum_{p\in\mathbf{P}}\sum_{j\in Q_p}\sum_{j<j'}\sum_{c\in\mathbf{C}} \alpha_p \cdot D_p^s \cdot f_{pj} \cdot \pi_{pjj'} \cdot \frac{\left(v_{pjj'c}^s + u_{pjj'c}^s\right)}{2} \le (\delta+1)\cdot O_s^* \qquad \forall s\in S \tag{4.70}$$

$$\sum_{c\in\mathbf{C}} x_{pjc}^s = 1 \qquad \forall p\in P, j\in Q_p, s\in S \tag{4.71}$$

$$L\cdot y_c \le \sum_{p\in\mathbf{P}}\sum_{j\in Q_p} x_{pjc}^s \le U\cdot y_c \qquad \forall c\in C, s\in S \tag{4.72}$$

$$\sum_{p\in\mathbf{P}}\sum_{\{j:m\in\mathbf{O}_{pj}\}} D_p^s \cdot \theta_{pj} \cdot f_{pj} \cdot x_{pjc}^s \le H_m \cdot z_{mc} \qquad \forall m\in M, c\in C, s\in S \tag{4.73}$$

$$x_{pjc}^s - x_{pj'c}^s = v_{pjj'c}^s - u_{pjj'c}^s \qquad \forall p\in P, j\in Q_p, c\in C,\ j'>j, s\in S \tag{4.74}$$

$$x_{pjc}^s \in \{0,1\} \qquad \forall p\in P, j\in Q_p, c\in C, s\in S \tag{4.75}$$

$$y_k \in \{0,1\} \qquad \forall c\in C \tag{4.76}$$

$$z_{mc} \ge 0 \text{ and integer} \qquad \forall m\in M, c\in C \tag{4.77}$$

$$v_{pjj'c}^s, u_{pjj'c}^s \ge 0 \qquad \forall p\in P, j\in Q_p, c\in C,\ j'>j, s\in S \tag{4.78}$$

The objective function (4.69) is the robustness measure for a given set of decision variables. The variable δ is the robust objective; it represents the maximum percentage deviation from optimality among all scenarios, which must be minimized. The maximum relative deviation across all scenarios is computed in constraint (4.70), and it measures the

relative difference between the value of the optimal solution for the DCFP in a particular scenario and the value of the objective function evaluated for the robust solution in such scenario. The remaining constraints are similar to those used for the robust model formulated under the framework of Mulvey *et al.* (1995), the same binary decision variables y_c applies to all scenarios, and also the integer decision variables z_{mc}; both determine respectively the cells to form and the allocation of machines to those cells. While the binary decision variable x^s_{pjc} represents the variable number of tasks to be processed in each cell c, associated to each scenario s.

4.3.4 Problems Complexity and Remarks

It has been proved that the cell formation problem is a variation of the NP-hard fixed charge problem with additional decision variables for cell formation (Cao and Chen, 2004), and therefore, it is essentially an NP-hard problem. Thus it is usually inefficient to use a brute force algorithm to solve it for real-life problems. Computational difficulties are mainly caused by the machines capacity constraints (essentially a knapsack type constraint).

It should be noted that both the DCFP-D and the DCFP-R are more complicated problems than the classical cell formation problem with additional decision variables associated to the use of multiple periods and scenarios, and it follows that both are also NP-hard problems difficult to solve even for relatively small instances. Also note that in solving the above problems, the number of machines assigned to the formed cells is relatively small. Therefore, round-off solutions of the corresponding relaxed linear programming models cannot be used as the approximate solutions to the originals problems. Actually, to solve practical problems, feasible integer solutions are obtained after a reasonable computational time in a commercial MIP solver.

In the subsequent sections within this thesis, a metaheuristic algorithm to solve the disassembly cell formation problem with demand variability is proposed. Because of regret models require a different approach to deal with robustness, which implies knowing a priori the optimal solution for any possible realization of the scenarios, the designed heuristic procedure shall deal just with the reconfigurable model and with the robust probabilistic model developed under the ROMF proposed by Mulvey *et al.* (1995).

Chapter 5

Solution Approaches

The aim of this chapter is to describe the solution approaches used in each corresponding problem. In Section 5.1 we propose a methodological approach, for designing a WEEE collection network, which combines integer programming, heuristic algorithms and simulation. In Section 5.2 we propose a GRASP algorithm for solving the vehicle routing problem with split loads and date windows. Finally in Section 5.3 two metaheuristic algorithms are designed: a Tabu Search algorithm for the basic disassembly cell formation problem and a Variable Neighborhood Search algorithm for the robust and reconfigurable disassembly cell formation problems.

5.1 Methodological Approach to Design WEEE Collection Networks

The problem sketched in Section 3.1, requires a redesign of the WEEE collection network. To solve this problem two interrelated sub-problems should be addressed: the facility location problem and the vehicle routing problem. Therefore, in this section, a three-phase hierarchical approach based on the logic of the *Preview-Solve-Review* approach is proposed. An overview of the solution approach is depicted on figure 5.1. Based on a geographically dispersed group of customers that need to be served, and a set of potential locations of depots for WEEE collection, the problem is solved as follows: (1) solve the facility location problem as an optimization problem, using a commercial general purpose optimizer; (2) for every depot solve the collection routing problem in a weekly basis in order to determine the set of routes for each period, using the customized algorithm described just below; (3) analyze the

routes performance through a simulation study in order to identify the most appropriate value for n_k (number of vehicles type k allotted to the depot) and the inventory policy in the depot.

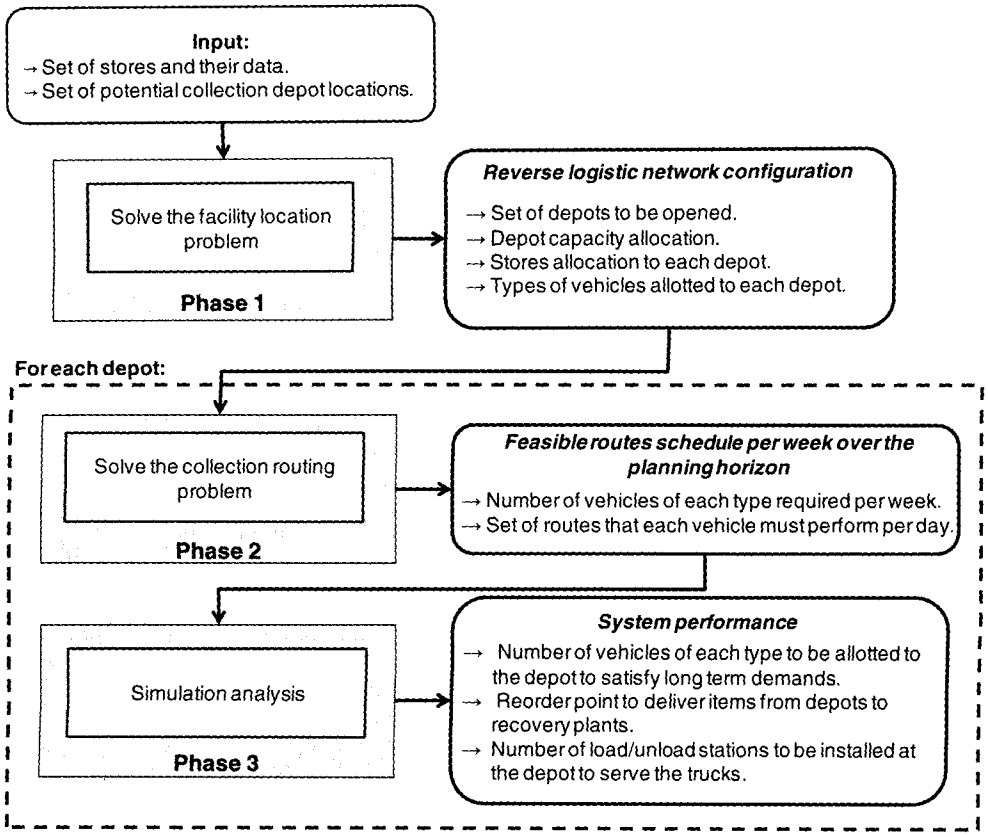


Figure 5.1 Methodological solution approach to design the recovery network for WEEE collection in Galicia.

5.1.1 Heuristic Algorithm for Solving the WEEE Collection Routing Problem

Since it has been proved that the *vehicle routing problem* is NP-hard, it is common to find implementations of heuristic algorithms to solve it. In general, heuristic algorithms for routing problems consist of two phases: a tour construction procedure to generate an initial route, and an improvement procedure, which tries to find a better solution given an initial tour. Probably the best known, more discussed and analyzed heuristic procedure in literature is the constructive savings algorithm C&W (Clarke and Wright, 1964). Since its publication,

several variants and improvements have been proposed (see e.g. Mandl, 1979), some of them aimed at improving its computational efficiency through a use of better data structures, and other aimed at improving their effectiveness in terms of the quality of their solutions (Cordeau *et al.*, 2007). The following approach for designing harvesting routes is tailored to the task of WEEE collection in Galicia, where the three-stage heuristic procedure is capable of managing a heterogeneous fleet of capacitated vehicles (Fig. 5.2).

```
READ problem data: T, B_ik, R_it, q_k, λ_k, d_ij, ω_i.

for each period t ∈ T in the planning horizon do

   RoutesGenerator Procedure( )
      starting(). Assign to each store a single route, to
      be served by the least cost vehicle able to serve
      the store.
      do{
         saving_list( ). Compute the saving matrix S_ij and
         sort it in a descending order in a list
         SL[i][j][S_ij].
         Set unions ← 0 and s_h,k ← First pair of routes in
         list SL.
         while SL list ≠ ∅ do
            if merge_feasibility(h,k) == True then
               merge(Route_h, Route_k)
               update_info(R)
               unions ++
            end if
            s_h,k ← next s_i,j ∈ SL not considered yet
         end while
      } while (unions > 0)

   CombineRoutes Procedure ( )
      for each vehicle type k ∈ K do
         if at least one route is serve by vehicle type k
         then
            1.Compute the number of blocks that a single
              truck k can execute at day DTR^k without
              exceeding its limits.
            2.Compute the number of vehicles type k needed
              in period t as: n_k^t = 1 + ⌊DTR^k/num_days_week⌋.
            3.Uniformly divide the blocks of routes between
              the working days in the system BR_k^t = ⌈DTR^k/n_k^t⌉,
              in a way that every vehicle make use of
              similar usage rate.
         end if
      end for

end for

return "a set of m routes for each period with an
appropriated number of vehicles"
```

Figure 5.2 Pseudo code of the heuristic algorithm proposed for solving the WEEE collection routing problem with a heterogeneous fleet of capacitated vehicles.

The purpose of the `RoutesGenerator` procedure is to construct a set of feasible routes, analyzing when required, the feasibility of using a different vehicle type for merging two different routes. A route is feasible if the capacity of the vehicle along the route is not exceeded, and the change in the use of a different vehicle type is feasible if when combining two routes: the new vehicle can access to all stores in both routes and using such new vehicle is less expensive than using both routes in their own. We make use of a recursive function implementation for the systematic generation of routes. The general procedure to generate routes is detailed below.

In the *starting* phase, using the logic of the C&W algorithm it is assumed that each store i allotted to the depot under consideration is served by an independent vehicle k of least cost that is able to respond, considering: (1) the amount of WEEE that such store has available for delivery every time it is visited, (2) the distance travelled between the depot and the store, and (3) the accessibility of the vehicle to the store. In the *saving_list* phase, the procedure determines a list of pairs of existing routes i and j to merge. The pairs are formed for all the combinations without repetition of nodes that are directly linked to the depot, without considering that pair of nodes that are in the same route. The list is sorted in decreasing order based on the resulting savings from the union (s_{ij}) from the point of view of the distance. The procedure continues combining routes according to the sorted list, while the list has not been exhausted and without limiting to strictly positive savings ($s_{ij} > 0$), as greater savings could result in changing the type of vehicle. The feasibility of their merge is tested according to the following conditions in the *merge_feasibility* phase:

i. If both routes (i, j) are served by the same type of vehicle k, then:
    1. Check if their union does not exceed both the capacity of the vehicle (q_k) used, as well as the bound set as the maximum distance (L_k) to travel by the vehicle type k; if both constraints are not exceeded the route i will join the route j, the remaining capacity of the vehicle and the length of the route are updated.
    2. Otherwise, we analyze whether a change in the use of a vehicle k' with larger capacity (load and travel), is economically more efficient to meet the union of the two routes. To this end, for each vehicle $h \in \mathbf{K} - \{k\} | q_h > q_k$ it will be assessed if: (a) the vehicle type h can access all the stores contained in the two routes, (b) the vehicle type h has sufficient load and travel capacity to cover both routes, (c) the use of the vehicle type h is cheaper than making the two existing routes separately, considering the savings by getting rid of the smaller vehicle and the costs incurred

when using a larger vehicle capacity across the route at the rate of fixed and variable costs per kilometer; the savings in the use of vehicle h is given by $A_h = \left(2\pi'_k + \lambda_k \times \left(d_{Route\_i} + d_{Route\_j}\right)\right) - \left(\pi'_h + \lambda_h \times d_{New\_Route}\right)$, where π'_k represents the fixed cost per kilometer traveled by the vehicle type k. If the three assumptions are positive the algorithm changes from vehicle k to vehicle k', combining two routes into one that will be addressed by the new vehicle type k', which updates its remaining capacity and the length of the route. Note that $k' = \text{argmax}\{A_h | A_h > 0\}$ and h satisfy conditional triplet from (a) through (c).

ii. If the route i requires a different vehicle than the route j, the formation of a new route will be analyzed, considering the vehicle with the larger capacity (load and travel) that has been assigned to any of the two routes, providing that such vehicle can visit every store in both routes. Repeat steps 1 and 2 above.

If the union of at least a couple of routes was feasible, the algorithm updates the saving list and repeats the procedure, otherwise, proceed with the combination of routes.

Finally, the `CombineRoutes` procedure determines the set of routes that one vehicle can serve in a given period, trying to optimize the number of trucks required to serve the routes. The result of this procedure is a set of routes per period, which will be input to the simulator. Below, an analysis of the performance of the procedure when compared with optimal solutions for small instances is provided.

5.1.2 Routes Simulation

Based on the routes previously generated by the heuristic algorithm for the T-week planning period, the routes behavior will be simulated varying the number of vehicles of each type to allot to the depot, using collection orders generated following empirical distributions. The interval between two successive collection orders includes the time required by the system to authorize the vehicle to leave the depot to undertake the tour. Every time an order is generated for a collection route of a vehicle type k, the simulator selects the next available truck within the fleet of such vehicles type to undertake the tour. Concluding the tour, the truck must be unloaded. Assuming limited resources in the unloading process, a truck that had just concluded a tour must wait in a queue to be served. Regarding to the shipping orders from depots to disassembly plants, these are made under a continuous review inventory

policy. Every time the depot reaches a $\xi\%$ of its maximum capacity a shipping order is generated to be completed during the planning period. The simulation analysis will give us insights on the evolution of the daily load in the depot for different values of the ξ parameter, as well as the vehicle usage rate and the average waiting time of trucks in the downloading process, as a function of the final number of vehicles of each type allotted to the depot.

5.2 GRASP Algorithm for the VRP-SLDW

In this section we introduce a Greedy Randomized Adaptive Searching Procedure (GRASP) based heuristic to solve the Vehicle Routing Problem with Split Loads and Date Windows (VRP-SLDW) described in Section 3.2 and modeled in Section 4.2. GRASP was introduced in Feo and Resende (1989) and later formalized in Feo and Resende (1995). The motivation for using GRASP in this particular application is based on the fact that there is a natural form to define a priority function to select the customer to be inserted in the initial solution, by linking the customer demand and the available time for its collection.

5.2.1 General Outline of the Metaheuristic Procedure

When designing an algorithm it is important to define a good structure for encoding the solution. Computationally speaking, in our particular case the simplest structure to consider for representing the solution is a LOAD matrix (see example in Fig. 5.3) which has as many rows as the total number of routes ($K \times R \times T$) and as many columns as the number of customers to serve (m). Every route π is defined by the set of three attributes: the day t, the vehicle k and the tour r (π^t_{kr}). Each cell (π, i) specifies the amount of demand $l_{\pi i}$ collected on route π from customer i. Since every row in LOAD corresponds to a route and every column corresponds to a customer, a feasible solution must satisfy that: a) for each row its sum must not exceed the capacity of the vehicle k used in route π, b) the cumulative sum for the traveling time in a subset of routes using a vehicle k in a given day t must not exceed L_k, and c) for each column its sum must be at most equal to the demand of each customer ω_i.

Although the mathematical model requires the collection to be done between s_i and f_i, the solution algorithm must relax this feature allowing the collection after f_i, but incurring in a penalty cost for each day after the collection date promised. Therefore, the use of an

appropriate data structure is required to identify when a route is *feasible* (i.e. the vehicle allotted to the route has enough time and load capacity to serve the customer), *unrestricted* (i.e. the vehicle allotted to the route may access to the customer's location) and *non-penalized* (i.e. the day allotted to the route belongs to the customer's date window) for each customer i. A feasible solution s has associated a set of feasible routes $\Pi = \{\pi_1, \ldots, \pi_v\}$. The simplest structure considered to represent the set of routes corresponds to another matrix ROUTE (see Fig. 5.4) with the same dimensions than the LOAD matrix. Each row in the ROUTE matrix represents a route π, each column a customer i and each cell (π,i) (with $l_{\pi i} \geq 1$) indicates the order in which the customer i is visited into route π. Given a new assignment in the LOAD matrix, the cost of the modified route is evaluated by solving the corresponding TSP (*Traveling Salesman Problem*) in ROUTE. Each route $\pi \in \Pi$ has an associated travel time h_π related to both the number of customers visited in the route and the order in which they are visited. Since the traveling time sum across the set of routes that uses the same vehicle k on the same day t should not exceed L_k (i.e. $\sum_{\pi \in \Pi | \pi = \pi_k} h_\pi \leq L_k$) both matrix LOAD and ROUTE are interdependent.

5.2.2 Greedy Randomized Construction

The construction mechanism at each iteration aims to allocate the largest amount of unsatisfied demand from customer $i \in V \backslash \{0\}$ to a single route $\pi \in \Pi$ such that π is feasible, non-penalized and unrestricted for i. A route π is feasible for customer i if and only if the vehicle allotted to the route can be used to serve such customer; on their own a route π is non-penalized for customer i if the day on which the route has been allotted is within the date window $[s_i, f_i]$ for customer i; and finally a route π is unrestricted for customer i if both the remaining capacity of the route $\bar{Q}_k$ and the maximum operation time L_k of all routes using the same vehicle k on the same day t are not exceeded when the customer i is inserted on route π.

The basic principle for the construction phase aims to fulfill the routes as much as possible and split the demand as few as possible. The constructive procedure starts with an empty solution, and iteratively the customers demand is distributed among the routes according to the greedy randomized construction scheme. When the total demand of each customer has been distributed among the routes or the pending demand to assign of each customer is smaller than a value ε, the constructive phase finishes and the local search procedure is

initiated. To select the next customer i to be included in the solution, we make use of a greedy function $\phi(i)$ which relates the pending demand ϖ_i to satisfy from each customer i and the time available to perform the service. The value $\phi(i)$ measures the critical ratio for customer i, so the higher it is the most urgent its service is. This ratio is computed as:

$$\phi(i) = \frac{\varpi_i}{slack_i} \tag{5.1}$$

where $slack_i = f_i - s_i$ represents the available time to carry out the collection from customer i. If $f_i = s_i$, then $\phi(i) = INF$.

		Customers											
	Route	1	2	3	4	5	6	7	8	9	10	$\bar{Q}_k$	Q_k
A	π^1_{11}				9		16					0	25
B	π^1_{31}						40					10	50
C	π^1_{32}			50								0	50
D	π^1_{33}					50						0	50
E	π^2_{11}	8	14									3	25
F	π^2_{12}			8	17							0	25
G	π^2_{31}										50	0	50
H	π^2_{32}			50								0	50
I	π^3_{11}			25								0	25
J	π^3_{21}							29				1	30
K	π^3_{31}					50						0	50
L	π^3_{32}						50					0	50
M	π^4_{11}								23			2	25
N	π^4_{12}								25			0	25
O	π^5_{11}								25			0	25
P	π^5_{31}					46						4	50
Q	π^6_{11}									10	11	4	25
	σ_i	0	0	1	0	0	0	0	0	0	0		
	ω_i	8	14	134	26	146	106	29	73	10	61		

Figure 5.3 LOAD matrix: solution to the trial example. Column $\bar{Q}_k$ states for the remaining capacity of the vehicles used in its corresponding route and σ_i the amount of items not collected from customer i. Gray cells indicate that the route π is penalized for a given customer i; i.e. those days are out of the specific date window for the customer. Only routes with load assigned are shown.

	Route	Customers 1	2	3	4	5	6	7	8	9	10
A	π^1_{11}				1		2				
B	π^1_{31}						1				
C	π^1_{32}			1							
D	π^1_{33}					1					
E	π^2_{11}	2	1								
F	π^2_{12}			2	1						
G	π^2_{31}										1
H	π^2_{32}			1							
I	π^3_{11}			1							
J	π^3_{21}							1			
K	π^3_{31}					1					
L	π^3_{32}						1				
M	π^4_{11}								1		
N	π^4_{12}								1		
O	π^5_{11}								1		
P	π^5_{31}					1					
Q	π^6_{11}									1	2

Figure 5.4 ROUTE matrix for the trial example. Each cell (π, i) indicates the order in which the customer i is visited into a route π. This figure only shows those routes (rows) used in a particular solution.

The detailed procedure is depicted in Fig. 5.5. If there is at least one customer i with $\phi(i) = INF$, then the Restricted Candidate List (RCL) is exclusively formed by such customers; otherwise, a customer i with $\phi(i) > 0$ belongs to the RCL if: $\phi(i) \geq \Phi_{max} - \alpha \cdot (\Phi_{max} - \Phi_{min})$, where Φ_{min} and Φ_{max} are respectively the lowest and highest value obtained by the greedy function, and α ($0 \leq \alpha \leq 1$) is a parameter that controls the degree of randomness allowed. The selection of a customer i within the RCL is done randomly. Subsequently a call to the `AssignLoad` procedure is performed, whose main function consist on finding a feasible route π for the selected customer i, to whom the largest amount of pending load (ϖ_i) for such customer can be assigned; to this end we define $\Pi^i \subseteq \Pi$ as the subset of feasible, unrestricted and non-penalized routes for customer i, and the following scenarios are considered: a) if $|\Pi^i| = 0$, then the date window for customer i is artificially augmented by one day, without affecting $slack_i$; b) if $|\Pi^i| = 1$, then $x_{\pi i} = \min\{\bar{Q}_\pi, \varpi_i\}$, where $x_{\pi i}$ represent the load of customer i to be assigned to route π; and c) if $|\Pi^i| > 1$, then customer i is assigned to route π

capable to carrying the largest load (in case of ties customer i is assigned to the route π where $H_\pi = \bar{Q}_\pi - x_{\pi i}$ is minimized). Finally, customer i is inserted into the route π, and both matrix ROUTE and LOAD are updated. The remaining load space $\bar{Q}_\pi$ in route π and the pending load to satisfy ϖ_i for customer i are also updated. When updated, if the pending load to satisfy for customer i is less than or equal to ε, then customer i is removed from the set of customers with pending load U. As pointed above, this is a common practice in any routing system, where a trip to collect a very small number of devices is never justified, however, these are considered in a following request.

```
procedure GreedyRandomizedConstruction( )
Input: α, LOAD, ROUTE.
Output: Set of feasible routes.

0     ϖ_i ← ω_i ∀i∈V\{0}; U ← V;
1     while U ≠ ∅ do
2        Compute φ(i) as in equation (5.1) for all i ∈ U;
3        Φ_min ← min_i{φ(i) | i ∈ U}; Φ_max ← max_i{φ(i) | i ∈ U};
4        if Φ_max < INF then RCL ← {i∈U | φ(i)≥ Φ_max - α·(Φ_max -Φ_min)};
5        else RCL ← {i∈U | φ(i) = INF};
6        Randomly select i from the RCL;
7        route ← null;
8        x_πi ← AssignLoad(i, route, DP, ϖ_i);
9        if x_πi > 0 then
10          if l_πi = 0 then
11             Update ROUTE matrix;
12             Update routing costs and traveling times;
13          end if;
14          Update LOAD matrix: l_πi ← l_πi + x_πi;
15          Q̄_π ← Q̄_π - x_πi;
16          ϖ_i ← ϖ_i - x_πi;
17          if ϖ_i ≤ ε then U ← U\{i}; endif;
18       end if;
19    end while;
20    Return LOAD, ROUTE, routes_cost, penal_cost.
end procedure GreedyRandomizedConstruction
```

Figure 5.5 Greedy Randomized Construction Procedure for the VRP-SLDW.

5.2.3 Local Search

The local search procedure is carried out by manipulating the allocation of the portion of demand $l_{\pi i}$ in the LOAD matrix and updating the routes involved in ROUTE. This procedure comprises two phases. Phase I verifies if penalties exist by serving the customers on a penalized route, and seeks to eliminate it. Phase II explores four neighborhoods of the

solution space in order to reduce the routing costs. All the movements described in the VRP-SL literature analyzed so far are included in our exploration. The movements are described below.

5.2.3.1 Phase I of the Local Search: penalties elimination

The movements considered in this phase of the local search are: m_{I1} the insertion move, and m_{E1} the exchange move. The neighborhoods created by these moves are completely explored under the best improving strategy. Given an initial solution, a greedy local search is implemented by choosing each time the best possible move among both neighborhoods. To detail both movements, the definition of the following sets is required: let P be the set of penalized customers i and C the set of customers $j \neq i$ that have an assignment in some of the routes of Π for which i is non-penalized and the insertion of i would be feasible and unrestricted.

Insertion Move 1(m_{I1}). This move comprises two steps. In the first step, for each customer $j \in C$ with $l_{\pi j} > 0$, the portion $x_{\pi j}^{\hat{\pi}}$ of load allotted to π that can be assigned to any other feasible, non-penalized and unrestricted route $\hat{\pi}$ for j is computed, such that $x_{\pi j}^{\hat{\pi}} = \min\{\bar{Q}_{\hat{\pi}}, l_{\pi j}\}$. Let $X_j^{\hat{\pi}} = max_{\hat{\pi}}\{x_{\pi j}^{\hat{\pi}}\}$ the maximum load for customer j that can be moved from π to $\hat{\pi}$', and $X^* = max\{X_j^{\hat{\pi}}\}$. The following options are considered: a) if $X^* = 0$, then it is not possible to reduce the load in penalization by means of the insertion move, and it is passed to the next move, b) if $X^* > 0$, then move the $x_{\pi j}^{\hat{\pi}}$ units from the route π of customer j that gave rise to X^*, to the route $\hat{\pi}$. In the second step we define $E_\pi = \bar{Q}_\pi + X^*$ as the space that would remain free in the route π after removing X^* units. Then for each customer $i \in P$ such that the route π is feasible, we compute $x_{ri}^{\pi} = min\{l_{ri}, E_\pi\}$ the maximum amount of penalized load for customer i that can be moved from its current penalized route r to the non penalized route π, and its corresponding pending load still penalized $CP_i = l_{ri} - x_{ri}^{\pi}$. The best move in this neighborhood is the one where x_{ri}^{π} is maximum and CP_i is minimum.

Exchange Move (m_{E1}). For each customer $i \in P$ visited on a route π, we consider the exchange of $x_{\pi i}^{\hat{\pi} j} = min\{l_{\pi i}, l_{\hat{\pi} j}\}$ units with all the routes $\hat{\pi} \in \Pi$ of each customer $j \in C$ such that π is a feasible and non-penalized route for j. With every move the corresponding CP_i is

computed: $CP_i = l_{\pi i} - x_{\pi i}^{\hat{\pi} j}$. The best move in this neighborhood is the one where $x_{\pi i}^{\hat{\pi} j}$ is maximum and CP_i is minimum.

5.2.3.2 Phase II of the Local Search: cost minimization

The movements considered in this second phase of the local search are: m_{I2} the insertion move, m_{E2} the interchange move, m_C the route combine move, and m_E the route elimination move. The neighborhoods created by these moves are also completely explored under the best improving strategy. Given an initial solution, a greedy local search is implemented by choosing each time the best possible move among all neighborhoods (see pseudo-code in Figure 5.6). Thus the order in which the neighborhoods are explored does not have any influence on the algorithmic performance.

```
procedure LocalSearchPhaseII( )
Input: LOAD, ROUTE.
Output: Improved cost solution.
0  do {
1    local_optima ← true;
2    x^insert ← Best move obtained from insertion move 2;
3    x^interchange ← Best move obtained from interchange move;
4    x^combine ← Best move obtained from combine move;
5    x^elimination ← Best move obtained from elimination move;
6    m* ← Best move among all neighborhoods;
7    if ∃ m* then
8       Update LOAD and ROUTE in according to m*.
9       Update routes_cost;
10      Local_optima ← false;
11   end if;
12 } while local_optima == false
13 Return improved LOAD, ROUTE.
end procedure LocalSearchPhaseII
```

Figure 5.6 Local search procedure for the VRSP-SLDW.

Insertion Move 2 (m_{I2}). This move is based on removing the largest portion of the load that a customer i has assigned on a route π, and inserting it into another feasible, non-penalized, and unrestricted route $\hat{\pi}$ for customer i. The portion of load to be moved is given by $x_{\pi i}^{\hat{\pi}} = min\{\bar{Q}_{\hat{\pi}}, l_{\pi i}\}$. If $x_{\pi i}^{\hat{\pi}} = l_{\pi i}$ implies that all the load of customer i on route π was removed and reassigned into another single route $\hat{\pi}$. See example in Fig. 5.7.

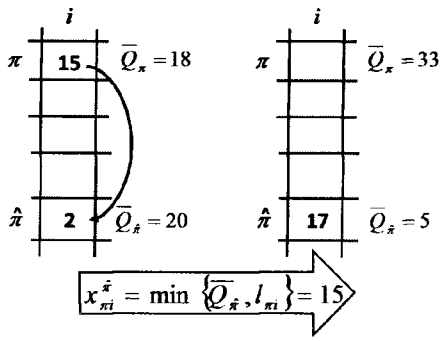


Figure 5.7 Insertion Move 2. This figure illustrates the m_{I2} move to insert 15 units from route π to route $\hat{\pi}$ for a given customer i.

Interchange Move (m_{E2}). This move is based on removing a customer from one route and inserting it into another route. For each pair of different customers $i \neq j$, such that i has an assignation $l_{\pi i}$ in $\pi \in \Pi$ and j an assignation $l_{\hat{\pi} j}$ in $\hat{\pi} \in \Pi$, the procedure explores the interchange of $x_{\pi i}^{\hat{\pi} j} = min\{l_{\pi i}, l_{\hat{\pi} j}\}$ units between both routes. See example in Fig. 5.8.

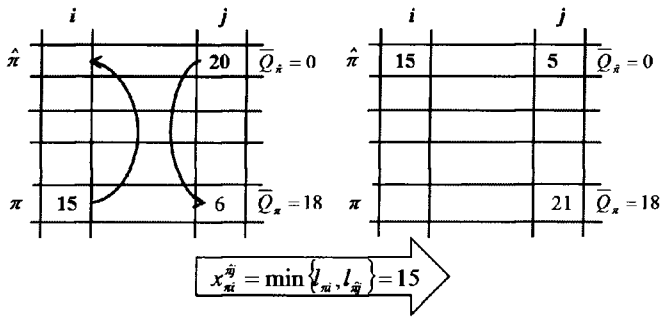


Figure 5.8 Interchange Move. This figure illustrates the m_{E2} move to interchange 15 units between the routes π and $\hat{\pi}$ for customers i and j.

Route Combine Move (m_C). This move aims to merge two different routes into a single one while the routing costs are minimized. For each pair of routes $\pi 1 \neq \pi 2$, the procedure evaluates in the first place the union of both routes in that with the largest vehicle. If the union is feasible in terms of the vehicle capacity and the accessibility and time constraints are satisfied, both routes are merged into a single one. Otherwise the procedure seeks for a larger vehicle in the fleet. See example in Fig. 5.9.

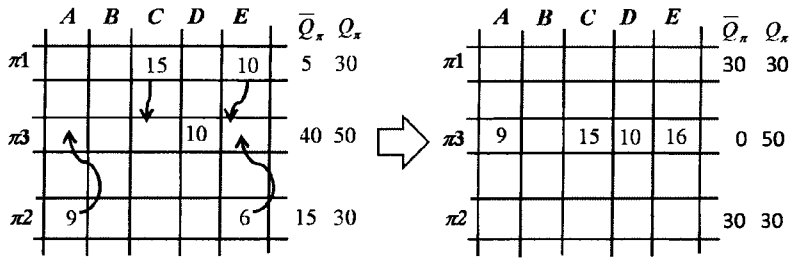


Figure 5.9 Combine Move. This figure illustrates the m_C move to combine routes π1 and π2 into route π3.

Route Elimination Move (m_E). This move aims to reduce the number of routes, by reallocating the entire load of a route among the others. For each route $\pi \in \Pi$, let C be the set of customers with load assigned in route π, and $\Pi^i \subseteq \Pi \backslash \{\pi\}$ the subset of feasible, unrestricted and non-penalized routes for customer $i \in C$. While $C \neq \emptyset$ or the reallocation of the entire load of π among the others routes were impossible, repeat: for each customer $i \in C$ the largest portion of load assigned to π that may be reassigned to any other route $\hat{\pi} \in \Pi^i$ is computed as $x_{\pi i}^{\hat{\pi}} = min\{\bar{Q}_{\hat{\pi}}, l_{\pi i}\}$. Let $i^* = argmax_i\{x_{\pi i}^{\hat{\pi}}\}$ the customer i who would reassign the largest amount of load from π to any other route $\hat{\pi} \in \Pi^i$. In case of tie, the customer i^* that when inserted into route $\hat{\pi}$ yields in the greatest routing cost saving is selected. Update i^*, $x_{\pi i}^{\hat{\pi}}$ and $\hat{\pi}$. If $x_{\pi i}^{\hat{\pi}} = l_{\pi i}$ then customer i^* is deleted from C. If the entire load of a route π was reassigned and the saving obtained by eliminating the route is greater than the resignations cost, then the movement is recorded as valid. See example in Fig. 5.10.

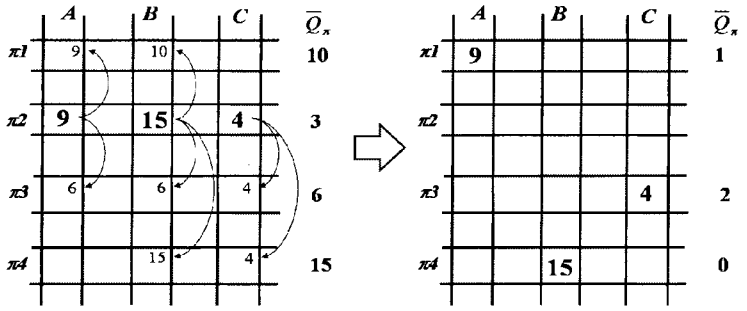


Figure 5.10 Elimination Move. This figure illustrates the m_E move, which eliminates route π2, by redistributing its load among routes π1, π3 and π4.

5.3 Metaheuristic Algorithms for the DCFP and its Variations

When a problem, such as the Disassembly Cell Formation Problem (DCFP) or any of its variations, is of such great complexity, the usual method for obtaining a solution consists in defining some kind of heuristic that will resolve it. In this section we describe not only the Tabu Search algorithm designed to analyze the performance of typical moves of the cell formation problem on a real-world instance of the DCFP, but we also describe a Variable Neighborhood Search algorithm that will be used in this thesis to generate quality solutions for both the robust and the reconfigurable disassembly cell formation problems.

Most of the techniques available on literature consider the operations commonality by defining a measure of similarity between parts and then grouping parts into families. The similarity measure helps ensure that cells are focused if parts produced within a cell have a high level of similarity. However, although new approaches have considered the processing sequence information by modifying the classic 0–1 part machine incidence matrix, a similarity measure approach generally does not consider part volumes, machine capacities and processing times. Part volumes are important because high volume parts may have a much larger impact than lower volume parts on material handling costs. Machine capacities are important for two reasons. First it may be that when processing loads are considered it may be required to have multiple machines of a given type and these machines can be allocated to smaller, more focused cells. Also, when considering processing loads it may happen that multiple machines of a type are required in a cell.

5.3.1 A Tabu Search Algorithm for the Disassembly Cell Formation Problem

In this section we introduce a Tabu Search (TS) heuristic to solve the disassembly cell formation problem depicted above. Tabu Search is an adaptive search procedure for solving optimization problems, designed to guide other methods to escape the trap of local optimality. TS has been used in a wide variety of classical and practical problems of high degree of complexity, giving rise to highly efficient algorithms. For a thorough description of TS, we refer to books (Glover and Laguna, 1996) and tutorials (Glover, 1990; Glover and Laguna, 1995) that cover this technique.

To implement a TS strategy the following questions must be answered: How is a solution defined? What is a move? What is the structure of the neighborhood of an existing solution? How many moves are retained in the tabu list? What are the criteria for stopping the procedure? What diversification strategy is employed? The remaining part of this section explains this procedure on the disassembly cell formation problem.

5.3.1.1 Solution Coding

In our first metaheuristic implementation for solving the disassembly cell formation problem a feasible solution x_{ic} consists of an assignment of each disassembly task ($i = \{p, j\}$: operation j of product p) to a cell c. In turn, x_{ic} can be view as a matrix that has as many rows as the total number of disassembly tasks (N) and as many columns as the number of cells to form (C); that satisfies:

$$\sum_{c \in \mathbf{C}} x_{ic} = 1 \qquad \forall i = 1,\ldots, N \qquad (5.2)$$

$$L \cdot y_c \leq W(c) \leq U \cdot y_c \qquad \forall c = 1,\ldots, C \qquad (5.3)$$

where, $W(c) = \sum_{i=1,\ldots,N} x_{ic}$ for each c, is a vector that counts the number of disassembly tasks assigned to cell c.

Once a solution x_{ic} has been defined the number of machines of each type required in each cell z_{mc} can be obtained from x_{ic} using the constraint (4.37) as follows:

$$\sum_{p \in P} \sum_{\{j: m \in O_{pj}\}} D_p \cdot \theta_{pj} \cdot f_{pj} \cdot x_{pjc} \leq H_m \cdot z_{mc} \qquad \forall m \in M, c \in C \qquad (4.37)$$

$$z_{mc} \geq \frac{1}{H_m} \sum_{i=1}^{N} D_{\{p:i=(p,j)\}} \cdot \theta_i \cdot f_i \cdot x_{ic} \qquad \forall m \in M, c \in C \qquad (5.4)$$

$$z_{mc} = \left\lceil \frac{1}{H_m} \sum_{i=1}^{N} D_{\{p:i=(p,j)\}} \cdot \theta_i \cdot f_i \cdot x_{ic} \right\rceil \qquad \forall m \in M, c \in C \qquad (5.5)$$

5.3.1.2 Initial Solution

The initial solution is obtained by the following two steps greedy procedure which aims to minimize intercellular transfer costs:

Step 1. In the first step, all products are assigned into a single cell c, resulting in a null cost of intercellular transfers. This configuration (i.e. the trivial solution) is an infeasible solution, given that the cell size constraints are not satisfied.

Step 2. In the second step, the procedure looks for feasibility while reducing machine acquisition costs, at the expense of slight increases in the intercellular transfer cost, by moving disassembly tasks into their own cells.

   a. The procedure begins by identifying the product p^* with the lowest expected cost of intercellular transfers, where $p^* = argmin_{(p)}\{\alpha_p \times D_p\}$
   b. Then, the procedure identifies the disassembly task i^* associated to product p^* that requires the machine m with the largest duplication cost (i.e. identify the machine that if duplicated would increase more the total cost).
   c. The remaining μ disassembly task associated to product p^* are moved from the current cell c to a new cell c' with their corresponding resources, where $\mu = min\{NOP(p^*) - 1,\ W(1) - L\}$ and $NOP(p^*)$ is the total number of disassembly task associated to product p^*.
   d. If $W(c') + n \leq L$ then continue with the same cell c' else open a new cell.

Step 3. Repeat step 2, until $L \leq W(c') \leq U$.

Finally, the number of machines allotted to cells and the total cost are computed. As will be seen, solutions with smaller number of cells are preferable, and hence the search will start from a solution with a large number of cells and will progressively try to reduce this number while intermediate solutions are being explored.

5.3.1.3 Neighborhood Structures and Move Definition

TS metaheuristic is an iterative procedure based on a neighborhood search technique; where at each iteration we move from a current solution x to a new solution x' in a neighborhood $N(x)$ of x. Hence the notion of neighborhood is a basic component of this procedure. The *neighborhood* of a solution $N(x)$ is the set of all the solutions that can be reached from the current solution x through a single *move*. This means that in general, the neighbor solutions x' are generated by applying slight modifications or moves $m \in MOVE$ to a solution x. The set of moves $MOVE$ is specific to each problem. Mathematically speaking, we denote by $x' = x \oplus m$ the neighbor solution x' generated by applying the move $m \in MOVE$ to solution x. Note

that if any constraint is imposed, not every move is feasible. The moves that provide a feasible solution are called feasible moves. Then, the neighborhood $N(x)$ of x is specified as $N(x) = \{x': x' = x \oplus m \text{ for some } m \in MOVE\}$.

In the cell formation problem the moves may be defined in several ways. The most common types of moves considered are the swap, insertion, union, and split moves (see for example Adenso-Díaz *et al.*, 2001; Tavakkoli-Moghaddam *et al.*, 2005; Cao and Chen, 2005; Schaller, 2005 and Schaller, 2007). The aim of this set of moves is to manipulate the assignment of tasks into cells. In the search of a better solution, for our particular case a move will be feasible if the cell cardinality constraints are satisfied. This means that a move would be feasible if $L \leq W(c) \leq U$. If a given move is implemented, $W(c)$ will be updated in a proper manner. The moves modified to deal with the disassembly cell formation problem are depicted below.

Insertion Move, m_I. An operation is deleted from its current cell and included in a different one. Thus for each disassembly task i such that $x_{ic} = 1$, if $x \oplus m_I$ then $x_{ic} = 0$, while $x_{ic'} = 1$ for each $c' \neq c \in C$. The cardinality of the source cell must be greater than the minimum number of operations per cell $W(c) > L$. In addition, the cardinality of the target cell must be less than the maximum number of operations per cell $W(c') < U$. The move will increment the cardinality of the target cell by one unit, reducing that of the source cell in the same quantity. Therefore $W(c)$ is updated as follows: $W(c) = W(c) - 1$ and $W(c') = W(c') + 1$.

Swap Move, m_E. Two operations belonging to different cells are swapped, while the rest of the variables remain unchanged. Thus for each pair of disassembly tasks i and j such that $x_{ic} = 1$ and $x_{jc'} = 1$, if $x \oplus m_E$ then $x_{ic} = 0$ while $x_{j'c} = 1$, and $x_{jc'} = 0$ while $x_{ic'} = 1$. Hence this move will not modify any of the values $W(c)$.

Union Move, m_U. All the operations from two different cells are put together. Thus for each pair of cells c and c' such that: i) $W(c) > L$, ii) $W(c') > L$, and iii) $W(c) + W(c') \leq U$, if $x \oplus m_U$ then all the disassembly tasks i such that $x_{ic'} = 1$, turn into $x_{ic'} = 0$ while $x_{ic} = 1$. $W(c)$ and $W(c')$ are updated as follows: $W(c) = W(c) + W(c')$ and $W(c') = 0$.

Splitting Move, m_S. To allow an increase in the number of cells in the search process, we consider the possibility of splitting one of the cells into two. This is only possible if the

cardinality of the cell to be split is large enough to produce two cells with more than L operations each. Therefore we select *nopers* operations ($L \leq nopers \leq surplus$) to be removed from cell c and reinserted into a cell c', where c' is originally an empty cell, and $surplus = W(c) - L$ is defined as the maximum number of operations that can be removed from cell c and reinserted into a cell c'. $W(c)$ and $W(c')$ will be updated as: $W(c) = W(c) - nopers$ and $W(c') = nopers$, what is exactly the opposite to the union of two cells.

The set of moves *MOVE* is the union of the four moves defined above $M = m_E \cup m_I \cup m_U \cup m_S$. Consequently, each move has its corresponding neighborhood, and we define neighborhood $N(x)$ as the union of all these types of neighborhoods $N(x) = N_E(x) \cup N_I(x) \cup N_U(x) \cup N_S(x)$.

Given the multiple representation of a single solution, not all the elements in $N(x)$ must be considered (because of cell size constraints), considerably reducing the number of candidates to explore. As will be seen, solutions with smaller number of cells are preferable, and hence the search will start from a solution with a large number of cells and will progressively try to reduce this number while intermediate solutions are being explored. Therefore, in relation to the neighborhood $N_S(x)$, good results are obtained when its corresponding move is considered only after δ number of iterations without improvement, thus acting as a *first-level search diversifier* that increase the number of cells once more and continue the search in another region of the solution space.

5.3.1.4 Short Term Memory Structures

Since the value of the objective function is not necessarily strictly increasing, a safeguard against cycling in local optimal solutions is required. TS uses a short term memory strategy to avoid returning to a solution already visited, which keep track of some attributes or characteristics of the moves that were recently used. In our TS implementation, we use two *short term memory structures* to keep track of moves that remain *tabu* during a specified time horizon (*tabu tenure*).

Swap Matrix. We label *tabu* those moves that reverse the exchange of disassembly tasks that recently had been swapped. The short term memory structure takes the form of a matrix that has the same dimensions than the solution x_{ic}, i.e. as many rows as total number of disassembly tasks (N) and as many columns as the number of cells to form (C). If a swap

move is implemented in iteration *iter* then the value *iter* + ρ_1 is added to the corresponding entries in the swap matrix, i.e. *swap_matrix*(*i*, *c*) = *iter* + ρ_1 and *swap_matrix*(*j*, *c*') = *iter* + ρ_1, where ρ_1 is the corresponding tabu tenure for swap. When evaluating the neighborhood of a solution in an iteration *iter*, a certain swap move will be discarded if *swap_matrix*(*i*, *c*)>*iter* or *swap_matrix*(*j*, *c*') > *iter*, with the exception described in the aspiration criterion.

Insertion Matrix. We also label *tabu* those moves that involve the reinsertion of any disassembly task that recently had been eliminated from a given cell. This second short term memory structure also takes the form of a matrix with the same dimensions than the solution x_{ic}. If an insertion move is implemented in iteration *iter* then the value *iter* + ρ_2 is added to the corresponding entry in the insertion matrix, i.e. *insertion_matrix*(*i*, *c*') = *iter* + ρ_2, where ρ_2 is the corresponding tabu tenure. When evaluating the neighborhood of a solution in an iteration *iter*, a certain insertion move will be discarded if *insertion_matrix*(*i*, *c*) > *iter*, with the exception described in the aspiration criterion.

The size of the tabu tenure for each matrix is an important parameter regarding the efficiency of the heuristic; however its best value is not easily determined. Furthermore, it is often more efficient to use a *variable tabu tenure size*. In this variant, the tabu tenure size used at each iteration is a random integer such that $\rho_{\min} \leq \rho \leq \rho_{\max}$. The extreme values used in this implementation were: $N/7 \leq \rho_1 \leq N/3$ for the swap move and $N/5 \leq \rho_2 \leq N/3$ for the insertion move.

5.3.1.5 Aspiration Criterion

In our implementation, we use the simple aspiration criterion that overrules the tabu restriction if the candidate solution is better than the current, best encountered one. This means that if a move will result in an improved incumbent value, then the move will be implemented even if it is tabu. Other refinements that could be implemented include overruling when the candidate is better than all the solutions in a region, or in a prescribed number of iterations.

5.3.1.6 Diversification Procedure

If η iterations have elapsed since the last update of the best solution then diversification move is implemented to restart the search in another zone. The diversification move will assign

each operation to a new cell, being more likely the assignment to the cells to which it was assigned more rarely in the past. To this end, the frequency (i.e. the fraction) of the solutions explored in which the disassembly task i has been assigned to each cell c (u_{ic}) is computed. Specifically, every time that a move is performed u_{ic} is updated according to:

$$u_{ic} \leftarrow \alpha \cdot x_{ic} + (1 - \alpha) \cdot u_{ic} \qquad \forall i, c \qquad (5.6)$$

where the smoothing coefficient $\alpha \in (0, 1)$ controls the weight given to the most recent visited solutions. The assignment of a disassembly task i to a cell c is made with a probability $v_{ic} = 1 - u_{ic}$. This is implemented through the table look-up method that is normally used to generate samples of a discrete distribution. After a number of experiments we opted for a fixed $\alpha = 0.75$.

5.3.1.7 Stopping Criterion

If γ iterations have elapsed without an improved solution, or *max_iter* iterations have been performed then the algorithm stops. We decide to fix $\gamma = 200$. With regard to the maximum total number of iterations before stopping, *max_iter*, this can be assigned a sufficiently high value (e.g. 1000) in order to guarantee that the search will not end too soon.

5.3.1.8 Outline of the Tabu Search Procedure

A detailed description of the TS procedure is depicted in Fig. 5.11.

```
procedure TabuSearch( )
Input: Data structure for the disassembly cells problem.
Output: Disassembly cellular configuration.

0     Set iter ← 0; iter_without_improv ← 0; iter_best ← 0;
1     value(x^curr) ← InitSol(x^curr);
2     x^best ← x^curr;
3     while iter < max_iter and iter_without_improve < γ do
4        ++iter; ++iter_without_improv;
5        swap_move(). Generate best move N_E(x^curr)
6        insertion_move(). Generate best move in N_I(x^curr)
7        union_move(). Generate best move in N_U(x^curr)
8        if iter_without_improv > δ then
9           split_move(). Generate best move in N_S(x^curr)
10       end if
11       x^curr ← "best" move from all those generated;
12       Update corresponding information;
13       if value(x^curr) < value(x^best) then
14          x^best ← x^curr;
15          iter_best ← iter;  iter_without_improv ← 0;
16       end if
17       if iter_without_improv == η then
18          diversification(). Diversification procedure.
19          iter_without_improv ← 0;
20       enf if
21    end while;
22    Return x^best.
end procedure GreedyRandomizedConstruction
```

Figure 5.11 Tabu search algorithm for the disassembly cell formation problem.

5.3.2 A Variable Neighborhood Search Algorithm for the Robust and Reconfigurable Disassembly Cell Formation Problems

In this section we introduce a Variable Neighborhood Algorithm (VNS) based heuristic to solve the robust (DCFP-R) and reconfigurable (DCFP-D) disassembly cell formation problems formulated in Sections 4.3.2 and 4.3.3. VNS is a meta-heuristic approach that has been used for solving many different types of combinatorial optimization problems. It was introduced by Mladenović and Hansen (1997), and since then it has had a rapid development in both methods and applications (see Hansen *et al.* (2009) for a review). The VNS is a simple and effective search procedure that performs a systematic change of neighborhood within the local search algorithm. To develop an effective VNS algorithm two kinds of neighborhood structures must be defined to achieve a valuable neighborhood change: the neighborhood structures for shake $N^{SK}_k(x)$ and the neighborhood structures for local search

$N^{LS}{}_l(x)$. The indices, k and l, are used to identify the neighborhoods structures within each function (shake and local search respectively), which ease switching from one to another neighborhood. Note that $1 \leq k \leq k_{max}$ and $1 \leq l \leq l_{max}$, where k_{max} and l_{max} denote the boundaries of both indices respectively. A typical VNS metaheuristic gets an initial solution, x, and then manipulates it through the three steps that enclose the main loop of the algorithm: shaking, local search and neighborhood change. Initially the neighborhood indexes k and l are set to 1, then the shake procedure acts as a diversifier of the incumbent solution, i.e., it produces a solution x' within $N^{SK}{}_k(x)$, which moves away from the current solution x. The local search procedure takes x' as a starting solution, and explores for an improved solution within the predefined neighborhood structure $N^{LS}{}_l(x)$ until a local minimum x'' is found. If x'' is better than the incumbent solution x, then x is replaced with x''. The neighborhood change procedure needs to answer two questions: (i) whether to move or not: what criteria to use in changing the incumbent solution; and (ii) next neighborhood: which will be the next neighborhoods $N^{SK}{}_k(x)$ and $N^{LS}{}_l(x)$ for shake and local search functions respectively, in case the solution is changed and in case the incumbent is not changed. The method repeats these three steps until the stopping condition is met.

The motivation for using VNS in this particular application is based on the fact that although a simple metaheuristic, which combines a systematic change of neighborhoods with a local search; it has shown to be highly efficient and easy to implement. Furthermore in the literature several neighborhood structures have been proposed for the cell formation problem which have been used under different metaheuristic schemes such as tabu search, genetic algorithms and simulated annealing among others. A modification of some of such structures can be easily applied to define the shaking and local search functions in a proper manner with the aim of reaching high quality solutions. It should be noted that the two models described above, namely DCFP-D and DCFP-R, have the same structure and variables, and if we replace the term scenario with the term period, the only difference lies in the evaluation function given by the distinct features of each objective function. We now present in more detail the steps of our VNS based heuristic for solving the robust and reconfigurable disassembly cell formation problems.

5.3.3.1 Solution Coding and Evaluation

When designing an algorithm it is important to define a good structure for encoding the solution. An appropriate selection of a solution coding provides an efficient way for

implementing the moves and evaluating the solutions. For designing this algorithm we improve the solution coding regarding the one used in the Tabu Search algorithm. Now for a formal definition of the solution space in the DCFP, let N be the number of disassembly tasks to be considered. Since the purpose of the DCFP is to group the disassembly tasks into cells with their respective resources, whilst satisfying some additional constraints, the simplest structure might be to consider an array of N elements in which the number of the cell to which the i-th task is assigned in a particular solution is indicated in the i-th position of the array. With this array the feasibility condition: $\Sigma_{c\in C}\, x_{pjc} = 1$ for all pj, which sets that each disassembly task must be assigned to one and only one cell, is easily satisfied. Once the solution concept has been thus defined, the concept of a feasible solution is defined for each version of the disassembly cells formation problem.

Reconfigurable DCFP. In the reconfigurable version of the DCFP the purpose is to group for every period the disassembly tasks into cells with their respective resources, whilst machines reallocation is allowed to deal with demand variability. Thus, a feasible solution x in the DCFP-D define a set of arrays $\hat{x}_t = \langle x_t(1), x_t(2), \ldots, x_t(N)\rangle$ for each period ($x = \bigcup_{t\in T} \hat{x}_t$) that does not violate the cardinality constraints. Each position of $\hat{x}_t$ indicates the cell c to which the i-th disassembly task has been assigned in period t ($x_t(i) = c$), see example in Fig. 5.12. Therefore, the use of an appropriate data structure is required to identify the feasible solution.

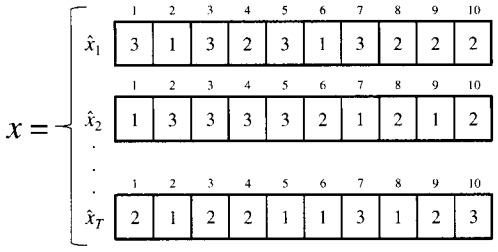


Figure 5.12 Solution coding for the reconfigurable approach. This figure shows a possible representation of a solution with three cells for a problem with 5 parts (where each one requires 2 disassembly operations) therefore there are 10 tasks. Each position of $\hat{x}_t$ indicates the cell c to which the i-th disassembly task has been assigned in period t ($x_t(i) = c$). For instance $x_2(7) = 1$ means that in period 2 the disassembly task 7 has been assigned to cell 1.

To comply with cardinality constraints a feasible solution x must satisfy that $L \leq W_t(c) \leq U$ for all c and for all t, where $W_t(c)$ is the number of disassembly tasks assigned to cell c during period t. For a given solution x, the number of machines of each type required in each cell during each period (z_{mct}) can be obtained using the constraint (4.47) as follows:

$$z_{mct} = \left\lceil \frac{1}{H_m} \sum_{p \in P} \sum_{i=l(p)}^{u(p)} \sum_{\hat{x}_t(i)=c} D_p^t \cdot \theta_i \cdot f_i \right\rceil \tag{5.7}$$

where $l(p)$ and $u(p)$ are the first and last disassembly tasks associated to product p respectively. Consequently the total number of machines type m reallocated in each cell c (Δ_{mc}) can be computed as follows: if $\Delta_{mc} \geq 0$, then $z^+_{mct} = \Delta_{mc}$ and $z^-_{mct} = 0$; otherwise $z^-_{mct} = \text{abs}(\Delta_{mc})$, and $z^+_{mct} = 0$, where $\Delta_{mc} = z_{mc(t)} - z_{mc(t-1)}$.

After the number and types of machines for each cell have been calculated the total cost which includes the machines acquisition and reallocation as well as the intercellular transportation costs can be calculated as follows:

$$\begin{aligned} \text{cost}(x) = & \sum_{t \in T} \sum_{m \in M} \sum_{c \in C} \left(\beta_m \cdot z_{mct} + \gamma_m \cdot \left(z^+_{mct} + z^-_{mct} \right) \right) \\ & + \sum_{t \in T} \sum_{p \in P} \sum_{i=l(p)}^{u(p)-1} \sum_{j=i+1; \hat{x}_t(i) \neq \hat{x}_t(j)}^{u(p)} \alpha_p \cdot D_p^t \cdot f_i \cdot \pi_{ij} \end{aligned} \tag{5.8}$$

Robust DCFP. The robust version of the DCFP aims at finding a robust cellular configuration of machines that performs well across all possible realizations of scenarios to deal with product demand variability. It should be noted that the cells are formed and the machines are assigned to them, and then decisions regarding disassembly tasks and flow patterns are made for each scenario realization. Thus, a feasible solution x in the DCFP-R define a set of arrays $\hat{x}_s = \langle x_s(1), x_s(2), \ldots, x_s(N) \rangle$ for each one of the S scenarios, i.e. $x = \bigcup_{s \in S} \hat{x}_s$, that does not violate the cardinality constraints. Each position of $\hat{x}_t$ indicates the cell c to which the i-th disassembly task has been assigned in scenario s ($x_s(i) = c$), see example in Fig. 5.13.

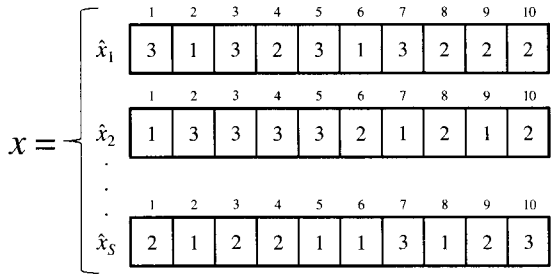


Figure 5.13 Solution coding for the robust approach. This figure shows a possible representation of a solution with three cells for a problem with 5 parts (where each one requires 2 disassembly operations) therefore there are 10 tasks. Each position of $\hat{x}_s$ indicates the cell c to which the i-th disassembly task has been assigned in scenario s ($x_s(i) = c$). For instance $x_2(5) = 3$ means that in scenario 2 the disassembly task 5 has been assigned to cell 3.

To comply with cardinality constraints a feasible solution x must satisfy that $L \leq W_s(c) \leq U$ for all c and for all s, where $W_s(c)$ is the number of disassembly tasks assigned to cell c under scenario s. For a given solution x the number of machines of each type required in each cell under scenario s ($\hat{z}^s_{mc}$) can be determined using the constraint (4.59) as follows:

$$\hat{z}^s_{mc} = \left\lceil \frac{1}{H_m} \sum_{p \in P} \sum_{i=l(p):\hat{x}_s(i)=c}^{u(p)} D^s_p \cdot \theta_i \cdot f_i \right\rceil \tag{5.9}$$

Thus, the number of machines required in the robust solution is given by $z_{mc} = \max_s\{\hat{z}^s_{mc}\}$, while the variable $em^s_c = z_{mc} - \hat{z}^s_{mc}$. Finally, it is easy to compute the total cost associated to a solution x as follows:

$$\begin{aligned} \text{cost}(x) = & \sum_{m \in M} \sum_{c \in C} \beta_m \cdot z_{mc} + \sum_{s \in S} \rho_s \sum_{p \in P} \sum_{i=l(p)}^{u(p)-1} \sum_{j=i+1:\hat{x}_s(i) \neq \hat{x}_s(j)}^{u(p)} \alpha_p \cdot D^s_p \cdot f_i \cdot \pi_{ij} \\ & + \sum_{s \in S} \rho_s \cdot \sum_{c \in C} \left(\lambda_1 \cdot el^s_c + \lambda_2 \cdot eu^s_c + \lambda_3 \cdot em^s_c \right) \end{aligned} \tag{5.10}$$

To facilitate the exposition of the proposed metaheuristic from hereafter, unless otherwise stated, index s will refer to the stage –period for the reconfigurable problem and scenario for the robust problem– within a solution.

5.3.3.2 Initial Solution

To obtain an initial solution of good quality, the greedy heuristic procedure designed for the basic DCFP was adapted. The heuristic procedure tries to find a feasible assignment of disassembly tasks to cells, period by period or scenario by scenario. The procedure begins with an infeasible solution, and then step by step it looks for feasibility while the machine amortization costs are minimized at the expense of slight increase of intercellular movement costs. It should be noted that although in the context of reconfigurable cell formation, the allocation of machines between two consecutive periods directly affects the costs of reconfiguration and thus the total cost, the objective of this procedure is to find a feasible solution in each period, so that the costs of reallocation are not considered in this phase. Subsequently, the procedure searches moves to reduce overall costs.

The following steps are repeated for all stages:

Step 1. In the first step, all products are assigned into a single cell c, resulting in a null cost of intercellular transfers. This configuration (i.e. the trivial solution) is an infeasible solution, given that the cell size constraints are not satisfied.

Step 2. In the second step, the procedure looks for feasibility while reducing machine acquisition costs, at the expense of slight increases in the intercellular transfer cost, by moving disassembly tasks into their own cells.

   a. The procedure begins by identifying the product p^* with the lowest expected cost of intercellular transfers, where $p^* = \text{argmin}_{(p)}\{\alpha_p \times D^s_p\}$.

   b. Then, the procedure identifies the disassembly task i^* associated to product p^* that requires the machine m with the largest duplication cost (i.e. identify the machine that if duplicated would increase more the total cost).

   c. The remaining n disassembly task associated to product p^* are moved from the current cell c to a new cell c' with their corresponding resources, where $n = \min\{NOP(p^*) - 1,\ W_s(1) - L\}$ and $NOP(p^*)$ is the total number of disassembly task associated to product p^*.

   d. If $W_s(c') + n \leq L$ then continue with the same cell c' else open a new cell.

Step 3. Repeat step 2, until $L \leq W_s(c') \leq U$.

Step 4. Update $\hat{x}_s$ and proceed with the next stage s, until the solution x is completed.

Finally, the number of machines allotted to cells and the total cost are computed for the corresponding problem. As will be seen, in both problems solutions with smaller number of cells are preferable (as in the basic DCFP), and hence the search will start from a solution with a large number of cells and will progressively try to reduce this number while intermediate solutions are being explored.

5.3.3.3 Neighborhood Structures and Move Definition

VNS metaheuristic is an iterative procedure that performs a systematic change of neighborhood within the local search algorithm. Hence the notion of neighborhood is also a basic component of this procedure. In the remaining of this section, we define a set of neighborhood structures that will be used in the shaking and local search phases of the algorithm. In general, we use the same moves than in the TS algorithm, but we also include new moves. The moves modified to deal with our particular problems are briefly depicted below. A detailed description of the moves is provided in Appendix A.

Insertion Move, m_I. Within this move a disassembly task is deleted from its current cell and reinserted in a different one. See Fig. 5.14.

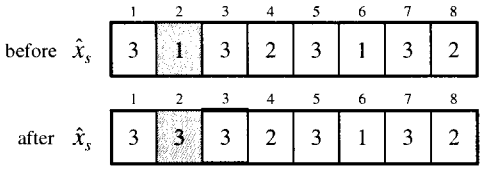


Figure 5.14 Insertion move. In the initial solution $x_s(2) = 1$, then after performing $x \oplus m_I$ a neighboring solution is $x_s(2) = 3$.

Swap Move, m_E. Within this move two disassembly tasks belonging to different cells are swapping, while the rest of the tasks remain unchanged. See Fig. 5.15.

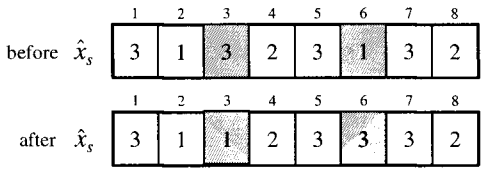


Figure 5.15 Swap move. In the initial solution $x_s(3) = 3$ and $x_s(6) = 1$, then after performing $x \oplus m_E$ a neighboring solution is $x_s(3) = 1$ and $x_s(6) = 3$.

Union Move, m_U. Within this move all the disassembly tasks of two different cells are put together. The implementation of this move is different in the robust problem than in the reconfigurable problem. The main difference lies in the fact that while in the reconfigurable problem the number of cells can vary from period to period, in the robust problem the number of cells must remain constant for all scenarios. See Fig. 5.16 for an example of the union move for the robust problem and Fig. 5.17 for an example for the reconfigurable problem.

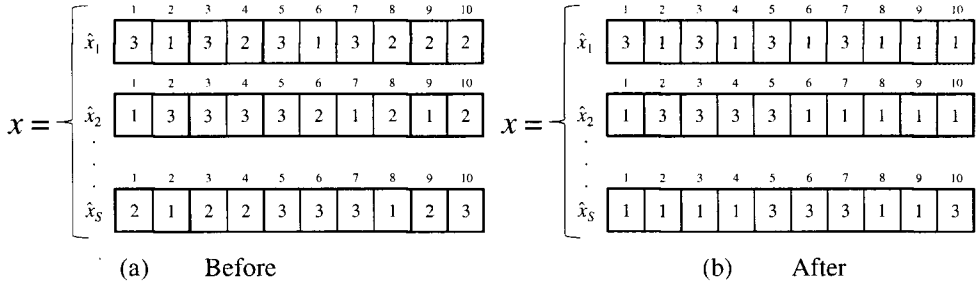


Figure 5.16 Union move for the robust problem. For illustration cells 1 and 2 are joined in all scenarios.

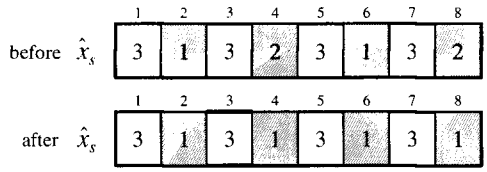


Figure 5.17 Union move for the reconfigurable problem. For illustration in the initial solution $x_s(2) = 1$, $x_s(4) = 2$, $x_s(6) = 1$, and $x_s(8) = 2$, then after performing $x \oplus m_U$ a neighboring solution is $x_s(2) = 1$, $x_s(4) = 1$, $x_s(6) = 1$, and $x_s(8) = 1$.

Split Move, m_S. To allow an increase in the number of cells in the search process, we consider the possibility of splitting one of the cells into two. As in the union move, the split move in the robust problem increments the number of cells in all scenarios, while in the reconfigurable problem the number of cells may be increased in only one period. The split move for the reconfigurable problem is illustrated in Fig. 5.18.

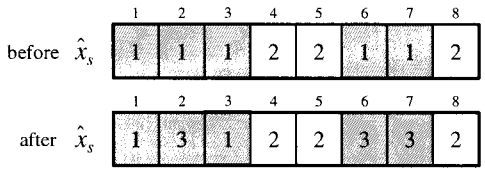


Figure 5.18 Split move for the reconfigurable problem. In the initial solution $x_s(1) = 1$, $x_s(2) = 1$, $x_s(3) = 1$, $x_s(6) = 1$ and $x_s(7) = 1$, then after performing the move $x \oplus m_S$ a neighboring solution is $x_s(1) = 1$, $x_s(2) = 3$, $x_s(3) = 1$, $x_s(6) = 3$ and $x_s(7) = 3$.

λ-Insertion Move, m_λ. In previous experiments we found that sometimes it is required to move more than one disassembly task a time which belongs to the same cell to obtain an improving move, therefore we consider the movement of λ disassembly tasks a time. Within this move λ disassembly tasks are deleted from their current cell and reinserted into a different one. For an illustration see Fig. 5.19.

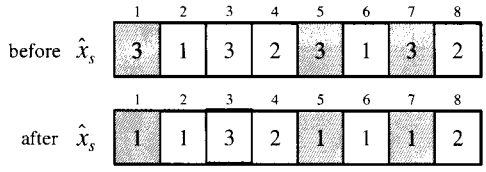


Figure 5.19 The λ-insertion move. Given λ = 3, three disassembly tasks that belong to the same cell are reinserted in a different cell. In the initial solution $x_s(1) = 3$, $x_s(5) = 3$ and $x_s(7) = 3$, then after performing $x \oplus m_\lambda$ a neighboring solution is $x_s(1) = 1$, $x_s(5) = 1$ and $x_s(7) = 1$.

5.3.3.4 Shaking

The shake procedure in the VNS heuristic allows escaping from a local minimum without destroying completely the good properties of the current solution. It is expected that the shaking procedure will diversify the incumbent solution x^*, i.e., it will produce a solution x' which moves away from the current solution f. To this end, our shake procedure consists of three neighborhood structures: the $N^{SK}{}_1(x)$ which use the split move, $N^{SK}{}_2(x)$ which use the random insertion move and $N^{SK}{}_3(x)$ which combines the random insertion move and the split move successively. The rotation of the three neighborhoods is described within the neighborhood change section.

Our first shaking procedure uses the split move defined above. We select μ disassembly tasks from cell c at random and reallocate them in cell c'. Our second shaking procedure uses the random insertion move defined above. Given a size η for the second shaking procedure, we choose η times a disassembly task at random and we insert it into a different cell. For $r = 1$ to η we randomly select the r-th disassembly tasks form the N available tasks to be reinserted into a different cell. If $x_s(i_r) = c$, we randomly select a c' ($c \neq c'$) and assess if $L \leq W_s(c') + 1 \leq U$ and $W_s(c) - 1 \geq L$ to evaluate the feasibility of the move. If the move is feasible then the disassembly task i_r moves from $x_s(i_r) = c$ to $x_s(i_r) = c'$, also $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) = W_s(c) - 1$, and $W_s(c') = W_s(c') + 1$, and r is updated as $r + 1$, else $W_s(c)$, $W_s(c')$ and r remains unchanged. We do not allow η become larger than $\lceil N/4 \rceil$. Finally, in our third shaking procedure, given a solution x we randomly reallocate η disassembly tasks into a different cell and then we split two cells to obtain a solution x'.

The random insertion and split neighborhood schemes were chosen for the shaking phase, as they can provide changes in the structure of the solution by changing both the allocation of disassembly tasks into cells and the number of cells, and hence provide diversification to the search. The split move provides solutions which cannot be reached via insertion or swap moves due to the capacity constraints. On the other hand, by applying η times the random insertion move very diverse solutions are reached.

5.3.3.5 Local Search

The local search looks for an improving solution within the type of neighborhood structure being explored. Our local search procedure uses four neighborhood structures: the insertion

neighborhood $N_I(x)$, the swap neighborhood $N_E(x)$, the union neighborhood $N_U(x)$, and the λ-insertion neighborhood $N_\lambda(x)$. These neighborhood schemes were chosen for the local search phase, as they have been used in previous research to provide quality local optimal solutions to the cell formation problem.

These neighborhoods are used according the following schema:

- $N^{LS}{}_1(x) = N_E(x)$
- $N^{LS}{}_2(x) = N_E(x) \cup N_I(x)$
- $N^{LS}{}_3(x) = N_E(x) \cup N_I(x) \cup N_U(x)$
- $N^{LS}{}_4(x) = N_E(x) \cup N_I(x) \cup N_U(x) \cup N_{\lambda=2}(x)$
- $N^{LS}{}_5(x) = N_E(x) \cup N_I(x) \cup N_U(x) \cup N_{\lambda=2}(x) \cup N_{\lambda=3}(x)$

Thus, given a solution, a greedy local search with respect to $N^{LS}{}_l(x) \in \{N^{LS}{}_1(x), N^{LS}{}_2(x), N^{LS}{}_3(x), N^{LS}{}_4(x), N^{LS}{}_5(x)\}$ is implemented by choosing each time the best possible move among the nested neighborhoods (see pseudo-code in Fig. 5.20). Note that the cardinality of each nested local search neighborhood is successively incremented. Therefore, even though the size of the neighborhood $N^{LS}{}_l(x)$ is reduced (because of cell size constraints, which consequently reduce the number of candidate neighboring solutions to explore) its cardinality could be too large for instances of realistic size to explore them in a best improvement basis. Note that when λ becomes larger there are more improvement choices and improvement results may be better. However, computation time increases exponentially with λ, which complicate a complete exploration of the λ-insert neighborhoods. In our case, λ is limited to 2 and 3.

There are two ways to improve the current solution: *best improvement* and *first improvement*. First improvement searches the current neighborhood until it finds the first solution which is better than the current solution. Best improvement searches the whole neighborhood and chooses the best solution found in each improvement step. Neighborhoods $N_E(x)$, $N_I(x)$ and $N_U(x)$ will be explored under the best improvement strategy whereas neighborhoods $N_{\lambda=2}(x)$ and $N_{\lambda=3}(x)$ will be explored under the first improvement strategy.

```
    procedure LocalSeach (x^0, l)
0   x ← x^0; neighborhood ← l;
1   switch ( neighborhood ) {
2
    case 1:
3        do {
4           local_optima ← true;
5           x^swap ← Best move obtained from N_E(x) by swap;
6           x´ ← x^swap;
7           if f(x´) < f(x) then
8             x ← x´;  f(x) ← f(x´); local_optima ← false;
9           end if
10       }while local_optima == false
11       break;
12  .
    .
13  case 5:
14       do {
15          local_optima ← true;
16          x^swap ← Best move obtained from N_E(x) by swap;
17          x^insert ← Best move obtained from N_I(x) by insertion;
18          x^union ← Best move obtained from N_U(x) by union;
19          x^λ-insert ← Best move obtained from N_λ(x) by λ-insertion;
20          x´ ← Best move obtained among all neighborhoods;
21          if f(x´) < f(x) then
22            x ← x´;  f(x) ← f(x´); local_optima ← false;
23          end if
24       }while local_optima == false
25       break;
    }
    end LocalSearch Procedure
```

Figure 5.20 Pseudo-code of the Local Search for the robust and reconfigurable DCFP.

Since the insertion move could reverse the random insertion generated in the shaking phase, it is applied after the first local search neighborhood. The split move diversify the search increasing the number of cells, which forces the swap and insertion moves to continue the search in another region of the solution space. However, as will be seen, solutions with smaller number of cells are preferable, and hence the union move is implemented in the local search. In relation to the λ-insertion neighborhood, its solution space is incremented as λ increase; therefore it is implemented at the end.

5.3.3.6 Neighborhood Change

Neighborhood change is a very important step in the VNS framework. Several different neighborhood change strategies within VNS have been suggested in the literature. The neighborhood change function requires to answer two questions: (*i*) whether to move or not:

what criteria to use in changing the incumbent solution; and (*ii*) next neighborhood: what will be the next neighborhood in a case when the solution is changed and in a case when the incumbent is not changed.

Since we use several neighborhoods for both shaking and local search, we need to define for each whether to move or not from one neighborhood to another and what will be the next neighborhood.

Move or not. The acceptance criterion for moving, within the local search, from x' to x'' is based on the objective function value, if $f(x'') < f(x')$ then inner solution x' is changed to x''. While the acceptance criterion for moving, within the shaking, from x^* to x' is also based on the objective function value, if $f(x') < f(x^*)$ then incumbent solution x^* is changed to x'.

Next neighborhood. We have defined three neighborhoods for shaking and four neighborhoods for local search. Within the local search, if the inner solution is improved then we continue the search in $l = 1$; otherwise, the next neighborhood is explored, i.e. $l = l + 1$. For the shaking, if the incumbent solution is improved then we use the first neighborhood for shaking to diversify the solution ($k = 1$); otherwise, the next neighborhood for shaking is considered.

5.3.3.7 Outline of the Proposed VNS Procedure

With the above introduced concepts and explanations, the proposed VNS procedure for the disassembly cells formation problem with demand uncertainty is described within this section. A detailed pseudo-code is given in Fig. 5.21. In line 1 the InitSol( ) procedure is used to generate an initial solution x^0 applying the greedy heuristic defined above, then a greedy local search among all the nested neighborhoods is performed to generate a high quality incumbent solution x^*. In line 5 the incumbent solution is perturbed using the k-th neighborhood of the Shake( ) procedure to obtain a solution x'. In line 8 the local minimum of the solution x' within the l-th neighborhood of the LocalSearch( ) procedure is searched to obtain the improved solution x''. If x' is improved then in line 10 the inner loop solution x' is updated with x'', and the next neighborhood of the LocalSearch(x) procedure is explored. After exploring all the neighborhoods of the local search, the inner loop solution is compared against the incumbent solution to decide if a change in the neighborhood of the Shake( ) procedure must be performed.

```
procedure VNSdcells (k_max, l_max)
0
1    x^0 ← InitSol(Π, z);
2    x* ← LocalSearch(x^0, l_max);
3    k ← 1; x ← x*;
4    while k ≤ k_max do
5       x´ ← Shake(x*, k);
6       l ← 1;
7       do{
8          x´´ ← LocalSearch(x´, l);
9          if f(x´´) < f(x´) then
10            x´ ← x´´;
11            f(x´) ← f(x´´);
12            l ← 1;
13         else l ← l + 1; end if
14      }while l  ≤ l_max ;
15      if f(x´) < f(x*) then
16         x* ← x´;
17         f(x*) ← f(x´);
18         k ← 1;
29      else k ← k + 1; end if
20   end while
21
end VNSdcells procedure
```

Figure 5.21 VNS algorithm for the robust and reconfigurable disassembly cell formation problems.

Chapter 6

Experimental Results

The objective of this chapter is to describe the experimental frameworks and to present the results obtained in each problem addressed in this thesis. In Section 6.1 the results obtained after applying the methodological approach for redesigning the WEEE collection network in Galicia are shown. Specifically, Section 6.1.1 explains how the data were gathered while from Section 6.1.2 through Section 6.1.5 the results of each element of the methodological framework are discussed. Section 6.1.6 compares the performance of the current configuration versus the proposed configuration. In Section 6.2 the experimental results obtained with an implementation of the GRASP algorithm proposed in Section 5.2 are presented. Section 6.2.1 describes the instance generator while the experimental framework and the computational results are discussed and analyzed in Section 6.2.2. Finally in Section 6.3 the experimental results obtained with an implementation of the TS and the VNS algorithms proposed in Section 5.3 are discussed and analyzed. Section 6.3.1 describe the application of the proposed TS algorithm to solve a real-world instance, while the computational results obtained by an implementation of the VNS proposed for the robust and the reconfigurable problems of the DCFP are discussed and analyzed in Section 6.3.3 and Section 6.3.4 respectively.

6.1 Results of the WEEE Collection Network

6.1.1 Data Gathering to Design the Recovery Network in Galicia

As discussed in Section 3.1, the CMS serves 707 geographically dispersed stores in Galicia from which the total amount of items collected per week are recorded. Data related to the fleet of vehicles used in the harvesting process, distances and travel times, as well as the real amount of items collected along 2008 in every store were provided by the CMS. In general two types of vehicles with different load capacity (15 and 25 items) are used in the harvesting process. The first (hereafter referred as vehicle type 1) are smaller than the corresponding vehicles with a load capacity of 25 items (hereinafter referred as vehicle type 2), therefore they can access to all stores, although the latter have a loading platform that facilitates the collection task. Annual fixed costs are respectively 34,993.47 and 41,723.96 €/year, while the variable costs reach 0.15 and 0.17 €/km. Every depot works an average of 40 hours per week, 5 working days, 245 days per year, and according to strategic requirements set by CMS, recovered items should stay at must an average of one week at the depot. Currently there are two recovery plants; one is able to process all items that are sent, while the second is only capable of processing at most 40% of the retrieved items given that it cannot process refrigerators. However, to ensure the economic viability of both facilities, at least 20% of the recovered items should be sent to each plant. Finally the cost per unit of capacity to install in a depot is 67.5 €/year, regardless of their location.

6.1.2 Location – Allocation Results

As mentioned above, we were able to solve to optimality the facility location model (Phase I). Although the current recovery network of the CMS in Galicia consists of seven collection depots, which serve all the 707 stores where they collected, the solution obtained by solving the model (4.1.1) – (4.11) with CPLEX v11.1 gives rise to a new network that consists of just five depots (Fig. 6.1) whose characteristics are shown in Table 6.1.

Table 6.1 Facility location model results.

Depot	Location	Installed Capacity (items)	Number of items and (trips) to each plant per year		Number of stores served	Range of weekly demand collected (items)
			Plant 1	Plant 2		
1	Mos	378	5,300 (53)	7933 (80)	166	60 – 921
2	Santiago	326	11,395 (114)	-----	175	20 – 1,478
3	Oleiros	387	13,550 (136)	-----	128	31 – 1,059
4	Lugo	306	10,700 (107)	-----	136	4 – 1,147
5	Carballiño	293	4,100 (41)	6144 (62)	102	26 – 829

Depots 2, 3 and 4 send all collected items to recovery plant 1, while the remaining depots carry shipments to both plants. As a result, to the facilities of plant 1 arrive 45,045 end-of-life electrical appliances each year, representing 76.19% of the harvest in Galicia, whereas to plant 2 arrive 14,076 items corresponding to the remaining 23.81%. At the aggregate level depots provide relevant information relating to the number of served stores, for instance, depot 2 serves 175 stores, whose weekly demand varies from 20 to 1478 items.

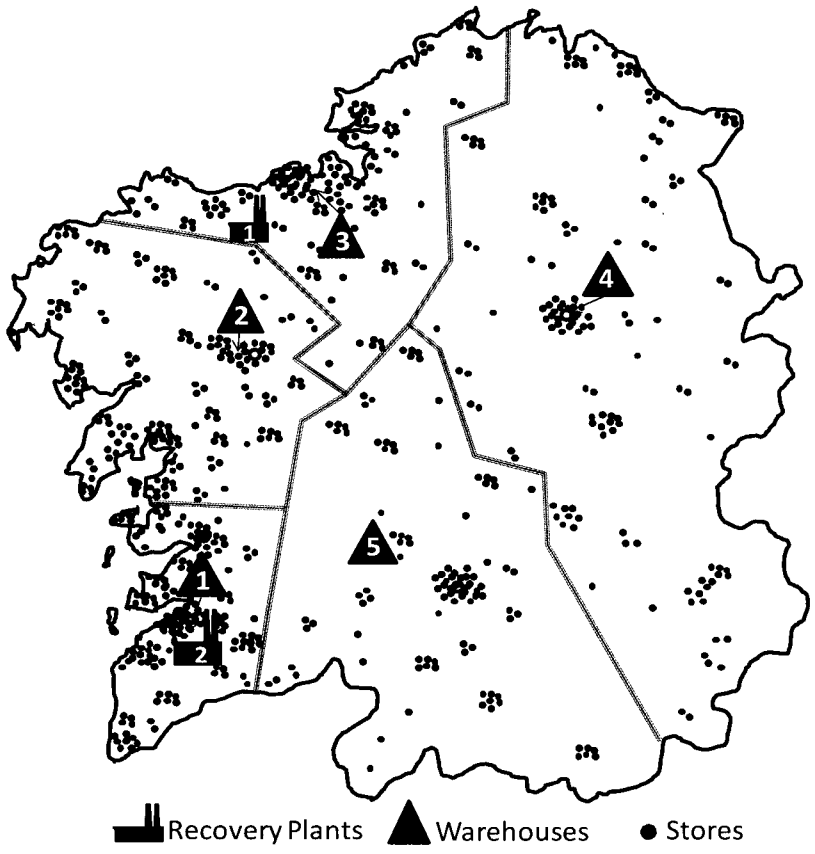


Figure 6.1 Optimal recovery network configuration for WEEE recovery in Galicia.

6.1.3 Validation of the Proposed Heuristic for the Collection Routing Problem

To assess the performance of the algorithm proposed in Section 5.1.1 for solving the routes collection problems in Galicia, and given that it is not possible to solve optimally the sized-problems handled here, a set of 30 random instances were generated –of up to 20 stores from the actual case study– for experimentation and analysis. The heuristic was coded in C using the Microsoft Visual C++ 6.0 programming environment under Windows Vista operating system. All solution procedures were implemented using an Acer AMD Athlon X2 PC with a processor operating at 1.90 GHz and 2.00 GB RAM. Results are summarized in Table 6.2, where the number of stores (n), the total amount of items collected ($\sum_{i=1}^{n} \omega_i$), and the set of vehicle types used (K) are listed for every instance.

In the comparison experiments we are interested in both optimal solutions and high quality feasible solutions. These are used to measure the *performance* of the heuristic algorithm in terms of its efficacy and efficiency. The best way to evaluate the *efficacy* of a heuristic algorithm is to compute the gap between *optimal* and heuristic solutions. However, given the complexity of our problem, optimal solutions are hard to find even for small instances. Thus, a time limit of 4 hours of CPU time was set on CPLEX v.11.1, taking the best solution found until that time. In Table 6.2 the costs obtained and solution times (in seconds) required by CPLEX, and by the heuristic algorithm are respectively reported. The best solution found is presented in bold characters, and when the optimal solution is reached, it is identified with (*). To measure the gap between both solutions the relative percentage deviation is computed as $((Z_{HEU} - Z_{CPLEX})/Z_{CPLEX})\times 100$, where Z_{CPLEX} and Z_{HEU} are the CPLEX and heuristic solutions respectively. Experimental results show a good performance of the proposed heuristic, on average the heuristic solution obtained is at 1.13% from the best solution found. In only one instance a deviation greater than 7% was obtained, however in 80% of the instances the deviation was lower than 2.50%, been possible to achieve the optimal value in 9 of the 30 instances generated (30%). In fact, for many instances our approach was able to reach better solutions in less than one second, than a commercial MIP solver after hours of computation.

Table 6.2 Comparative study of the heuristic performance, versus optimization model (with a time limit of 4 hours of CPU time). The best solution found is presented in bold characters, when the best solution found corresponds to the optimal solution, this is identified with (*). The relative percentage deviation is computed as: $((Z_{HEU} - Z_{CPLEX})/Z_{CPLEX}) \times 100$.

INSTANCE SIZE			CPLEX MODEL (4.12) – (4.20)				PROPOSED HEURISTIC (Section 5.1.1)		
N	$\sum \omega_i$	**K**	Z_{CPLEX}	Comp. Time (s)	Best Feasible	Abs. mipgap	Z_{HEU}	Comp. Time (s)	Percentage Deviation
10	110	{15, 25}	**72.48***	1.14	**72.48***	----	72.53	0.00	0.07
	112	{15, 25}	**74.19***	0.56	**74.19***	----	74.42	0.00	0.31
	100	{15, 25}	**75.36***	0.52	**75.36***	----	75.50	0.00	0.19
	104	{15, 25}	**65.49***	4.12	**65.49***	----	66.54	0.00	1.60
	110	{15, 25}	**122.84***	0.78	**122.84***	----	123.35	0.00	0.42
	115	{15, 25}	**158.88***	0.50	**158.88***	----	163.15	0.00	2.69
	94	{15, 25}	**55.30***	4.57	**55.30***	----	56.91	0.00	2.91
	75	{15, 25}	**43.17***	6.3	**43.17***	----	44.55	0.00	3.20
	124	{15, 20, 25}	**68.57***	0.83	**68.57***	----	69.73	0.00	1.69
	121	{15, 20, 25}	**97.52***	2.12	**97.52***	----	**97.52**	0.00	0.00
15	151	{15, 25}	**156.87***	958.29	**156.87**	40.96	159.63	0.00	1.76
	146	{15, 25}	**121.44***	77.73	**121.44**	18.87	**121.44**	0.00	0.00
	160	{15, 25}	**82.55***	97.96	**82.55**	10.77	**82.55**	0.00	0.00
	159	{15, 25}	**85.14***	127.05	**85.14**	14.72	85.46	0.00	0.38
	154	{15, 25}	**152.12***	15.03	**152.12***	----	156.05	0.00	2.58
	186	{15, 25}	**169.55***	3.61	**169.55***	----	**169.55**	0.00	0.00
	102	{15, 25}	**61.54***	3077.49	61.56	15.96	63.99	0.00	3.98
	135	{15, 20, 25}	**60.35***	4427.16	60.50	17.89	61.02	0.00	1.11
	178	{15, 20, 25}	**131.38***	4.77	**131.38***	----	131.57	0.00	0.14
	156	{15, 20, 25}	**121.10***	201.11	**121.10**	36.81	**121.10**	0.00	0.00
20	203	{15, 20, 25}	**174.91**	14400.00	181.72	113.00	177.49	0.00	1.48
	188	{15, 20, 25}	158.63	14400.00	166.19	88.81	**156.39**	0.00	-1.41
	197	{15, 25}	**131.39**	14400.00	135.39	49.07	**131.39**	0.00	0.00
	189	{15, 25}	**109.10**	14400.00	111.26	36.59	**109.10**	0.00	0.00
	212	{15, 25}	**195.02***	1680.25	195.10	27.52	**195.02**	0.00	0.00
	237	{15, 25}	**238.04***	579.80	**238.04**	41.10	**238.04**	0.00	0.00
	159	{15, 25}	**72.10**	14000.00	72.45	37.37	77.32	0.00	7.24
	193	{15, 25}	**87.00**	14400.00	89.91	44.01	**87.00**	0.00	0.00
	219	{15, 25}	**179.01**	14400.00	186.61	51.06	183.37	0.00	2.43
	220	{15, 25}	**211.65***	13973.58	**211.65**	82.46	214.03	0.00	1.12

On the other hand, the *efficiency* of the proposed algorithm is related to the CPU required by the heuristic to reach a solution. However, our heuristic algorithm usually requires a negligible computing time, which cannot be fairly compared against the time required by CPLEX to get a solution, even if the model's formulation is strengthen. To overcome this drawback, the default values for the algorithmic control directives of CPLEX were shifted to give more emphasis towards finding *high quality feasible solutions*. Under these settings the

solutions founded by CPLEX in a CPU time limit of 5 minutes can be fairly compared against the heuristic solutions, to evaluate the *effectiveness* of the proposed algorithm to reach *high quality feasible solutions* in negligible computing times. It is possible given that under these settings the MIP optimizer works hard to find high quality feasible solutions, rather than proving optimality, which let the optimizer quickly find feasible solutions that are otherwise very difficult to find. In Table 6.3 the best feasible solution found by CPLEX, and the absolute MIP *gaps* for model (4.12)–(4.20) are shown. When the optimality of the best feasible solution found by CPLEX is proved, it is identified with (*). It can be seen that high quality feasible solutions found by CPLEX in 5 minutes are optimal or very close to optimal solutions in instances with 10 and 15 customers; giving better bounds (1.14% better) than the heuristic solutions. Indeed in some instances the best feasible and optimal solutions are equal (see for example instance 11), however optimality has not been proved for the best feasible solution, so the corresponding MIP gap is reported. For instances with 20 customers the heuristic algorithm reached better solutions than the high quality feasible solutions found by CPLEX. This proves the *effectiveness* of the proposed algorithm. It was not possible to determine any negative trend in the algorithm performance, as a result it is expected a similar behavior in larger instances, such as those commonly found in real situations.

6.1.4 Collection Routing Results

To design the collection routes for the harvesting on the 707 collection points (stores) served every week, the collection data for the last 4 months of 2008 were used, given that during this period a significant upturn of WEEE was generated as a consequence of governmental policies that promote recycling, taking what could be considered a worst-case scenario. By applying the heuristic procedure all over the five depots 1,638 routes were generated totaling 161,060.50 km. Regarding the tour length characteristics, Table 6.3 shows that while the number of routes per depot has a range of variation of 79 tours, the average tour length between the corresponding areas remains almost constant. Nor it is possible to determine any relationship between the number of tours and the number of stores allocated to each depot. For example, while the number of stores allocated to depot 1 ranks it as the second place in descending order, the number of tours ranks it in the third place; however depot 5 is located in the fifth place in both categories. A similar situation is presented when assessing the annual travelled distance by the vehicles allotted to each depot in combination with the specific costs of transportation.

Table 6.3 Details of the tours in each depot, given by the proposed heuristic.

Depot	Location	Number of tours per year	Min. time in tours (hrs.)	Avg. time in tours (hrs.)	Max. time in tours (hrs.)	Total kilometres travelled (km/year)
1	Mos	312	1.08	2.58	5.14	24,581
2	Santiago	369	1.50	3.24	5.64	40,975
3	Oleiros	310	1.08	2.80	6.23	27,956
4	Lugo	357	1.16	3.12	6.84	37,922
5	Carballiño	290	1.10	3.04	5.57	29,628

Since for every period every depot requires a variable number of type 1 and 2 vehicles, variable transportation costs reflect this features. The annual travelled distance in each depot provides information on two parameters: the size of territory to be covered by the depot, and the frequency of visits to the stores. These results indicate the differences between the size of the regions and the density of stores. For example, depot 4 features an annual harvest of 10,700 items, against 13,233 items collected by depot 1 (23.67% more); however the land area covered is smaller. Therefore, in the latter the length of tours is very small. Due to demand variability, in all regions there are periods in which few vehicles are required, in contrast there are other periods that require more vehicles and more tours are conducted.

6.1.5 Simulation Results

After running the vehicle routing heuristic in a weekly basis and knowing the routes to be served, we could determine the maximum, minimum and mode regarding to the number of type 1 and 2 vehicles required for every depot (see Table 6.4). It should be noted that in general, the number of vehicles of each type most often required is the minimum one. For vehicle type 1 the number of trucks required ranges from 1 to 2, while for vehicle type 2 it ranges from 0 to 4 trucks. If the number of vehicles type k to allot to every depot were defined as $n_k = \max_t\{n'_k\}$ lower rates of vehicle use will result, especially if such amount of vehicles is required infrequently, in contrast, defined as $n_k = \min_t\{n'_k\}$ will lead to inefficient service levels.

Table 6.4 Number of vehicles of each type required by every depot, considering the weekly results of the vehicle routing heuristic.

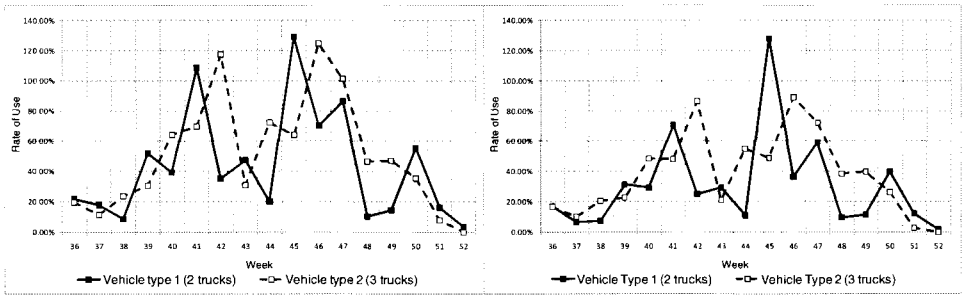
	Depot 1		Depot 2		Depot 3		Depot 4		Depot 5	
	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2
Min.	1	0	1	0	1	0	1	0	1	0
Mode	1	2	1	2	1	2	1	2	1	1
Max.	2	2	2	4	2	2	3	4	2	3

With the aim of finding the most appropriate values for these and other variables, different scenarios were analyzed by means of a simulation study, as detailed above. Two values for ξ were considered: 50% and 75% to generate a shipment order to the recovery plant, while for every depot being analyzed, we range n_k from its lower to its upper bound for each k. The CMS has defined as a target performance for vehicle use, utilization rates up to 50% on average. In addition, it seems reasonable to consider that in no more than three weeks the utilization rate achieves values over than 100% (and therefore extra vehicles should likely be needed), and never yields an utilization rate greater than 130%. Table 6.5 describes the scenarios for which the simulation results show the best system performance under the set of conditions stated above.

Table 6.5 Simulation study results for every depot, for determining the most appropriate number of vehicles of each type, under the predefined constraints.

ξ	Inventory Level		Vehicles		Usage Vehicle Rate		Weeks over 100%	Avg. Queue Length (trucks)	Avg. Time in Queue (min)
	Avg.	Max.	Type	Number	Avg.	Max.			
Depot 1									
75 %	216	316	1	1	62.94	108.88	2	0.3429	24.27
			2	2	51.43	90.91	0		
Depot 2									
50 %	108	229	1	2	59.31	92.22	0	0.7863	44.22
			2	3	56.13	123.28	2		
Depot 3									
50 %	142	243	1	2	57.64	117.18	2	0.6176	39.08
			2	3	52.14	90.98	0		
Depot 4									
75 %	174	279	1	2	55.47	129.06	2	0.7795	49.14
			2	3	50.91	124.52	3		
Depot 5									
75 %	160	252	1	2	56.12	72.51	0	0.4081	28.10
			2	2	50.75	129.56	2		

Concluding the simulation analysis, two dissimilar situations were compared to define bounds on the number of load/unload stations to install at every depot. In the first situation (scenario A) it is assumed that every depot has just one load/unload station, therefore every truck who has finished a tour needs to wait in a queue to be downloaded; in the opposite (scenario B, the most expensive) it is assumed that every truck is served independently in its own station, as a result once the truck has completed a tour it is immediately downloaded. Derived from Table 6.5 results, figure 6.2 presents a week by week comparison of the utilization rate for type 1 and 2 vehicles under both scenarios at depot 4. The value of this variable as well as the inventory policy are strategic factors that the decision-maker must adjust in every depot to improve the overall recovery system performance.



(a) One single station for all trucks (b) One station for each truck

Figure 6.2 Usage rate for each vehicle type at depot 4. Graph (a) show the usage rate for type 1 and 2 vehicles for the last 4 months in 2008, considering a single load/unload station serving to all trucks, while graph (b) shows the usage rate for the same type of vehicles and period, considering as much load/unload stations as trucks used.

6.1.6 Overall Comparison Regarding Current Configuration

Table 6.6 shows a detailed comparison of the current configuration against those obtained by the optimization models. The analysis on the results obtained leads to the major conclusion that the proposed network outperforms the one currently in use. Particularly, the annual costs of depot leasing turn out in a 12.5% improvement. Given that in the current configuration there are 2 more depots than in the proposed configuration, the size of the territory covered by each depot in the first of them is smaller. Therefore, the number of tours in the proposed configuration is 9.4% greater than the number of tours in the current configuration, however the current configuration makes a frequent use of vehicles type 2 (the most expensive). In

fact, in the current configuration the vehicle type 2 runs over 924 tours regarding the vehicle type 1, while in the proposed configuration the number of tours with both types of vehicles is almost uniform. Hence, even though the variable transportation cost in tours for the proposed configuration is 10.8% worst, this drawback was offset by an improvement of 29.2% in the fixed cost of vehicles in-use.

Table 6.6 Comparative results between current and proposed configurations. In parentheses the percentage of the improvement obtained by the proposed configuration regarding to the current one in use.

Characteristics	Current Configuration	Models (4.1.1) – (4.11) and (4.12) – (4.19) configuration		
		Value	Reduction	
Number of depots	7	5	2	(40.0%)
Annual cost (€/year) for depot in leasing	140,301.87	124,737.39	15,564.48	(12.5%)
Annual travelled distance (km) from depots to plants	69,975.06	64,195.84	5,779.22	(9.0%)
Annual number of harvesting tours in vehicles type 1	280	717	-437	(-60.9%)
Annual number of harvesting tours in vehicles type 2	1204	921	283	(30.7%)
Number of vehicles type 1	9	9	0	(0.0%)
Number of vehicles type 2	19	13	6	(46.2%)
Variable transportation cost (€/year) in tours	23,206.14	26,021.19	-2815.05	(-10.8%)
Fixed cost of vehicles in-use (€/year)	1,107,696.47	857,352.71	250,343.76	(29.2%)
ANNUAL TOTAL COST (€/year)	1,271,204.48	1,008,111.29	263,093.19	(26.1%)

6.2 Results and Analysis of the VRP-SLDW

In this section we present the experimental results obtained with an implementation of the GRASP algorithm for this vehicle routing problem. The heuristic was coded in C using the Microsoft Visual C++ 6.0 programming environment under Windows Vista operating system. All experiments were conducted on an Acer AMD Athlon X2 PC with a processor operating at 1.90 GHz and 2.00 GB RAM. The proposed algorithm must be capable to solve efficiently real-world sized instances of the problem described above. To test whether this is so, a sufficiently large number of instances is required for analyzing the performance of the algorithm. However, large-sized instances were not available in the literature published to date. Moreover, to the best of our knowledge, the VRP with date windows has not been

addressed before. To overcome this drawback, a random instance generator was designed to generate instances for which their *best known feasible solutions* were available, allowing us to compare our results.

Although this is not a common practice, this approach has been used in the literature when describing a new problem for which, besides the absence of instances against which to make a benchmark, there is not a mathematical formulation of the problem (see for example González and Adenso-Díaz 2006), or when although there is a mathematical formulation of the problem it is of such a great complexity that it is not possible to find bounds by a "brute force" approach using a MIP commercial software (see for example Taillard, 1993). In González and Adenso-Díaz (2006) the pseudo-optimal (or best known solution) of the instances for its disassembly sequence problem is obtained by disassembling the components in the reverse order of their assembly sequence, which is always available. Taillard (1993) test his proposed algorithms on a set of random generated problems for which he conjecture that the optimum solutions are known. For him the optimality has been conjectured because his algorithm has produced such type of solution for small instances for which he has not found any evidence for such solutions to not be optimum.

6.2.1 Instance Generator

Real-world instances are mainly characterized by situations where the date window and the collection capacity are both moderate and highly constrained. However, with the aim of studying the behavior of the algorithm under different situations, we have defined the random instances generator considering the following factors: F1 – *number of customers* {F_{11} – 50, F_{11} – 100, F_{13} –150}; F2 – *demand type* {F_{21} – balanced among all the customers, F_{22} – unbalanced}; F3 – *date windows tightness* { F_{31} – tightly, F_{32} – halfway, F_{33} – loosely}, F4 – *capacity tightness regarding demand* {F_{41} – collection capacity of 15% over the total customers demand, F_{42} – idem 50%, F_{43} –idem 85%}, F5 – *dispersion of customers grouped in clusters*, measured as the distance from the center of the cluster to the depot {F_{51} – 1/3, F_{52} – 1/15}. An instance assignment (or class) is defined by a combination of these five factors; for illustration the Fig. 6.3 shows an instance generated for the assignment ⟨2, 2, 1, 3, 1⟩. It should be noted that the number of classes obtained is (3)(2)(3)(3)(2) = 108.

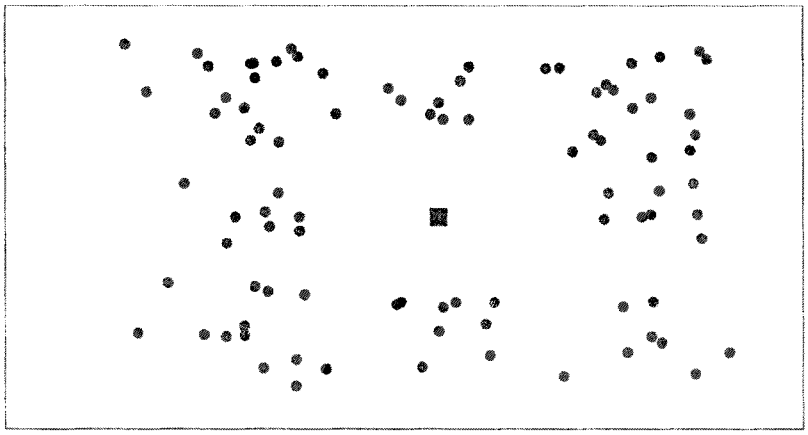


Figure 6.3 Random instance generated for the assignment ⟨2, 2, 1, 3, 1⟩, representing the location of 100 customers with unbalanced demand, a tightly date window and a load capacity of 85% over demand, with customers spread 1/3 of the distance between the center of the cluster to which they are assigned and the depot.

Due to the complexity of the problem, *optimal solutions* with the minimum routing costs are difficult to obtain. Therefore, the purpose of the instance generator is to create random instances for which *high quality feasible solutions* were available. These best known feasible solutions will be referred, from now on, as BK solutions. It should be noted that BK solutions are not obtained from the instance by applying some kind of heuristic procedure (i.e. BK solutions are not *heuristic solutions*). Instead, for each instance, we start by designing a solution and then we derive the instance data like the distances and travel times for that solution. Thus our GRASP algorithm, without a previous knowledge of the solution, will try to find the solution or even improve it. Below we describe the procedure to generate the random instances, and in Appendix B we illustrate the procedure by designing a trial example, and we show that the instance generator is able to create instances with BK solutions close to optimal.

Each instance contains the following data: the number of customers to be served, the number of vehicles, which were fixed at 4 (two of 15 units of capacity and two with capacity of 30 units). Each one of these vehicles can complete at most 3 tours. Thus, the total collection capacity of the systems is computed as: $\sum_{k \in K} Q_k \cdot T \cdot R = 1{,}620$. The customer's demand is computed and generated considering factors F1, F2 and F4. The next step consists on forming the LOAD matrix, which is generated by taking the customer with the largest demand pending to allocate and inserting the largest portion of such demand to the first route that more load is able to receive, the data structure is then updated, and the procedure is

repeated until the total demand of all customers have been allocated. To ensure full truckload routes, the customer's demand is complemented in each route that requires it. To define the date windows, for each customer the procedure identifies the first and last day in which it is visited; then under the factor F3 consideration it determines the width of the date windows. The size of the area to locate customers ranges from (-50 to 50) for x and y, the depot is located at (0, 0). To ensure *high quality feasible routes*, to be used as benchmark, all customers assigned to the same route are grouped into clusters, the coordinates of the clusters are calculated, and its amplitude is defined by the factor F5.

It should be noted that for instances with spread customers this solution could not be close to a high quality solution, however this solution is always available, and therefore it can be used as a reasonable reference to measure the efficacy of the algorithm.

6.2.2 Analysis and Discussion of Computational Results

The experimental framework consists of four different experimental situations used to analyze the performance of the proposed algorithm and to gain insights into the implementation issues. To this end, the solution found by GRASP was compared against the BK solution for each instance in its corresponding experiment. To measure the difference between both solutions the percentage deviation (*DEV*) is computed as $\frac{Z_{GRASP}-Z_{BK}}{Z_{BK}} \times 100$, where Z_{GRASP} and Z_{BK} are the GRASP and BK solutions respectively. The remainder of this section describes the design of every experiment and discuses its results.

6.2.2.1 Experiment A – Behavior of the Quality Parameter α

This experiment is focused on the appropriate choice of the quality parameter α in the construction phase, which regulates the size of the Restricted Candidate List. The purpose of this experiment is to evaluate the algorithmic performance as a function of the value of α. For evaluating the performance of the algorithm in terms of α, we compare the results obtained with the different values of $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, on a subset of nine problem classes. A summary of the results is displayed in Table 6.7. For each problem class the average percent deviation from BK, the worst percent deviation from BK, and the percentage of infeasible solutions in the construction phase are shown. The average is computed over 5

instances from each class. The class of problems selected to perform the analysis, correspond to instances with 100 customers, which are spread 1/3 of the distance between the center of the cluster to which they are assigned and the depot, and with unbalanced demand. The variation in the date windows and load capacity characterize each problem class. The choice of problem class is justified because we want to evaluate the algorithmic performance in problems with both date windows and load capacity loosely constrained and harder constrained, to resemble the situations currently observed in real-world applications. In addition, performing an Analysis of Variance test it can be demonstrated that factors like the date windows and load capacity affects the quality of the obtained solutions.

Table 6.7 Evaluation of quality parameter α. For each problem class it shows: (A) the average percent deviation from BK, (B) the worst percent deviation from BK, and (C) the percent of infeasible solutions in the construction phase.

Problem Class		$\alpha = 0.1$	$\alpha = 0.3$	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.9$
DS1 – ⟨2, 2,1,1,1⟩	(A)	4.69	5.83	-2.72	4.93	4.20
	(B)	7.88	10.69	5.97	10.25	9.79
	(C)	12.80	11.60	13.60	12.40	13.20
DS2 – ⟨2, 2,1,2,1⟩	(A)	3.09	-1.87	0.56	1.12	-4.86
	(B)	4.77	2.47	2.41	3.10	-0.11
	(C)	0.00	0.00	10.23	14.00	67.20
DS3 – ⟨2, 2,1,3,1⟩	(A)	-1.07	-0.27	-0.38	-0.10	-1.93
	(B)	1.31	0.77	0.25	1.17	1.16
	(C)	0.00	0.00	0.00	0.00	0.00
DS4 – ⟨2, 2,2,1,1⟩	(A)	2.32	3.82	3.24	3.74	2.28
	(B)	4.73	6.43	5.10	5.87	5.15
	(C)	0.00	0.00	0.00	0.00	0.00
DS5 – ⟨2, 2,2,2,1⟩	(A)	4.32	5.54	6.19	9.68	7.57
	(B)	16.33	16.56	8.55	17.57	17.58
	(C)	0.00	0.00	0.00	0.00	0.00
DS6 – ⟨2, 2,2,3,1⟩	(A)	0.16	2.23	6.05	8.74	11.68
	(B)	6.10	8.87	9.39	13.71	16.36
	(C)	0.00	0.00	0.00	0.00	0.00
DS7 – ⟨2, 2,3,1,1⟩	(A)	30.69	37.42	33.46	28.91	30.82
	(B)	47.42	44.32	50.88	48.91	47.26
	(C)	0.00	0.00	0.00	0.00	0.00
DS8 – ⟨2, 2,3,2,1⟩	(A)	34.38	30.07	37.34	39.44	37.10
	(B)	47.79	42.96	43.61	45.06	46.45
	(C)	0.00	0.00	0.00	0.00	0.00
DS9 – ⟨2, 2,3,3,1⟩	(A)	25.84	29.42	32.43	35.66	40.85
	(B)	41.55	43.57	34.78	43.77	46.08
	(C)	0.00	0.00	0.00	0.00	0.00

As can be seen, for the instances in the DS1, DS2, and DS3 class (characterized by hardly constraint date windows), the best results were obtained with $\alpha = 0.5$, whereas for instances

in DS4, DS5, and DS6 class, $\alpha = 0.1$ gave the best results. Possibly, $\alpha = 0.4$ is a better compromise in this sense. For the instances in the DS7, DS8, and DS9, $\alpha = 0.1$ gave the best results, even if the worst relative gap is the largest one. In any case, the obtained results indicate that as the instances become more tightly constrained and, thus, more difficult to solve, medium to low values of α can be preferred. From the analysis of the table it can be noticed that the percent deviation clearly increases as higher α values are used, meaning that in terms of solution quality the variants which favor greedy implementations are to be preferred to random implementations. Moreover, high values of α require a significantly larger computational time, which is especially spent by the local search phase to reach local optima from highly random solutions.

6.2.2.2 Experiment B – Behavior of the Local Search Procedure

The aim of this experiment was to investigate the behavior of the local search procedures. To this end we set the GRASP parameters *iters* = 100 and $\alpha = 0.4$, and we compare the results obtained on the same set of instances used in Experiment A. A summary of the results is displayed in Table 6.8. For each problem class the average percent deviation obtained after each phase of our GRASP algorithm in comparison with the *best solution known* (BEST) is shown. Such a comparison permits us to evaluate the importance of each phase of our GRASP algorithm, including the local search procedures, in the overall quality of the solutions generated. The time ratio used for each phase in comparison with the total solution time are also shown.

As can be seen, the results in Table 6.8 indicate that the local search indeed provide with considerably improved solutions. For instance, for the DS2 class the average percent deviation from BEST obtained by the greedy randomized construction phase is 25.18%, the local search phase permits an additional improvement, reaching a final solution deviated 0.78% on average from BEST. The greedy randomized construction phase takes the 19.11% of the total time, while the local search takes the remaining 80.89%. Most of the solutions obtained after the greedy randomized phase are feasible (i.e. unpenalized solutions), but are often far from the best known solutions (34.91% on average and up to 62.50% in some instances). The local search phase improves these results significantly (32.08% of improvement in average). Even, the instances with poor results (DS7, DS8, and DS9, which correspond to classical VRP instances) show an improved, though still poor results (with a final solution above 25% form BEST) following the local search phase.

Table 6.8 Evaluation of local search. For each problem class it shows: the average and the worst percent deviation from BEST and the average ratio time required by each phase of our algorithm.

Problem Class		Construction Phase		Local Search Phase	
		Deviation from BK (%)	Ratio Time (%)	Deviation from BK (%)	Ration Time (%)
DS1 – ⟨2, 2,1,1,1⟩	Avg.	23.06	17.43	2.92	82.57
	Worst	28.12		4.59	
DS2 – ⟨2, 2,1,2,1⟩	Avg.	25.18	19.11	0.78	80.89
	Worst	32.90		2.00	
DS3 – ⟨2, 2,1,3,1⟩	Avg.	28.74	10.12	0.00	89.88
	Worst	35.76		0.00	
DS4 – ⟨2, 2,2,1,1⟩	Avg.	34.93	17.19	4.33	82.81
	Worst	39.84		7.87	
DS5 – ⟨2, 2,2,2,1⟩	Avg.	29.04	18.37	4.86	81.63
	Worst	39.42		7.83	
DS6 – ⟨2, 2,2,3,1⟩	Avg.	30.48	14.45	4.47	85.55
	Worst	38.98		6.88	
DS7 – ⟨2, 2,3,1,1⟩	Avg.	47.61	14.56	28.90	83.44
	Worst	62.50		39.79	
DS8 – ⟨2, 2,3,2,1⟩	Avg.	50.36	16.70	39.37	83.3
	Worst	59.22		42.13	
DS9 – ⟨2, 2,3,3,1⟩	Avg.	44.82	16.93	33.07	83.07
	Worst	51.40		42.07	

In some GRASP solutions, infeasible solutions were generated, but even in such cases the local search phase was able to reduce infeasibilities. In general, initial solutions were improved by 23.71%. Especially relevant are the phase I local search moves which were able to reduce infeasibilities in all cases. Moreover, Phase I moves were required 20% of the times reaching an average improve of 68.91% over the initial solution. Regarding the phase II moves, the m_{I2} and m_{E2} moves were applied 99.05% and 93.64% of the times respectively, while m_C was applied 65.77% of the times. These improvements have a cost, the local search phase uses the greater part of the CPU time (more than 80%).

6.2.2.3 Experiment C – Solution Spectrum of Our Complete GRASP Procedure

The aim of this experiment was to analyze the influence of different factors on the quality of the solution obtained and on the efficiency of the algorithm. To this end, an experimental design was established to include the five factors (from F1 to F5) used in the generation of the random instances. We set the GRASP parameters *iters* = 100 and $\alpha = 0.4$. In order to gain insight into the performance of the proposed algorithm, the percentage deviation was

computed for each instance. The number of observations obtained for the dependent variable DEV is 540, which corresponds to five replicates for each one of the 108 problem classes.

Experimental results show a good performance of the proposed algorithm: on the average the metaheuristic solution obtained is at 4.32% from the BK solution in an average computing time of 113.68 seconds. In fact, for many instances (225) our approach was able to reach better solutions than the BK solution. The average DEV for each combination of factors ranges from -7.49% to 40.37%. This situation tells us that our algorithm performs better in some combination of factors than in others. Therefore, the results were analyzed using a 5-way ANOVA with repeated measures, where the five factors were considered to have fixed effects and are the object of the analysis.

Table 6.9 shows the inter-subject effects corresponding to the dependent variable DEV. The last column gives the significance level of each factor (respectively two-way interaction), which measures the probability of occurrence of the observed averages of the dependent variable for the different levels of the factor (respectively combination of levels of two factors) if they all had the same mean. Thus, if the significance level of a certain factor is below the significance level 0.05, this means that we can consider that factor as statistically significant, thus having influence on the results of our algorithm.

Table 6.9 ANOVA table for the quality of the solution *DEV* ($R^2 = 0.827$).

Source	D. of Freedom	Sum of Squares	Mean Square	F	Significance
F1	2	164386	81743	332.97	0.000
F2	1	438	438	1.78	0.182
F3	2	5179	2590	10.55	0.000
F4	2	354969	177485	722.96	0.000
F5	1	17712	17712	72.15	0.000
F1*F2	2	1905	953	3.88	0.021
F1*F3	4	4317	1079	4.40	0.002
F1*F4	4	21203	5301	21.59	0.000
F1*F5	2	3086	1543	6.29	0.002
F2*F3	2	1564	782	3.18	0.042
F2*F4	2	1345	673	2.74	0.066
F2*F5	1	40	40	0.16	0.686
F3*F4	4	13356	3339	13.60	0.000
F3*F5	2	431	216	0.88	0.416
F4*F5	2	5706	2853	11.62	0.000
Error	506	124221	245		
Total	539	718960			

Analyzing the factorial design with the ANOVA, it can be seen that factors F1, F3, F4 and F5 are statistically significant for DEV. It cannot be demonstrated that a factor like the

demand type (F2) affects the quality of the obtained solutions. Since factors F3 and F4 are obviously significant (because each combination level corresponds to a different instance), this result implies that the problem class makes a difference. All two-way interactions are significant at the 0.05 level, except some that involve factor F5. The significance level of the two-way interaction between F2 and F5 is larger than 0.60. This implies that there are not significant interactions between the demand type and the dispersion of customers, as also does the dispersion of customers and the date windows tightness. It should be noted that the model fit (R^2= 0.827) indicates that the five factors and their two-way interactions explain much of the variance of the observed results.

A depth analysis reveals that the algorithm performs better in instances where the date window and the collection capacity are tighter constrained; on the other hand worst results are obtained in those instances where the collection capacity is larger and customer demand is shorter. This is because the algorithm was designed to deal with the first group of instances, while instances easier to solve with wide date windows, large collection capacity and customers with low demand do not require to be split; in fact such instances resemble classical VRP problems, wherein inter and intra routes moves are required to improve heuristic solutions.

6.2.2.4 Experiment D – Case of Study

In this part of the work, we perform a comparison of the proposed approach with current practice. To this end we consider the particular situation of a national WEEE Collective Management System operating in Galicia (northwestern of Spain); who in 2008, collected more than 3,239 tons of WEEE from 707 geographically dispersed stores via 5 depots. Data related to the fleet of vehicles used in the harvesting process, distances and travel times, as well as the real amount of items collected along 2008 in every store were provided by the collective management system. In general two types of vehicles with different load capacity (15 and 25 items) are used in the harvesting process. The first (hereafter referred to as vehicle type 1) are smaller than the corresponding vehicles with a load capacity of 25 items (hereinafter referred to as vehicle type 2), therefore they can access to all stores, although the latter have a loading platform that facilitates the collection task. Annual fixed costs are respectively 34,993.47 and 41,723.96 €/year, while the variable costs reach 0.15 and 0.17 €/km. To design the collection routes, for each depot we select the week with the higher demand collected within the last quarter of 2008 (see Table 6.10), given that during this

period a significant upturn in WEEE generation was generated due to governmental policies to promote recycling, taking what could be considered a worst-case scenario. It should be noted that each store has its own demand, and its corresponding date window. The GRASP parameters were set on *iters* = 100 and $\alpha = 0.5$.

Table 6.10 Input data do the case of study.

Depot	Installed Capacity (items)	Number of stores served	Range of items collected by store
1	378	166	5 – 159
2	326	175	5 – 228
3	387	128	3 – 168
4	306	136	2 – 529
5	293	102	4 – 77

By applying the heuristic procedure all over the five depots 229 routes were generated totaling 15,589.62 km. Regarding the tour length characteristics, Table 6.11 shows that except for depot 2 the number of routes per depot and the number of kilometers traveled between the corresponding areas remain almost constant. The travelled distance in each depot provides information on two parameters: the size of territory to be covered by the depot, and the number of stores; characterizing the differences between the five regions.

Table 6.11 Routing details in each depot, given by the proposed algorithm

Depot	Number of tours	Number of trucks		Kilometers traveled
		Type 1	Type 2	
1	39	1	2	1,677.99
2	63	1	5	4,711.45
3	44	1	2	2,824.23
4	48	1	2	3,142.94
5	35	1	2	3,233.01

To perform the comparison of the proposed approach with current practice, the field data were analyzed to estimate the transportation cost of the current delivery method from stores to the corresponding regional depots. The datasets employed contained only the number of items collected each day from each store for the corresponding regional depot, and they did not specify the types of trucks or the number of trucks that were utilized. Additionally, the records did not include specific cost information. Hence, some assumptions to estimate the cost were made, reflecting the current standard for routing and scheduling WEEE collection activities. From the records, the number of items was transformed to the type and number of trucks that might be utilized. We assume that type 1 trucks were used, and based on the

analysis of current practices, this assumption appears valid. We assume that whenever possible, a full truck load was used, and we also attempted to minimize the number of trucks used. In other words, routes are optimized based on the known supply of end-of-life electronic items from each store. After the number of tours was estimated we calculated the distances of the entire tours by multiplying the two-way distance from each store to the corresponding regional depot. To compare the performance of the two methods more directly, we estimated the transportation cost of each method. The results are presented in Table 6.12, along with the results of the proposed algorithm.

Table 6.12 Comparison between the current method and the GRASP algorithm.

	Current operation				Proposed operation	
Depot	No. of tours	Km. travelled	No. of trucks	Cost (€/week)	Cost (€/week)	% of cost reduction
1	73	3,011	4	3,143.31	2,561.60	18.51%
2	118	8,638	5	4,660.41	3,875.47	16.84%
3	84	5,092	4	3,455.55	2,748.13	20.47%
4	86	5,330	4	3,491.34	2,784.25	20.25%
5	66	6,210	3	2,950.37	2,840.05	3.74%

Even though one cannot directly compare the distances of the two methods because of the different types of computation used, distance was reduced between 1,333 and 3,927 km with the proposed method, and the number of tours was reduced between 31 and 55. The transportation cost of the five depots fells an average of 15.96% from the current levels per week, with the largest reduction at depot 3, which had the largest installed capacity, and the smallest reduction occurred at depot 5, which had the smaller territory. Results indicate that both the appropriate use of vehicles and a suitable partition loads is a determinant factor in the performance of any WEEE collection system.

6.3 Results and Analysis for the DCFP and its Variations

In this section we present the experimental results obtained with an implementation of the proposed algorithms for the disassembly cell formation problem (DCFP) and its variants (Robust: DCFP-R, and Reconfigurable: DCFP-D. The heuristics were coded in C using the Microsoft Visual C++ 6.0 programming environment under Windows Vista operating system. All experiments were conducted on an Acer AMD Athlon X2 PC with a processor operating at 1.90 GHz and 2.00 GB RAM.

6.3.1 Analysis of the TS Algorithm for the DCFP

As previously stated the purpose of this experiment is to analyze the behavior of the swap, insert, union and split neighborhoods, which have been previously used in manufacturing cell formation problems, in a real-world instance of the DCFP (taken from Adenso-Díaz *et al.* 2006) implementing them under a metaheuristic scheme like Tabu Search, to further formalize and expand them for the robust and reconfigurable disassembly cell formation problems.

6.3.1.1 Problem Data

There are many electrical and electronic products, which according to a European Commission Directive must be recycled. Disassembly is required for all those products and the following assumptions hold: a) a large number of these products are small-sized products (electric toothbrushes, clocks, etc.) whose automated disassembly is hard to justify economically; b) in other cases, the products are large appliances (tobacco vending machines, for instance), but the number of units is not large enough to justify the creation of a treatment facility at a regional level. Products chosen are both large in volume and weight as well as having a substantial number of units that must be disposed of annually. For each product to disassemble, its annual demand (units available for recycling) and its unit material handling cost in the shop (which depends basically on their weight) are known. Each disassembly operation that must be carried out in the facility, as well as the machinery type it requires is known. Finally, we know the annual depreciation cost and capacity of each machine.

In the given instance 14 products, 16 operations and 6 machine types were considered. The products were classified into computer products (CPUs, CRTs and TV sets), cold appliances (refrigerators, combis and freezers), appliances for washing clothes (washing machines and dryers), kitchen appliances (ovens, microwave ovens, ceramic hobs, cookers, hobs and dish washers), and appliances used in the bathroom (water heaters). Table 6.13 lists the products to disassemble, with their annual demands (units available for recycling) and unit material handling costs in the shop (which depends basically on their weight).

Table 6.13 Annual demands and unit material handling costs (€). Taken form from Adenso-Díaz *et al.* 2006.

Product type	Demand	Handling cost
TV set	24,099	1.13
CPU	9,468	0.56
Refrigerator	14,630	2.63
Combi	9,754	3.00
Freezer	5,630	2.25
Washing machine	32,348	1.88
Dryer	2,815	1.69
Dish washer	6,204	1.50
Oven	9,286	1.69
Microwave oven	3,980	0.19
Ceramic hob	3,159	0.56
Cooker	13,538	0.94
Hob	5,867	0.38
Water heater	963	0.30

6.3.1.2 Results and Analysis

Table 6.14 and Table 6.15 show the results of the optimal solution found by CPLEX for the instance. In this particular case, with 14 products to process on 6 machines, the model is composed of 2,400 variables (384 of which are integer), and 1,132 constraints, the CPU time needed being just 44.53 seconds for the exact optimal solution. The solution found by CPLEX considers the use of two cells. Cell C1 carries out all the operations required by products 1, 3, 4, 5, 6, 7, 9 and 12, while the remaining products perform almost all their operations in cell C2, except those assigned to cell C1. The resulting intercellular transportation cost, associated to intercellular movements is 19,595.94 €. It can be seen that the cheaper machine/tools (specifically M3, M4 and M5) are the only ones that appear duplicated in both cells, thus satisfying the capacity constraints. The total depreciation cost of purchasing all the required machine/tools comes to 68,695.00 €, which, added to the

intercellular transport costs, comes to a total of 88,290.94 € per year, a sum which is the objective function value of the optimal solution.

Table 6.14 Configuration (assignment of operations to cells) within the optimal solution.

	1st Op	2nd Op	3er Op	4th Op	5th Op	6th Op	7th Op	8th Op
Product 1	C1	C1	C1	C1	C1	C1	--	--
Product 2	**C2**	**C2**	**C2**	**C2**	**C2**	C1	--	--
Product 3	C1	C1	C1	C1	C1	C1	C1	--
Product 4	C1	C1	C1	C1	C1	C1	C1	--
Product 5	C1	C1	C1	C1	C1	C1	C1	--
Product 6	C1	C1	C1	C1	C1	C1	C1	C1
Product 7	C1	C1	C1	C1	C1	--	--	--
Product 8	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 9	C1	C1	C1	C1	C1	C1	C1	--
Product 10	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 11	**C2**	**C2**	**C2**	**C2**	C1	--	--	--
Product 12	C1	C1	C1	C1	C1	C1	C1	--
Product 13	**C2**	**C2**	C1	--	--	--	--	--
Product 14	**C2**	**C2**	**C2**	**C2**	C1	--	--	--

Table 6.15 Number of machines and their distribution to cells within the optimal solution.

	Acquisition cost	Cell 1	Cell 2	Total acquisition cost
M1	52,000.00 €	1		52,000.00 €
M2	2,400.00 €	1		2,400.00 €
M3	50.00 €	3	1	200.00 €
M4	45.00 €	4	1	225.00 €
M5	100.00 €	2	1	300.00 €
M6	13,570.00 €	1		13,570.00 €
				68,695.00 €

The solution obtained by the Greedy Procedure is shown in Table 6.16 and Table 6.17. The transportation costs associated to intercellular movement reach 187,642.49 €, while the total depreciation cost of purchasing all the required machine/tools comes to 71,340.00 €. Thus, the total cellular configuration costs reach 258,982.49 € per year.

Table 6.16 Assignments of operations to cells in the greedy solution.

	1st Op	2nd Op	3er Op	4th Op	5th Op	6th Op	7th Op	8th Op
Product 1	C4	C4	C4	C1	C4	C4	--	--
Product 2	C3	C3	C3	C3	C3	C1	--	--
Product 3	C4	C4	C4	C4	C4	C4	C1	--
Product 4	C4	C4	C4	C4	C4	C4	C1	--
Product 5	C3	C3	C3	C3	C3	C3	C1	--
Product 6	C1	C1	C1	C1	C1	C1	C1	C1
Product 7	C2	C2	C2	C2	C1	--	--	--
Product 8	C3	C3	C3	C3	C3	C3	C3	C1
Product 9	C4	C4	C4	C4	C4	C4	C1	--
Product 10	C2	C2	C2	C2	C2	C2	C2	C1
Product 11	C2	C2	C2	C2	C1	--	--	--
Product 12	C3	C3	C3	C3	C3	C3	C1	--
Product 13	C2	C2	C1	--	--	--	--	--
Product 14	C2	C2	C2	C2	C1	--	--	--

Table 6.17 Number of machines and their distribution to cells within the greedy solution.

	Acquisition cost	Cell 1	Cell 2	Cell 3	Cell 4	Total acquisition cost
M1	52,000.00 €	1	0	0	0	52,000.00 €
M2	2,400.00 €	0	0	1	1	4,800.00 €
M3	50.00 €	2	1	1	2	300.00 €
M4	45.00 €	2	1	1	2	270.00 €
M5	100.00 €	1	1	1	1	400.00 €
M6	13,570.00 €	0	0	0	1	13,570.00 €
						71,340.00 €

The TS procedure obtains a solution, shown in Table 6.18 and Table 6.19, which also considers the use of two cells. Intercellular transportation costs, associated to intercellular movements shown in red in the table are 22,179.57 €. The assignment of machines to cells is equal to the one obtained by the optimal solution. The total depreciation cost of purchasing all the required machine/tools comes to 68,695.00 €, which, added to the intercellular transport costs, comes to a total of 90,874.57 € per year.

Table 6.18 Configuration within the TS solution.

	1st Op	2nd Op	3er Op	4th Op	5th Op	6th Op	7th Op	8th Op
Product 1	C1	C1	C1	C1	C1	C1	--	--
Product 2	**C2**	**C2**	**C2**	**C2**	**C2**	C1	--	--
Product 3	C1	C1	C1	C1	C1	C1	C1	--
Product 4	C1	C1	C1	C1	C1	C1	C1	--
Product 5	C1	C1	C1	C1	C1	C1	C1	--
Product 6	C1	C1	C1	C1	C1	C1	C1	C1
Product 7	**C2**	**C2**	**C2**	**C2**	C1	--	--	--
Product 8	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 9	C1	C1	C1	C1	C1	C1	C1	--
Product 10	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 11	**C2**	**C2**	**C2**	**C2**	C1	--	--	--
Product 12	C1	C1	C1	C1	C1	C1	C1	--
Product 13	C1	C1	C1	--	--	--	--	--
Product 14	**C2**	**C2**	**C2**	**C2**	C1	--	--	--

Table 6.19 Number of machines and their distribution to cells within the TS solution.

	Acquisition cost	Cell 1	Cell 2	Total acquisition cost
M1	52,000.00 €	1		52,000.00 €
M2	2,400.00 €	1		2,400.00 €
M3	50.00 €	3	1	200.00 €
M4	45.00 €	4	1	225.00 €
M5	100.00 €	2	1	300.00 €
M6	13,570.00 €	1		13,570.00 €
				68,695.00 €

Table 6.20 summarizes the performance results for the proposed heuristic procedures. The table shows the time in seconds required to solve the problem and the percentage deviation from the optimal solution for each procedure on this test problem. TS performed significantly better than the greedy solution procedure. Thus, TS deviated 2.93% from the optimal solution and took less than 3.12 seconds, while the greedy solution deviated 193.32% from the optimal solution and required negligible computing time.

Table 6.20 Comparative evaluation of proposed heuristic procedures.

CPU Times (seconds)			Deviation (%)	
Greedy	TS	CPLEX	Greedy	TS
0.00	3.12	44.53	193.32 %	2.93 %

The results showed that the greedy heuristic procedure proposed although quite efficient provides a poor quality solution. However, the Tabu Search procedure generates a relative good solution in 92.7% less time compared to CPLEX.

6.3.1.3 Comments on the Algorithm Performance

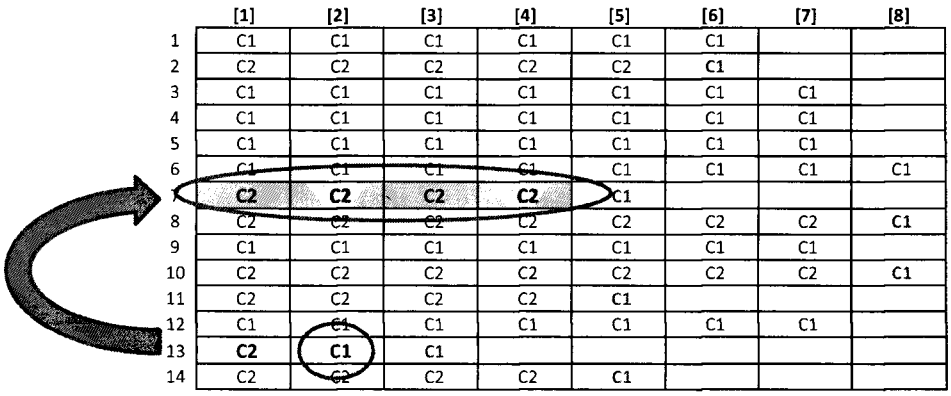
When comparing the optimal and the resulting TS configurations we observe that both are very similar except for the allocation of operations for products 7 and 13. In fact, the number and allocation of machines are identical. Therefore, we performed a deeper analysis to discover why the optimal solution was not reached.

We use the same greedy procedure but we force it to form exactly two cells as in the optimal solution. Results are shown in Table 6.21. Only the assignment of operations to cells is shown since the number and allocation of machines is the same than in the optimal solution. The transportation cost associated to intercellular movement of products is 24,353.29 €, which added to the machine cost comes to a total of 93,048.29 € per year. Note that in this case, the greedy solution deviates just 5.39% from the optimal solution in a negligible computing time. The only difference is in the assignment of the 1st to 4th operations from product 7 as regards to the optimal solution.

Table 6.21 Configuration within the greedy solution starting with C_{min} cells. Total cost 93,048.2935€: Machines acquisition 68,695.00€ plus Intercellular movement 24,353.2935€.

	1st Op	2nd Op	3er Op	4th Op	5th Op	6th Op	7th Op	8th Op
Product 1	C1	C1	C1	C1	C1	C1	--	--
Product 2	**C2**	**C2**	**C2**	**C2**	**C2**	C1	--	--
Product 3	C1	C1	C1	C1	C1	C1	C1	--
Product 4	C1	C1	C1	C1	C1	C1	C1	--
Product 5	C1	C1	C1	C1	C1	C1	C1	--
Product 6	C1	C1	C1	C1	C1	C1	C1	C1
Product 7	**C2**	**C2**	**C2**	**C2**	**C1**	--	--	--
Product 8	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 9	C1	C1	C1	C1	C1	C1	C1	--
Product 10	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	**C2**	C1
Product 11	**C2**	**C2**	**C2**	**C2**	C1	--	--	--
Product 12	C1	C1	C1	C1	C1	C1	C1	--
Product 13	**C2**	**C2**	C1	--	--	--	--	--
Product 14	**C2**	**C2**	**C2**	**C2**	C1	--	--	--

To improve the initial solution the TS procedure was applied. It was expected that the TS would identify the move from the 1st to the 4th operations of product 7 from cell two to cell one to reach the optimal solution. However, the best move is to insert the 2nd operation of product 13 into cell one (see Fig. 6.4), thus the allocation of machines remains the same. Such move implies an improvement in the objective function of –501.6285.



	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
1	C1	C1	C1	C1	C1	C1		
2	C2	C2	C2	C2	C2	C1		
3	C1	C1	C1	C1	C1	C1	C1	
4	C1	C1	C1	C1	C1	C1	C1	
5	C1	C1	C1	C1	C1	C1	C1	
6	C1	C1	C1	C1	C1	C1	C1	C1
7	**C2**	**C2**	**C2**	**C2**	**C1**			
8	C2	C2	C2	C2	C2	C2	C2	C1
9	C1	C1	C1	C1	C1	C1	C1	
10	C2	C2	C2	C2	C2	C2	C2	C1
11	C2	C2	C2	C2	C1			
12	C1	C1	C1	C1	C1	C1	C1	
13	**C2**	**C1**	C1					
14	C2	C2	C2	C2	C1			

Figure 6.4 Improved solution after the first move. Total cost 92,546.6650€: Machines acquisition 68,695.00€ plus Intercellular movement 23,851.6650€.

This is the best move as detailed below:

- Originally, the only inter-cell transfers are carried out if the product 13 goes from the 1st to the 3rd operation, which occurs with a probability of 10%, or if it goes from the 2nd to the 3rd operation, which occurs with a probability of 100%, which implies a intercellular movement cost 2,173.72 for product 13. Note that the first operation of product 13 is required just 75% of the times.
- In the greedy solution the intercellular transfer cost of product 13 comes to 2,173.72€, which is the sum of two events. The first happens when the product 13 passes from the 1st operation to the 3rd operation, which occurs with a probability of 10%. The second happens when the product 13 passes from the 2nd operation to the 3rd operation, which occurs with a probability of 100%. It should be noted that the 1st operation of product 13 is requires just 75% of the times (see Table 6.22).

Table 6.22 Computation of the intercellular transport cost of product 13 within the greedy solution.

Product	j	j'	D_p	α_p	f_{pj}	$\pi_{pjj'}$	$D_p \times \alpha_p$	Total
13	1	3	5867	0.38	0.75	0.10	2,229.46	167.21
13	3	3	5867	0.38	0.90	1.00	2,229.46	2,006.51
								2,173.72

- However, if the 2nd operation switches from cell 2 to cell 1, only the intercellular transfers will be given if the product 13 moves from the 1st to the 2nd operation, which occurs with a probability of 90%, or if it moves directly from 1st to the 3rd operation, which occurs with a probability of 10%. This implies an intercellular movement cost of 1,672.10€ for product 13 (see Table 6.23).

Table 6.23 Computation of the intercellular transport cost of product 13 after the first move of TS.

Product	J	j'	D_p	α_p	f_{pj}	$\pi_{pjj'}$	$D_p \times \alpha_p$	Total
13	1	3	5867	0.38	0.75	0.10	2,229.46	167.21
13	1	2	5867	0.38	0.75	0.90	2,229.46	1,504.89
								1,672.10

- The difference between both solutions results in a saving of 501.63 €.

On their own the move of any of the first four operations of product 7 results in the following changes:

Oper = 1: from cell 2 - to cell 1, delta 4281.615000 (increase)
Oper = 2: from cell 2 - to cell 1, delta 9038.965000 (increase)
Oper = 3: from cell 2 - to cell 1, delta 9514.700000 (increase)
Oper = 4: from cell 2 - to cell 1, delta 0.000000 (no change)
Oper = 5: from cell 1 - to cell 2, delta 47242.65000 (increase)

The second improvement move of the TS is to insert the 1st operation of product 13 into cell 1. After that, any feasible move considerably increases the cost solution because it will be accompanied by an increase in the acquisition cost of machines. It should be noted that if the probabilities ($\pi_{pjj'}$) were not considered, by moving the 2nd operation of product 13 from cell 2 to cell 1 had yielded in a delta value of zero. This same value variation of zero would be obtained by any operation j assigned to a cell 1 and contiguous to another operation j' assigned to a cell 2.

We conclude that for the disassembly cell formation problem to reach an improvement move some times it is required to move more than one operation at time. Therefore for the robust and reconfigurable variations of the DCFP (which are the topics of this part of the research) the λ-insertion move is proposed (see Section 5.3.3.3).

6.3.2 Experimental Framework for the Robust and Reconfigurable DCFP

The two versions of the VNS algorithm designed for solving the corresponding two versions of the disassembly cells formation problem with demand uncertainty were implemented in C using the Microsoft Visual C++ 6.0 programming environment under Windows Vista operating system. Both problems (DCFP-D and DCFP-R) were modeled using AMPL 11.1 and ILOG CPLEX 11.1 was used as the mixed-integer programming solver. All experiments were conducted on a Dell Intel Core 2 Quad PC with a processor operating at 2.66 GHz and 4.00 GB RAM.

To test whether the proposed algorithms are capable to solve efficiently real-world sized instances of the DCFP-R and DCFP-D problems, a sufficient number of instances is required

for analyzing the performance of the algorithms. However, instances were not available in the literature published to date. Moreover, to the best of our knowledge, the disassembly cell formation problem with demand uncertainty has not been addressed before. To overcome this drawback, we adapt some instances previously published for the manufacturing cell formation problem, and we design a random instance generator to complete the required data.

6.3.2.1 Data Treatment

In this section we describe the data set used to test the proposed algorithms. A set of 40 instances were used in the experimental work: 20 for the DCFP-D and 20 for the DCFP-R, each one with a different number of products and disassembly operations. Each instance contains the following data: the set of disassembly tasks (and its sequence) required to dismantle each product type, the set of machine types required to perform each operation and the available time of each machine type per stage, the time required to perform each disassembly task, the relative frequency with which each disassembly task is require, the demand in units for each product type during each stage, the amortized cost per period of one unit of each machine type, the reallocation cost of one unit of a machine type into cells, the cost of moving one unit of each product type between two cells, and the lower and upper bounds in terms of number of disassembly task allowed per cell. Twelve of the twenty instances generated for each problem (DCFP-D and DCFP-R) were adapted from problems previously used in the literature, where the operation sequences and machine types for the set of products used are described. In some instances all the required data are not specified in the source paper, hence they were generated as described in the instance generator. The remaining eight instances were randomly generated with the procedure described below.

Table 6.24 shows the reference for each instance type, the number of products (P) in the instance, the total number of operations (Q), disassembly tasks ($N = \Sigma_p Q_p$) and maximum number of operations required by any product (G), the number of machine types (M) included in the instance, the maximum number of cells to form (C_{max}) and the number of periods or scenarios ($T|S$) considered.

Table 6.24 Instance Set. Where (P) is the number of products in the instance, (Q) the total number of operations, the total number of disassembly tasks ($N = \Sigma_p Q_p$) , $G = \max_p\{Q_p\}$ is the maximum number of disassembly operations required by any product, C is the maximum number of cells to form, and ($T|S$) is the number of periods or scenarios considered.

Instance Type	Reference	P	Q (N) (G)	M	C	T
DS01	Cao and Chen (2005)	10	6 (25) (3)	6	3	3
DS02	Adenso-Díaz, Lozano, Andrés and García (2006)	14	16 (89) (8)	6	4	3
DS03	Ramabhatta and Nagi (1998)	15	15 (104) (10)	15	4	4
DS04	Selim (2002)	19	12 (75) (7)	10	3	3
DS05	Askin, Selim and Vakharia (1997)	19	12 (76) (6)	10	3	4
DS06	George, Rajendran and Ghosh (2003)	20	8 (61) (5)	8	3	5
DS07	Mungwattana (2000)	20	18 (74) (7)	18	3	4
DS08	Mungwattana (2000)	20	14 (102) (7)	14	5	3
DS09	Askin and Subramanian (1987)	24	14 (61) (4)	14	3	5
DS10	Defersha and Chen (2006)	25	40 (160) (9)	10	5	3
DS11	Pandian and Mahapatra (2009)	40	25 (133) (6)	25	5	4
DS12	Spiliopoulos and Sofianopoulou (2008)	50	30 (154) (6)	30	7	4
DS13	Randomly Generated	15	25 (100) (9)	10	4	5
DS14	Randomly Generated	20	25 (182) (13)	15	6	5
DS15	Randomly Generated	25	25 (223) (13)	18	7	4
DS16	Randomly Generated	30	50 (192) (10)	18	5	3
DS17	Randomly Generated	35	50 (197) (9)	20	5	3
DS18	Randomly Generated	40	75 (215) (9)	20	7	3
DS19	Randomly Generated	45	75 (228) (8)	25	7	2
DS20	Randomly Generated	50	100 (254) (8)	30	8	2

The data for the first instance type (DS01) was adapted from Cao and Chen (2005) where was used to test the performance of a tabu search algorithm in the design of a robust cellular configuration. The instance type DS02 was adapted from Adenso-Díaz *et al.* (2006), and corresponds to a real-world instance for the disassembly cells formation problem with 14 product types whose demand is known for a single period. The instance type DS11 was adapted from Defersha and Chen (2006), and corresponds to an instance used in the design of a reconfigurable cellular manufacturing system with lot splitting and system flexibility. The instances type DS07 and DS08 were adapted from Mungwattana (2000) and correspond to instances used in the design of cellular manufacturing systems with both dynamic and uncertain conditions. Finally, instances type DS03–DS06, DS09–DS10 and DS12 were taken from previously published papers addressing typical cell formation problems.

6.3.2.2 Instance Generator

In order to design the instance generator, let us consider that in a disassembly environment there are a wide variety of products with diverse characteristic such as volume, level of use and life-cycles that distinguish them. For example, there are products whose volume is small and whose demand is high, such as toasters or blenders. Other products such as TV sets or dishwashers, which are of common use, have short life-cycles and have a tendency to move with regular frequency. On the other hand, refrigerators whose cost is higher and whose life-cycle is longer are less frequently recovered. Finally, products like water heaters have long life cycles, and low demand and therefore its rate of return is low. Thus, three levels of demand for products can be identified. The demand for the first stage is generated accordingly the following schema: for each product type we generate a random number $\psi_1 \in [0,1)$, if $0 \leq \psi_1 < 0.20$, then a high demand is generated for such product using a uniform distribution with parameters (13000, 19000] and rounding to the nearest integer; if $0.20 \leq \psi_1 < 0.80$, then an intermediate demand is generated for such product using a uniform distribution with parameters (4000, 13000] and rounding to the nearest integer; finally, if $0.80 \leq \psi_1 < 1$, then an low demand is generated for such product using a uniform distribution with parameters (900, 4000] and rounding to the nearest integer. To obtain the demand for each of the remaining stages (periods) we adopt the same schema used for Schaller (2007) in which a percentage increase or decrease is randomly generated for each product and applied to the most recent stage's demand for the product to obtain the next stage's demand for the product. The percent increase or decrease for each product for each stage was generated using a uniform distribution with parameters [-50, 50] rounded to the nearest 10%. To obtain the occurrence probability of each scenario, it should be note that the higher probability assigned to a scenario will lead the solution closer to the optimal solution of the particular scenario, as empirically found by Cao and Cheng (2005), to overcome this we assign similar probabilities to each scenario. The intercellular movement cost was randomly generated using a normal distribution $c_i \sim N(\mu=1.34, \sigma=0.90)$ and taking the absolute value generated.

The number of operations required to dismantle a given product type $Q_p \subseteq Q$, theoretically, range from 1 (if only one operation is required) to card(Q) (if all the set of operations available are required), however both boundaries are unlikely in a real-world system. Therefore, we assume that the number of operations required to dismantle a product type until a predefined level, must be an integer between $\lceil \text{card}(Q)/8 \rceil$ and $\lceil \text{card}(Q)/2 \rceil$. We consider that the maximum number of cells to form in a given instance should be 10,

therefore, and with the aim to keep cells small and functional the minimum and maximum number of disassembly task allowed per cell are properly defined respectively. Given a subset Q_p of disassembly tasks required to dismantle a product type p, the sequence in which such tasks are required, is randomly generated as a feasible permutation of the disassembly tasks in Q_p. Note that two important issues should be considered: a) each one of the disassembly operations must be requested at least once by any of the product types, and b) one product cannot require more than once the same operation.

Usually, disassembly operations require manual class tools to break simple ties (e.g. cutting shears, screwdrivers, soldering irons), whose acquisition cost ranges from 45€ to 300€, and special mechanical multi-device equipment for trituration, separation and treatment of gases, whose cost reach 55,000€. Therefore the amortization cost per stage of a machine of each type was randomly generated accordingly to the following schema: for each machine type we generate a random number $\psi_2 \in [0,1)$, if $0 \leq \psi_2 < 0.10$, then the amortization cost is generated using a uniform distribution with parameters [30000, 53000] and rounding to the nearest integer; if $0.10 \leq \psi_2 < 0.40$, the amortization cost is generated using a uniform distribution with parameters [5000, 15000] and rounding to the nearest integer; if $0.40 \leq \psi_2 < 0.70$, the amortization cost is generated using a uniform distribution with parameters [900, 3000] and rounding to the nearest integer; finally, if $0.70 \leq \psi_2 < 1$, then the amortization cost is generated using a uniform distribution with parameters [45, 300] and rounding to the nearest integer. The cost for reallocating one machine of a type was set equal to 15% of the amortization cost of the type per stage and rounding to the nearest integer. The processing capacity of each machine type (in time units per stage) is randomly generated using a uniform distribution $H_m \sim U$ (102000 – 134200). The subset of machine types $M_q \subseteq M$ required to perform operation q is randomly taken from the set of machine types previously defined. We do not allow $|M_q|$ become greater than $\lceil M/2 \rceil$. It should be noted that some disassembly operations are performed manually and any machine is required in such for operation.

Processing times and relative frequencies were randomly generated. The processing time (in time units) for each disassembly task was generated using a uniform distribution with parameters [0.10, 5.00]. The relative frequency with which a disassembly task is required is randomly generated accordingly to the following schema: for each disassembly task we generate a random number $\psi_3 \in [0,1)$, if $0 \leq \psi_3 < 0.35$, then the relative frequency is 1 (i.e. such disassembly task is always required); if $0.35 \leq \psi_2 < 1$, then the relative frequency is generated using a uniform distribution with parameters [0.75, 1.00). This ensures that a large

percentage of disassembly tasks are required with a frequency of 1.00. All instances data are available upon request.

6.3.2.3 Experimental Framework

The experimental framework consists of three experiments whose main objective is the assessment of the relative performance of the proposed algorithms. In the comparison experiments we are interested in finding both optimal solutions and high quality feasible solutions. These are used to measure the performance of the heuristic algorithm in terms of its efficacy and efficiency. The best way to evaluate the efficacy of a heuristic algorithm is to compute the difference between optimal and heuristic solutions. However, given the complexity of our problem, optimal solutions are hard to find even for small instances. Therefore, we used two strategies that allow us to know not only the efficiency of CPLEX to find high quality feasible solutions but also lower bounds for the instance set.

Table 6.25 and Table 6.26 show the results obtained for the robust and reconfigurable approaches respectively. The first column of both tables shows the name of the instance. The second, third and fourth columns show the number of branch-and-bound nodes explored, the best feasible solution found by CPLEX when the default values for its algorithmic control directives were shifted to give more emphasis toward finding high quality feasible solutions, and the CPU time (in seconds) required to solve the problem instances respectively. Under these settings the MIP optimizer works to find high quality feasible solutions, rather than proving optimality, which allow the optimizer to quickly find feasible solutions that are otherwise very difficult to find. The purpose of this strategy is to determine the efficacy of CPLEX on finding high quality feasible solutions. For this strategy a 5 hr time limit was set, which we consider enough time to find high quality feasible solutions. The fifth and sixth columns show the solution reached by CPLEX and the optimality gap, when the solution found by our VNS procedure is provided to CPLEX as the starting solution. At this point we shift the CPLEX directives toward the optimality proof, which concentrates the optimizer effort on moving the best bound value.

The purpose of this strategy is to obtain a lower bound for the corresponding instance to compare the performance of the proposed VNS procedure. For this second strategy a 1 hr time limit was set, which we consider enough time to find high quality lower bound with an optimality proof. It should be noted that none of these emphasis settings changes the fundamental nature of the CPLEX branch & cut algorithm, which is to deliver proved optimal

solutions if given enough time; the setting merely changes some internal strategies along the way. The best solution found is presented in bold characters, and when the optimal solution is reached, it is identified with (*). Finally, the solution obtained by our VNS algorithm, the CPU time (in seconds) required, and the relative deviations of the heuristic solution from the best solution found are presented on columns 7 and 8 respectively. The relative deviation (DEV) is computed as $((Z_{VNS} - Z_{COMB})/ Z_{COMB}) \times 100$, where Z_{COMB} and Z_{VNS} are the combined solution obtained by VNS + CPLEX and our VNS solution respectively. Because VNS is reliant on randomization, the random number seed could influence the performance of the VNS. In order to control this variation we ran the algorithms five different times on the same problem instance using different random number seeds, and we have used the average of these five runs as its final result.

6.3.3 Results of the VNS for the Reconfigurable Problem

In this section we present the results of the reconfigurable DCFP. All instances produce mixed integer programming (MIP) models with more than 28,007 variables and 13,916 constraints. Experimental results show a good performance of the proposed algorithm. From Table 6.25, we can see that, except for a few cases when the optimal solution is found, our proposed procedure is able to find better solutions than CPLEX after 5 hrs of computing time. On average the heuristic solution obtained is 7.96% better that the CPLEX solutions. In only seven instances (five of them optimal) a deviation greater than 7% was obtained. Even for instance DS03_d the MIP optimizer was not able to find and integer solution after 5 hrs of computing time.

For the instances in which the optimal solution is found, the proposed VNS algorithm reach solutions at most 3.5% form optimality. When the VNS solution is provided to CPLEX as the initial solution most of the times these solutions are improved. Now the algorithm reach solutions above 2% form best-known solutions in an average computing time of 222.37 seconds.

Table 6.25. Results for the Reconfigurable Disassembly Cell Formation Problem.

The best solution founded is presented in bold characters, when the best solution corresponds to the optimal solution, this is indentified with (*).

The deviation is computed as: DEV = $((Z_{VNS} - Z_{COMB})/ Z_{COMB}) \times 100$, where COMB = VNS + CPLEX.

Instance	B & B Nodes	Best Feasible	Time (s)	VNS + CPLEX	(%) gap	VNS	Time (s)	DEV
DS01_d	579	**231764.00***	457.43	-----	-----	234158.28	0.30	1.03
DS02_d	196903	**314815.94***	754.30	-----	-----	324416.69	2.84	3.05
DS03_d	------	not found	18000	**1519705.36**	26.33	1521705.36	62.37	0.13
DS04_d	854133	554238.16	18000	**504509.05**	35.10	505509.05	8.44	0.20
DS05_d	192910	819819.26	18000	**733012.64**	23.21	734390.64	9.19	0.19
DS06_d	257349	**945407.00***	17880.76	-----	-----	**945407.00**	4.86	0.00
DS07_d	110	**53984.00***	230.54	-----	-----	55222.95	2.02	2.30
DS08_d	82447	1002753.05	18000	**1002753.05**	19.56	1037674.12	24.16	3.48
DS09_d	560211	489930.00	18000	**438672.47**	26.95	440541.47	5.56	0.43
DS10_d	36909	1469046.56	18000	**1258120.12**	31.26	1260706.12	56.83	0.21
DS11_d	44593	1503055.21	18000	**1241272.21**	29.25	1246504.21	163.25	0.42
DS12_d	12476	3208669.03	18000	**2692344.35**	16.25	2701276.35	269.32	0.33
DS13_d	98707	595897.00	18000	**590636.00**	26.65	592447.47	39.28	0.31
DS14_d	76436	**4164218.00***	16358.49	-----	-----	**4164218.00**	599.07	0.00
DS15_d	84397	3189248.88	18000	**2402430.24**	29.56	**2402430.24**	872.38	0.00
DS16_d	19806	3091534.50	18000	**2797660.24**	25.80	2810308.24	451.07	0.45
DS17_d	12988	2436868.01	18000	**2040484.98**	23.14	**2040484.98**	529.96	0.00
DS18_d	21271	2145895.17	18000	**2132311.17**	34.48	2286000.86	519.09	7.21
DS19_d	10811	1252693.65	18000	**1209700.53**	25.35	1214268.53	269.26	0.38
DS20_d	20604	2824918.65	18000	**2330733.51**	32.87	2343331.51	558.3	0.54

The initial solutions (obtained by the greedy procedure) are obtained in less than 2 seconds. In average the initial solutions are improved by 76.60 %. The swap move is required with a frequency 86.08%, while the union and the insert move are required with a frequency of 4.95% and 6.13% respectively. The λ=3 insertion move is required 2.35 % of the times while the λ=2 insertion move has a frequency below 1%.

We see that our algorithm finds solution of given quality faster than CPLEX. In fact for small problems (DS01-d, DS05_d, DS06-d and DS07_d) our algorithm converges to very good solutions in a very short computation time. However for difficult problems (from DS15_d to DS20_d) our algorithm requires on the average more than 540 seconds to reach solutions, that although are close to best-known solutions they have gaps of up to 30%.

6.3.3.1 Comparison with respect to a static cellular manufacturing design

One of the advantages of CMS is reduction in setups. However, setups may become an issue again when flexible layouts such as reconfigurable designs are used to handle part mix and demand changes. The setups costs (may) include machine reallocation costs. In this section we analyze the behavior (in total costs) of a dynamic cells design (in which physical cells are reconfigured periodically) versus a static cells design (in which cells stay static and demand changes are approaches through strategies such as worst case scenario). Given the assumptions under which the DCFP-D is modeled, i.e. the product demand varies from period to period in a deterministic manner over the entire planning horizon; an alternative static design could be to consider as the product's demand the largest demand of each product among the T periods, and solve the deterministic problem.

In this experiment the demand between periods (coefficient variation) varies from 10% to 100%. Five replications of each coefficient variation were performed, and the average value was taken as the result. Thus, fifty instances were designed to perform the comparative study. For solving the deterministic problem we use CPLEX, while for solving the reconfigurable problem we use the VNS algorithm. Figure 6.5 shows the behavior of both designs. Results show that the multi-period approach with a planned rearrangement of cells performs better than the static situation when the product's demand varies above 60%.On the other hand, when the demand's variation between periods is below 50% static designs are preferable. However, if the reallocation costs were lower, the difference could decrease. It was noted that when the product's demand varies above 60% intercellular movements were avoid at expenses of light increases in machine acquisition costs. This is straightforward to see that a higher product's demand results in higher transportation, therefore

products are disassembled in a single cell; however, it would be interesting to evaluate the percentage of utilization of machines, if there were any cost for lack of use, the graph could also show different results.

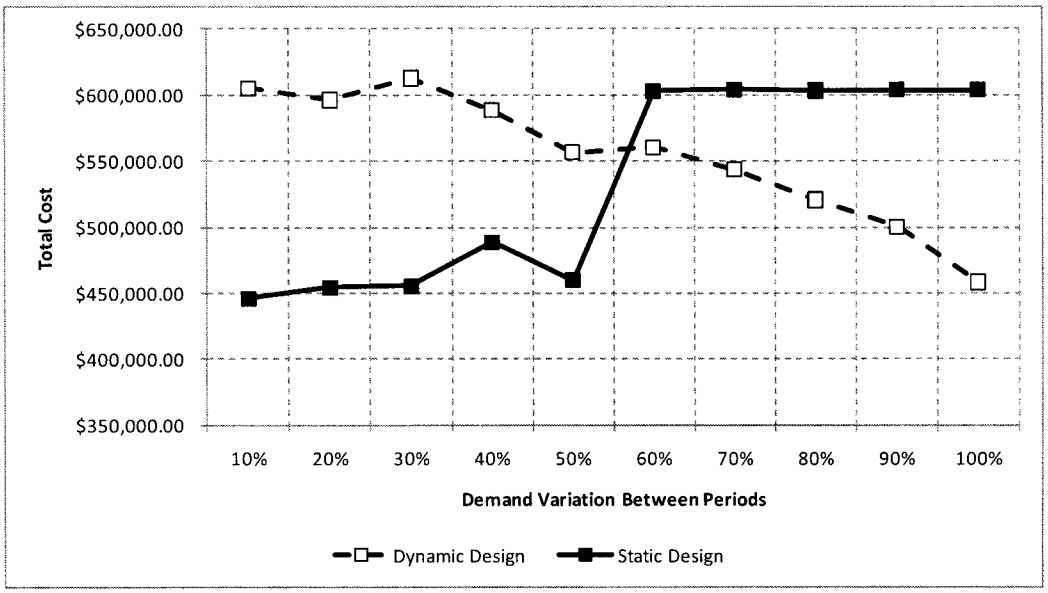


Figure 6.5 Comparison between a dynamic design versus an static design.

6.3.4 Results of the VNS for the Robust Problem

In this section we present the results for the robust DCFP. All instances produce mixed integer programming (MIP) models with more than 9,355 variables and 4,110 constraints. Experimental results show a regular performance of the proposed algorithm. From Table 6.26, we can see that, for small problem instances (from DS01_r to DS09_r and DS13_r) our proposed algorithm exhibits an average performance below 7.24%; however, for the remaining instances the heuristic solutions are up to 24.45%. For instances DS03_r, DS08_r and DS10_r our algorithm is able to find better solutions than CPLEX after 5 hrs of CPU time. In general, we see that our algorithm finds solutions faster than CPLEX in an average computing time of 326.51 seconds. It should be noted that CPLEX is able to find optimal solutions to most of the problem instances.

Table 6.26. Results for the Robust Disassembly Cell Formation Problem.

The best solution founded is presented in bold characters, when the best solution corresponds to the optimal solution, this is indentified with (*).

The deviation is computed as: DEV = $((Z_{VNS} - Z_{COMB})/ Z_{COMB}) \times 100$. where COMB = VNS + CPLEX.

Instance	B & B Nodes	Best Feasible	Time (s)	VNS + CPLEX	(%) gap	VNS	Time (s)	(%) DEV
DS01_r	305	**73574.00***	160	-----	-----	79298.00	19.03	7.78
DS02_r	149578	**120790.00***	1029.98	-----	-----	124351.00	71.92	2.95
DS03_r	183704	1175646.44	18000	**438441.00***	-----	460748.37	150.24	5.09
DS04_r	931191	**204722.94***	16137.93	-----	-----	221537.51	138.72	8.21
DS05_r	5869	**208322.87***	241.07	-----	-----	219267.33	65.83	5.25
DS06_r	10482	**214189.00***	140.55	-----	-----	214989.00	56.89	0.37
DS07_r	222	**13008.00***	422.17	-----	-----	13508.00	70.20	3.84
DS08_r	769856	2869422.07	18000	**341425.36***	-----	411068.49	124.20	20.40
DS09_r	5600	**107539.00***	144.64	-----	-----	120326.78	34.09	11.89
DS10_r	224663	565084.92	18000	**472492.4395**	14.11	540271.18	45.67	14.34
DS11_r	20915	**317744.72**	18000	**317744.72**	23.83	393850.11	372.92	23.95
DS12_r	72176	**759723.40**	18000	**759723.40**	29.56	906533.37	592.64	19.32
DS13_r	61460	**122627.00***	3057.58	-----	-----	130824.00	183.75	6.68
DS14_r	44724	**761851.30**	18000	**761851.30**	31.79	920019.26	230.78	20.76
DS15_r	43672	**584251.90**	18000	**584251.90**	22.21	706351.33	560.34	20.89
DS16_r	54046	**944911.74**	18000	**944911.74**	37.72	1118756.61	908.98	18.39
DS17_r	60484	753067.54	18000	**751116.95**	17.56	919239.04	960.52	22.38
DS18_r	19697	**675118.95**	18000	**675118.95**	28.12	840204.20	789.63	24.45
DS19_r	54201	**563125.88**	18000	**563125.88**	23.57	678952.00	638.97	20.56
DS20_r	15193	**1146495.35**	18000	**1146495.35**	29.83	1343284.51	514.86	17.16

When the VNS solution is provided to CPLEX as the initial solution, usually better lower bounds are obtained. Of particular interest are the instances DS03_r and DS08_r, which were improved until reaching the optimal solution. In such cases, our algorithm is 5.09% from the optimal solution for the instance DS03_r, but for the instance DS08_r it is 20.40% from optimum. The initial solutions (obtained by the greedy procedure) are also obtained in less than 2 seconds. On the average initial solutions are improved by 57.27%. The swap move is required with a frequency of 82.33%, while the union and the insert move are required with a frequency of 15.25% and 2.29% respectively. The λ-insertion moves are required less than 15 % of the times.

The effect of problem size on the time required can be seen by the results for problems DS14 to DS20 in both tables. It is interesting to note the effect of problem size on the performance of the VNS procedure for the robust problem with respect to the % vs best known.

The shaking procedures provide with considerable diverse solutions. In 14 of the 20 instances the *initial improved solutions* (those solutions which take the greedy solution and apply an exhaustive local search within all neighborhoods of the local search procedure) were improved up to 25%.

From the analysis of the results it can be inferred that the proposed heuristic algorithm performs better for the reconfigurable problem than for the robust problem. This could be due to the fact that when the union move is performed, the robust approach reduces the number of cells in all scenarios while the reconfigurable approach verifies each period independently, which let the algorithm to perform intermediate adjustments (moves) between successive union moves in the periods. However, we have not enough evidence to demonstrate (or refuse) whether it is so.

Our computational experiences show that for small instances (up to 100 disassembly tasks) of the robust problem the algorithm is effective and can reach near optimal solutions with less than 100 seconds. However, for problems much larger than such sizes, generally speaking, much more iterations will be needed to arrive at near optimal solutions. Probably by using a type of short term memory for the shaking phase would provide diverse solutions.

Based on the results of the test the proposed VNS procedure is highly recommended for the reconfigurable disassembly cells formation problem. However, for the robust problem some improvements are needed. Since the main interests of this part of the research were in modeling and designing a VNS algorithm for solving cell formation problems with varying demands (under robust and reconfigurable approaches), we plan to conduct more computational oriented investigations on different versions of VNS or other metaheuristics methods for the robust problem in our future work in this area.

Chapter 7

Conclusions and Future Research

This dissertation analyzes interesting real-life problems that lie in the field of Reverse Logistics (RL). We have addressed three closed interrelated logistics-manufacturing problems related to network design, vehicle routing and cellular manufacturing (disassembly). Even though there is plenty of published literature on each of these individual fields, new problems involving vehicle routing issues and cell formation issues are proposed and formulated, which have not been previous addressed in the literature. Therefore, this dissertation bridges a gap in existing literature. In addition, this dissertation uses real-world data (when possible) to test the proposed solution procedures. For example, the network design problem addresses a case of study for Galicia (northwestern Spain); also the algorithm designed for solving the vehicle routing problem with split loads and date windows is tested using real data; finally the algorithm designed for solving and analyzing the behavior of several neighborhoods in the disassembly cell formation problem is tested in real data. The use of industry representative data from Waste of Electric and Electronic Equipment (WEEE) in case studies ensures that these insights would be useful to a Collective Management System (CMS) that faces a similar situation.

The aim of the research project is on describing (Chapter 3), modeling (Chapter 4) and solving (Chapter 5) these logistics-manufacturing problems faced by a CMS when (re)designing a WEEE collection system. Therefore, after proving the problem background, describing the need for the research and defining its scope in Chapter 1, Chapter 2 reviews the essential literature published in order to acquaint the reader with the areas related to the research, which also will allow them to gauge the contribution of this research.

The remaining of this Chapter will draw on conclusions and future research directions for each one of the problems addressed in this dissertation.

7.1 Network Design Problem

In this part of the research we have described the implementation of an efficient hierarchical approach to design the reverse logistics network for collection of WEEE in a Spanish region. The associated facility location and collection routing problems were mathematically formulated as integer programming models. In both models the use of a heterogeneous fleet of capacitated vehicles is assumed. The design of the collection routes is complemented by a simulation analysis to complete a robust solution. Given the complexity of the routing problem a heuristic algorithm is developed to solve the related collection routing problems. By means of an iterative repetition of the route construction procedure the algorithm is able to perform a change on the in-use vehicle to improve the previous tours.

An experimental analysis was carried out on this algorithm to compare its performance against the optimal solutions in small instances. Results show a good performance in a computational time close to 0.1 seconds with an average deviation lower than 1.2% over the optimal solution. On their own, facility location and routing models provide a recovery network that outperforms the current one in use. The improved recovery network reduces the number of vehicles and depot size required.

Even though this part of the research is based on a case of study several potential directions for future research arose.

Regarding to the facility location model the real-world problem was easily solved by CPLEX v.11.1, therefore an alternative solution approach was not required. It dealt with a single family of recovered products for a single planning period. However there are certain characteristics of the recovery process like the uncertainty that would require a robust or stochastic approach to solve them. Also the price of the recovered products and the selling price after the recovery process (refurbishment or remanufacturing) could be considered, as well as the stochastic lead times.

In relation to the routing problem with heterogeneous fleet, the proposed heuristic algorithm proved to give high quality results. Therefore it may be enhanced to include other practical issues like time windows or split loads. Also this can be taken as a basis to design robust algorithm which include uncertainty in collected demand.

7.2 The VRP-SLDW

The Vehicle Routing Problem with Split Loads and Date Windows (VRP-SLDW) is a problem of practical relevance. Due to the computational complexity of the problem, it is important to develop polynomial time heuristic solution procedures. In this part of the research the problem was formulated as a Mixed Integer Programming model, and a GRASP based metaheuristic algorithm was designed to solve it. Computational results indicate that the proposed algorithm handles satisfactorily a real-life problem as well as other test random generated instances. Experimental results on a set of 540 random generated instances show the convenience of the algorithm to solve problems wherein the date window and the collection capacity are tightly constrained. These kind of problems are commonly found in real situations, as a result it is expected a good behavior in real problems. The performance of the metaheuristic algorithm was analyzed with respect to the amount of processing time required by the procedure to generate a solution and the quality of the solutions.

The results of the ANOVA test showed that factors like the date windows and the capacity tightness regarding demand are statistically significant for the algorithm performance. On the other hand, it cannot be demonstrated that a factor like the dispersion of the customers affects the quality of the obtained solutions, which is a sign of the robustness of the algorithm to deal with this kind of problems. Results from the local search analysis indicate that the local search indeed provide with considerably improved solutions. Especially relevant are the phase I local search moves which were able to reduce infeasibilities in all cases. Finally, the comparison of the proposed approach with current practice reveals that the transportation cost of the five depots fall an average of 15.96% from the current levels.

As future work it would be desirable to derive lower bounds for the VRP-SLDW to further evaluate the obtainable results. Modifications of existing exact solution procedures for the Split Delivery Vehicle Routing Problem (SDVRP) (Nowak, 2005) could yield such bounds. Also, there are many ways in which our proposed algorithm can be improved so that the number and length of tours needed to reach quality levels can be reduced, making its application to larger problem instances feasible. First, local optimization heuristics like 2-opt, 3-opt, GENUIS or Lin-Kernighan (Lin and Kernighan, 1973) can be embedded in the algorithm (this is a standard approach to improve efficiency of general purpose algorithms for vehicle routing problems) to improve the performance of long routes. Second, the development of alternative metaheuristics algorithms like Tabu Search or Scatter Search, which have proved to give good results in classical vehicle routing problems, could be considered.

Some further research could be directed towards an extension of the problem, for example considering two-level time windows, which include the classical time-windows into the date-windows, or taking into account that usually the customer imposes strict service constraints for collections on a specific day or even within given time windows during that day. Another issue of interest is based on the fact that usually collection orders have different priorities for the customer.

7.3 Disassembly Cell Formation Problems

In this part of the research, the Disassembly Cell Formation Problem (DCFP) was discussed and two variants were studied. The problems were formulated as integer programming models, and given its computational complexity heuristic algorithms were designed to solve them.

For solving the basic DCFP a Tabu Search (TS) algorithm was designed, and we illustrate the performance of previously published neighborhoods for the manufacturing cell formation problems (CFP) on a real-world test problem. The optimal solution was found by CPLEX and compared to the one obtained by the heuristic procedure. The performance of the heuristic procedure was analyzed with respect to the amount of processing time required to generate a solution and the quality of the solution, measured by its generated configuration cost. The results showed that the greedy heuristic procedure proposed (used as the initial solution for the TS) was very efficient and the quality of the solution reveals its effectiveness with 5.39% deviation from the optimal solution. The TS procedure generated relative good solutions with 2.93% deviation from the optimum in 33.7 % less time compared to CPLEX. We analyze the reason why the optimal solution was not found in this sample problem, and we conclude that some times more than one operation should be moved at a time to reach an improving move. Also, the differences on solving the DCFP versus the CFP were commented.

The DCFP with demand variability was mainly addressed in this part of the dissertation. Two different approaches were considered: robust and reconfigurable. In the reconfigurable approach the product demand varies form period to period in a *deterministic manner*. Due to this dynamic condition of the demand the best cell configuration for one period may not be efficient or even feasible for subsequent periods, and some redesigns would be required; for instance to change the number and allocation of machines to cells from period to period, to adjust to demand variability. The main assumption that supports this approach states that although the demand for each period is known a priori, finding an optimal configuration for each period could be prohibitively expensive. On the other hand, in the robust approach the product demand varies in a *random manner*.

However, this variation can be described in a number of probabilistic scenarios with a given occurrence probability. The aim of this approach is to obtain a cellular design in which the machines remain constant and do not move, only the flow of products changed once the demand have been observed. Both problems were mathematical formulated.

Based on the results obtained by the TS procedure and in its application on the basic DCFP, we designed a Variable Neighborhood Search (VNS) algorithm, which incorporate a new neighborhood based on the λ-insertion move, which reallocate λ disassembly task at time. Experimental results show the convenience of the algorithm on obtaining near optimal solutions for the reconfigurable problem; however for the robust problem some reservations are stated.

Results from this part of the dissertation raise new questions for several potential directions for further research. The first extension is in the robust DCFP problem realm, the development of alternative solution methodologies may be considered. We would like to see if incorporating memory into the shaking procedure could improve the efficiency of the algorithm. Also, it would be interesting to analyze if using a TS with varying tabu list sizes will improve the efficiency and quality in searching for optimal solutions of robust DCFP.

Additional research on the problem consists of several avenues. First, although it was not the aim of this dissertation, a larger experimental work may be conducted on the basic DCFP by comparing the proposed TS algorithm versus an implementation of a VNS algorithm, or even a hybrid VNS-TS, to derive a highly efficient algorithm for the DCFP. Second, the development of a metaheuristic algorithm for the minimax regret model proposed for the DCFP. Although such models require to known previously the best-known solution for each scenario in order to measure the regret, an efficient algorithm could consider an advanced strategy to constantly update the target solutions against to which compare the regret (see for example Yamashita et al., 2007). Finally, a future investigation may consider using the simulation-optimization framework for any DCFP problem with uncertain data.

Appendix A

Detailed description of the moves used in the VNS algorithm

This appendix provides a detailed description of the moves used in the Variable Neighborhood Search algorithm for solving the robust and reconfigurable disassembly cell formation problems.

Insertion Move (m_I)

Within this move a disassembly task is deleted from its current cell and reinserted in a different one. Thus, for each disassembly task i such that $x_s(i) = c$, if $m_I = \text{Move}(i, c, c', s)$ is performed then $x_s(i) = c'$ for each $c' \neq c \in C$. To be a feasible move, it requires that the cardinality of the source cell be greater than the minimum number of operations per cell $W_s(c) > L$ and the target cell must satisfy that $L \leq W_s(c') < U$. This move will increment the cardinality of the target cell by one unit, whilst reduce the carnality of the source cell in the same quantity. Therefore the matrix $W_s(c)$ is updated as follows: $W_s(c) = W_s(c) - 1$ and $W_s(c') = W_s(c') + 1$ if and only if the move $m_I = \text{Move}(i, c, c', s)$ is performed during stage s. Consequently, the neighborhood of an existing solution x, if $x \oplus m_I$ is defined as $N_I(x)$, and contains all those solutions obtained from x by changing the assignment in x of one disassembly task from its existing cell to another cell, during one stage. If there are C cells, N disassembly tasks and S stages then the size of this neighborhood is computed as $|N_I(x)| = S \times N \times (C - 1)$. Note that some of the potential neighboring solutions may be eliminated because they violate the cardinality constraints.

Swap Move (m_E)

Within this move two disassembly task belonging to different cells are swapping, while the rest of the tasks remain unchanged. Thus, for each pair of disassembly task i and j ($i \neq j$) such that $x_s(i) = c$ and $\hat{x}_s(j) = c'$, if m_E = Move(i, j, c, c', s) is performed then $x_s(i) = c'$ and $x_s(j) = c$. This move will not modify the cardinality of any cell during any stage, i.e. $W_s(c)$ and $W_s(c')$ remain unchanged, only the allocations will be modified. Consequently, the neighborhood of an existing solution x, if $x \oplus m_E$ is defined as $N_E(x)$, and contains all those solutions obtained from x by swapping the assignment in x of two disassembly tasks to cells, during one stage. The size of this neighborhood is computed as $|N_E(x)| = S \times \Sigma_{r=1\ldots(N-1)} (N - r)$. Note that some of the potential neighboring solutions may be eliminated because violates the cardinality constraints.

Union Move (m_U)

Within this move all the disassembly tasks of two different cells are put together. The implementation of this move is different in the robust problem than in the reconfigurable problem. The main difference lies in the fact that while in the reconfigurable problem the number of cells can vary from period to period, in the robust problem the number of cells must remain constant for all scenarios. The description of this move for each problem is described below:

Union move for the robust problem.

Within this move for each pair of cells c and c' ($c < c'$) the cost of reallocating all the disassembly tasks from c' to c in all the scenarios is evaluated. If m_U = Move(c, c') is performed then any disassembly task i such that $x_s(i) = c'$ changes to $x_s(i) = c$ for all s. Before performing the move m_U it must be verified if $W_s(c) + W_s(c') \leq U$ for all the scenarios s, to asses that the move is feasible. If $x \oplus m_U$ is a feasible move then $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) = W_s(c) + W_s(c')$, and $W_s(c') = 0$ for all s. Consequently, the neighborhood of an existing solution x, if $x \oplus m_U$ is defined as $N_U(x)$. The size of this neighborhood is computed as $|N_U(x)| = \Sigma_{r=1\ldots(C-1)} (C - r)$. Note that some

of the potential neighboring solutions may be eliminated because they violate the cardinality constraints.

Union move for the reconfigurable problem.

For each pair of cells c and c' ($c \neq c'$) all disassembly tasks i such that $x_s(i) = c'$, if m_U = Move(c, c', s) is performed then any disassembly task i assigned to cell c' is reallocated in cell c, i.e. $x_s(i) = c$. Before performing the move m_U it must be verified that $W_s(c) + W_s(c') \leq U$ to asses that the move is feasible. If $x \oplus m_U$ is a feasible move then $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) = W_s(c) + W_s(c')$, and $W_s(c') = 0$. Consequently, the neighborhood of an existing solution x, if $x \oplus m_U$ is defined as $N_U(x)$. The size of this neighborhood is computed as $|N_U(x)| = S \times \Sigma_{r=1\ldots(C-1)} (C - r)$. Note that some of the potential neighboring solutions may be eliminated because they violate the cardinality constraints.

Split Move (m_S)

To allow an increase in the number of cells in the search process, we consider the possibility of splitting one of the cells into two. As in the union move, the split move in the robust problem increments the number of cells in all scenarios, while in the reconfigurable problem the number of cells may be increased in only one period. The description of this move for each problem is described below:

Split move for the robust problem.

Given to cells c and c' ($c \neq c'$) with $W_s(c') = 0$ for all s, if m_S = Move(c, c', μ) is performed then μ disassembly task from cell c are chosen at random and reallocated in cell c', i.e. μ disassembly tasks move from $x_s(i) = c$ to $x_s(i) = c'$. This is only possible if the cardinality of the cell to be split is large enough to produce two cells with more than L tasks each for all scenarios. Thus, before evaluating the move m_S it must be verified the feasibility of the move, where $\mu = \min_s\{W_s(c) - L\}$, i.e. the maximum number of disassembly tasks that can be removed from cell c and reinserted into a cell c' in all scenarios. If $x \oplus m_S$ is a feasible move then $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) =$

$W_s(c) - \mu$, and $W_s(c') = \mu$ for all s, which is exactly the opposite to the union of two cells. Consequently, the neighborhood of an existing solution x, if $x \oplus m_S$ is defined as $N_S(x)$. If μ disassembly task will be changed then the size of this neighborhood is computed as $|N_U(x)| = \mu \times S \times \Pi_{r=1...(C-1)} (C - r)$. Note that some of the potential neighboring solutions may be eliminated because they violate the cardinality constraints.

Split move for the reconfigurable problem.

Given to cells c and c' ($c \neq c'$) with $W_s(c') = 0$, if $m_S = \text{Move}(c, c', \mu, s)$ is performed then μ disassembly tasks from cell c are chosen at random and reallocated in cell c', i.e. μ disassembly tasks move from $x_s(i) = c$ to $x_s(i) = c'$. This is only possible if the cardinality of the cell to be split is large enough to produce two cells with more than L tasks each. Thus, before evaluating the move m_S it must be verified if $W_s(c) - \mu \geq L$ in order to guarantee the feasibility of the move. Parameter μ is bounded by $L \leq \mu \leq surplus$, where $surplus = W_s(c) - L$ is defined as the maximum number of disassembly tasks that can be removed from cell c and reinserted into a cell c'. If $x \oplus m_S$ is a feasible move then $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) = W_s(c) - \mu$, and $W_s(c') = \mu$, which is exactly the opposite to the union of two cells. Consequently, the neighborhood of an existing solution x, if $x \oplus m_S$ is defined as $N_S(x)$. If μ disassembly task will be changed then the size of this neighborhood is computed as $|N_U(x)| = \mu \times S \times \Pi_{r=1...(C-1)} (C - r)$. Note that some of the potential neighboring solutions may be eliminated because they violate the cardinality constraints.

λ-Insertion Move (m_λ)

In previous experiments we found that sometimes it is required to move more than one disassembly task a time which belongs to the same cell to obtain an improving move, therefore we consider the movement of λ disassembly tasks a time. Within this move λ disassembly tasks are deleted from their current cell and reinserted into a different one. Thus, for each λ-tuple of disassembly tasks belonging to cell c, i.e. $x_s(i_r) = c$: $r = 1, \ldots, \lambda$, if $m_\lambda = \text{Move}(\{r1, \ldots, r_\lambda\}, c, c', s)$ is performed then the λ disassembly tasks are reinserted in cell c' ($c' \neq c$) during stage s, i.e. $x_s(i_r) = c'$: $r = 1, \ldots, \lambda$. Before evaluating the move m_λ it must be reviewed that $W_s(c) - \lambda \geq L$ and $L \leq W_s(c') + \lambda \leq U$ to asses that the move is feasible. If $x \oplus m_\lambda$ is a feasible move and it is performed, then $W_s(c)$ and $W_s(c')$ are updated as follows: $W_s(c) = W_s(c) - \lambda$, and $W_s(c') = W_s(c') + \lambda$. Consequently, the

neighborhood of an existing solution x, if $x \oplus m_\lambda$ is defined as $N_\lambda(x)$, and contains all those solutions obtained from x by changing the assignment of λ disassembly task from its current cell to another cell, during one stage. If λ disassembly tasks will be reinserted in a different cell, then the size of this neighborhood is computed as $|N_\lambda(x)| = \binom{N}{\lambda} \times (C - 1)$. Note that this previous general expression of the size of the neighborhood does not take into account that for some solutions the size of some neighborhoods may be less if some of the moves are infeasible because they violate the bounds of the number of minimum and maximum disassembly tasks per cell.

Appendix B

Illustrative example of the instance generator for the VRP-SLDW

This appendix exemplifies not only how the instance generator works, but also provides a valid argument regarding the quality of the BK solutions. For designing instances with BK high quality feasible solutions the following principles for designing routes are considered:

1. Optimal solutions are those in which the customer's demand is split as few as possible. This is consistent with the results of Archetti *et al.* (2006) for the split delivery routing problem.
2. Customers assigned to the same route are close to each other.
3. Routes with full truck load are preferable in optimal solutions.

We design a trial example with 15 customers, which allow us to illustrate its operation. The following assumptions are hold:

- There are 4 vehicles (two of 15 units of capacity and two of 30 units).
- Each vehicle can complete at most 3 tours.
- The total collection capacity of the system is 1,620 units per week.
- There are 15 customers (F1).
- We want to create customers with balanced demand (F2).
- We desire tightly date windows (F3).
- We desire a collection capacity above 15% over the total customer demand (F4).
- We desire customers with low dispersion (F5).

Step 1 – Compute customer demand considering factors F1, F2 and F4.

The total demand generated is of 1,381, which represents the 85.25% of the collection capacity of the systems. The demand for each customer is given in Table B.1.

Table B.1 Demand for each customer.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
85	87	85	110	105	97	90	74	100	86	92	92	99	90	89	**1381**

Step 2 – Form the LOAD matrix (i.e. the solution).

The load matrix is generated by taking the first customer with the largest demand and inserting a portion of its demand into the first route where both the largest load may be allocated and the remaining capacity of the vehicle is minimized. The procedure is repeated until the entire total customer's demand has been allocated. Then, the data structure is updated and the next customer is selected.

The solution is given in table B.2. To define the date windows, for each customer the procedure identifies the first and last day in which it is visited; then under the factor F3 consideration it determines the width of the date windows. Note that to ensure full truckload routes some customer demands were adjusted. Table B.3 shows the new demand for each customer.

Table B.3 New demand for each customer.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
90	90	90	120	105	97	90	75	105	90	92	96	105	90	90	**1425**

It should be noted that the difference between the customer with the higher demand (customer 4) and the customer with the lower demand (customer 8) is 45 units, and in average each customer holds a demand of 95 units, thus the customer demand is considered to be balanced.

Table B.2 LOAD matrix (solution) of the trial example.

		Customers																
Route		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	$\bar{Q}_k$	Q_k
Day 1	π^1_{11}								15								0	15
	π^1_{12}						7					2	6				0	15
	π^1_{13}									15							0	15
	π^1_{21}					15											0	15
	π^1_{22}					15											0	15
	π^1_{23}					15											0	15
	π^1_{31}	30															0	30
	π^1_{32}	30															0	30
	π^1_{33}	30															0	30
	π^1_{41}								30								0	30
	π^1_{42}								30								0	30
	π^1_{43}														30		0	30
Day 2	π^2_{11}					15											0	15
	π^2_{12}					15											0	15
	π^2_{13}					15											0	15
	π^2_{21}					15											0	15
	π^2_{22}													15			0	15
	π^2_{23}													15			0	15
	π^2_{31}														30		0	30
	π^2_{32}														30		0	30
	π^2_{33}															30	0	30
	π^2_{41}															30	0	30
	π^2_{42}															30	0	30
	π^2_{43}												30				0	30
Day 3	π^3_{11}													15			0	15
	π^3_{12}													15			0	15
	π^3_{13}													15			0	15
	π^3_{21}													15			0	15
	π^3_{22}													15			0	15
	π^3_{23}							15									0	15
	π^3_{31}												30				0	30
	π^3_{32}												30				0	30
	π^3_{33}						30										0	30
	π^3_{41}						30										0	30
	π^3_{42}						30										0	30
	π^3_{43}		30														0	30
Day 4	π^4_{11}							15									0	15
	π^4_{12}							15									0	15

	π^4_{13}							15									0	15
	π^4_{21}							15									0	15
	π^4_{22}							15									0	15
	π^4_{23}																15	15
	π^4_{31}		30														0	30
	π^4_{32}		30														0	30
	π^4_{33}			30													0	30
	π^4_{41}			30													0	30
	π^4_{42}			30													0	30
	π^4_{43}				30												0	30
Day 5	π^5_{11}																15	15
	π^5_{12}																15	15
	π^5_{13}																15	15
	π^5_{21}																15	15
	π^5_{22}																15	15
	π^5_{23}																15	15
	π^5_{31}				30												0	30
	π^5_{32}				30												0	30
	π^5_{33}				30												0	30
	π^5_{41}											30					0	30
	π^5_{42}											30					0	30
	π^5_{43}											30					0	30
Day 6	π^6_{11}																15	15
	π^6_{12}																15	15
	π^6_{13}																15	15
	π^6_{21}																15	15
	π^6_{22}																15	15
	π^6_{23}																15	15
	π^6_{31}										30						0	30
	π^6_{32}										30						0	30
	π^6_{33}										30						0	30
	π^6_{41}											30					0	30
	π^6_{42}											30					0	30
	π^6_{43}											30					0	30
	ω_i	90	90	90	120	105	97	90	75	105	90	92	96	105	90	90		

In this step a solution has been designed and the remaining of the procedure derive the instance data for that solution.

Step 3 – Form clusters and locate customers.

The size of the area to locate customers ranges from (-50 to 50) for *x* and *y*, the depot is located at (0, 0). To ensure high quality feasible routes, all customers assigned to the same route are grouped into clusters. The coordinates of the clusters are calculated, and its amplitude is defined by factor F5. Table B.4 shows the clusters that were identified and the center of the clusters. It should be noted that clusters are fairly distributed in the area. Therefore, two imaginary circles are drawn with radius of 50 and 25 for the outer and the inner respectively. Figure B.1 shows the location of the clusters.

Table B.4 Clusters.

Cluster	Elements	Center of the cluster
1	{1}	(50.00, 0.00)
2	{2}	(40.00, 29.05)
3	{3}	(20.40, 47.00)
4	{4}	(-9.89, 49.68)
5	{5}	(-33.35, 37.26)
6	{6, 11, 12}	(-48.26, 10.97)
7	{7}	(-47.62, -15.24)
8	{8}	(-5.52, -50.97)
9	{9}	(43.12, -25.32)
10	{10}	(22.00, 10.00)
11	{13}	(-16.61, 17.87)
12	{14}	(-15.83, -19.35)
13	{15}	(11.20, -22.35)

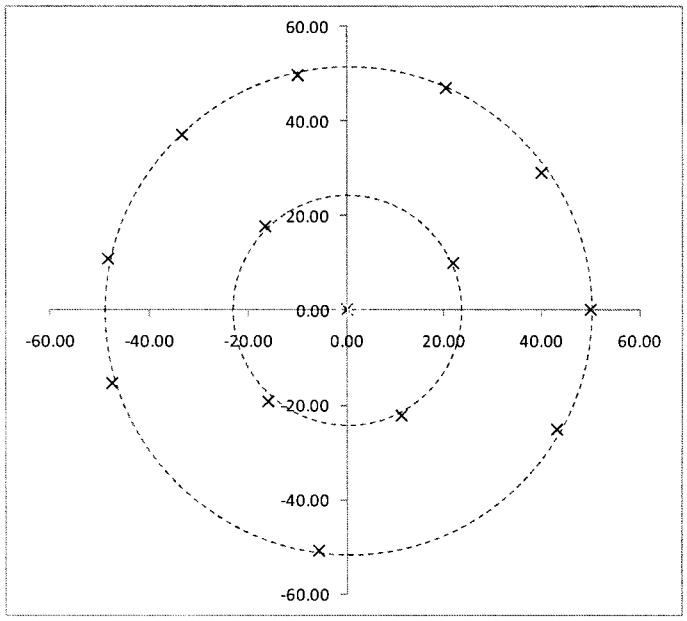


Figure B.1 Location of clusters

Now we allocate the customers around the center of their corresponding clusters according to factor F5. The coordinates of the customers are given in Table B.5.

Table B.5 Coordinates of customers.

Customer	*x*	*y*
DEP	0.00	0.00
1	49.00	-1.00
2	40.00	29.03
3	20.40	47.00
4	-9.89	49.01
5	-33.35	37.26
6	-48.78	10.97
7	-47.62	-15.24
8	-5.52	-49.69
9	43.12	-25.32
10	22.00	10.00
11	-47.00	11.06
12	-50.00	9.00
13	-17.48	17.87
14	-15.83	-19.35
15	11.20	-22.35

In figure B.2 we show the location of the customers.

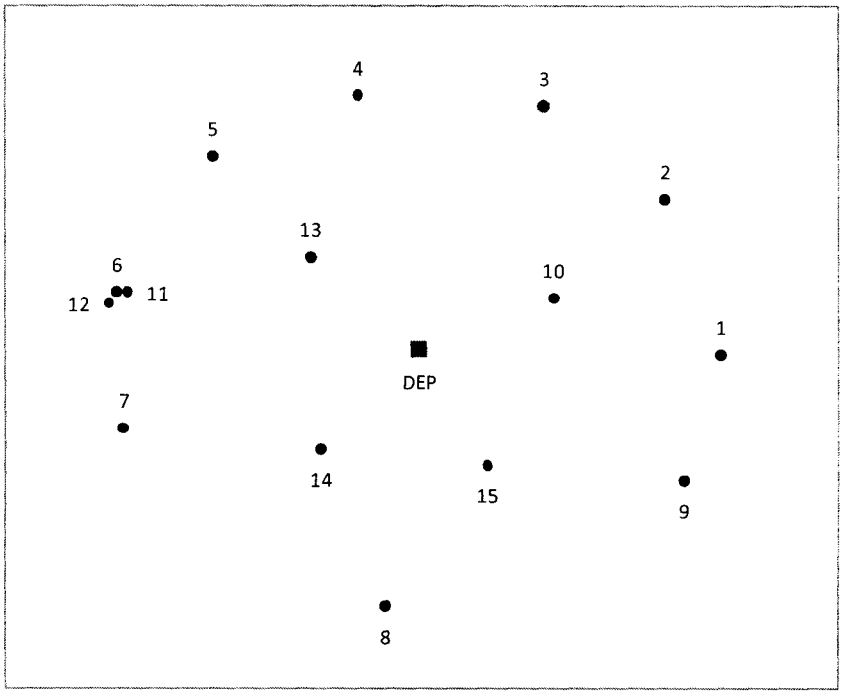


Figure B.2 Location of clusters

The distances are given in table B.6. With this information, the data instance is completed, which includes:

1. Number of customers.
2. Number of periods.
3. Number of vehicles.
4. Max number of tours per vehicle.
5. Capacity of each vehicle.
6. Unit cost per travel distance.
7. Max operation time (hrs) for each vehicle per day.
8. Demand for each customer.
9. Service time for each customer.
10. Date Windows.
11. Travel distance and time matrices.

Table B.6 Distance matrix. To compute the travel time, we consider an average speed of 50 km/hr.

	DEP	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
DEP	0.00	49.01	49.42	51.24	50.00	50.00	50.00	50.00	50.00	50.00	24.17	48.28	50.80	25.00	25.00	25.00
1	49.01	0.00	31.35	55.87	77.26	90.80	98.51	97.66	73.10	25.02	29.15	96.75	99.50	69.11	67.38	43.41
2	49.42	31.35	0.00	26.59	53.74	73.81	90.60	98.17	90.94	54.44	26.19	88.84	92.20	58.55	73.88	58.90
3	51.24	55.87	26.59	0.00	30.36	54.63	78.00	92.20	100.11	75.80	37.03	76.39	80.00	47.79	75.60	69.96
4	50.00	77.26	53.74	30.36	0.00	26.24	54.40	74.51	98.80	91.29	50.39	53.08	56.65	32.05	68.62	74.41
5	50.00	90.80	73.81	54.63	26.24	0.00	30.48	54.40	91.29	98.80	61.69	29.54	32.80	25.05	59.25	74.41
6	50.00	98.51	90.60	78.00	54.40	30.48	0.00	26.24	74.51	98.80	70.79	1.78	2.32	32.05	44.78	68.62
7	50.00	97.66	98.17	92.20	74.51	54.40	26.24	0.00	54.40	91.29	74.05	26.31	24.36	44.78	32.05	59.25
8	50.00	73.10	90.94	100.11	98.80	91.29	74.51	54.40	0.00	54.40	65.73	73.56	73.64	68.62	32.05	32.05
9	50.00	25.02	54.44	75.80	91.29	98.80	98.80	91.29	54.40	0.00	41.15	97.18	99.24	74.41	59.25	32.05
10	24.17	29.15	26.19	37.03	50.39	61.69	70.79	74.05	65.73	41.15	0.00	69.01	72.01	40.26	47.88	34.10
11	48.28	96.75	88.84	76.39	53.08	29.54	1.78	26.31	73.56	97.18	69.01	0.00	3.64	30.29	43.54	67.11
12	50.80	99.50	92.20	80.00	56.65	32.80	2.32	24.36	73.64	99.24	72.01	3.64	0.00	33.71	44.40	68.76
13	25.00	69.11	58.55	47.79	32.05	25.05	32.05	44.78	68.62	74.41	40.26	30.29	33.71	0.00	37.26	49.40
14	25.00	67.38	73.88	75.60	68.62	59.25	44.78	32.05	32.05	59.25	47.88	43.54	44.40	37.26	0.00	27.20
15	25.00	43.41	58.90	69.96	74.41	74.41	68.62	59.25	32.05	32.05	34.10	67.11	68.76	49.40	27.20	0.00

Step 5 – Form ROUTE matrix and compute total cost.

Now we can compute the ROUTE matrix, see table B.7, we only show those routes used in the solution.

Table B.7 Route matrix.

		Customers														
Route		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Day 1	π^1_{11}								1							
	π^1_{12}						2					3	1			
	π^1_{13}									1						
	π^1_{21}					1										
	π^1_{22}					1										
	π^1_{23}					1										
	π^1_{31}	1														
	π^1_{32}	1														
	π^1_{33}	1														
	π^1_{41}								1							
	π^1_{42}								1							
	π^1_{43}														1	
Day 2	π^2_{11}					1										
	π^2_{12}					1										
	π^2_{13}					1										
	π^2_{21}					1										
	π^2_{22}													1		
	π^2_{23}													1		
	π^2_{31}														1	
	π^2_{32}														1	
	π^2_{33}															1
	π^2_{41}															1
	π^2_{42}															1
	π^2_{43}												1			
Day 3	π^3_{11}													1		
	π^3_{12}													1		
	π^3_{13}													1		
	π^3_{21}													1		
	π^3_{22}													1		
	π^3_{23}							1								
	π^3_{31}												1			
	π^3_{32}												1			

	π^3_{33}						1									
	π^3_{41}						1									
	π^3_{42}						1									
	π^3_{43}		1													
Day 4	π^4_{11}							1								
	π^4_{12}							1								
	π^4_{13}							1								
	π^4_{21}							1								
	π^4_{22}							1								
	π^4_{31}		1													
	π^4_{32}		1													
	π^4_{33}			1												
	π^4_{41}			1												
	π^4_{42}			1												
	π^4_{43}				1											
Day 5	π^5_{31}				1											
	π^5_{32}				1											
	π^5_{33}				1											
	π^5_{41}											1				
	π^5_{42}											1				
	π^5_{43}											1				
Day 6	π^6_{31}									1						
	π^6_{32}									1						
	π^6_{33}									1						
	π^6_{41}										1					
	π^6_{42}										1					
	π^6_{43}										1					

The total cost of our solution is: 1,077.36.

Using AMPL – CPLEX to evaluate the solution

We have modeled this problem in AMPL and solve it in CPLEX. The adjusted problem after the *presolve* phase consists of 5,775 variables and 10,919 constraints. After 20 hr of computing time CPLEX explores 2,048,832 nodes however it was unable to find an integer solution. When we provide the BK solution to CPLEX the solution obtained was 1,077.36, which corresponds to the BK solution.

	B&B Nodes	Variables	Constraints	Time (s)	Solution	Optimality Gap (%)	Deviation from BK (%)
CPLEX	2048832	5775	10919	72000	Not found	Not info	Not info
BK+CPLEX	1396			36000	1077.36*	-----	0.00

Given the results obtained we don't have enough information to reject the hypothesis that our instance generator builds instances with a high quality upper bound.

References

Adenso-Díaz B., S. García-Carvajal and S. Lozano (2007). An efficient GRASP algorithm for Disassembly Sequence Planning. *OR Spectrum* 29: 535–549.

Adenso-Díaz B., S. Lozano, J. Racero and F. Guerrero (2001). Machine cell formation in generalized group technology. *Computers and Industrial Engineering* 41: 227–240.

Adenso-Díaz, B., S. Lozano, C. Andrés and J. M. García (2006). Disassembly cells design. *Proceedings of the Group Technology/Cellular Manufacturing Conference, J. Slomp and J. Riezebos (eds.), University of Gröningen, The Netherlands July* 3 - 5: 443–450.

Alonso, F., M. J. Álvarez and J. E. Beasley (2007). A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions. *Journal of the Operational Research Society* 59: 963–976.

Alshamrani, A. M. (2003). Combined routing and product take-back strategies in reverse logistics. *Case Western Reserve University*. PhD in Operations Management.

Álvarez, A., J. Pacheco, F. Angel-Bello and I. Garcia (2009). Un problema de distribución de productos alimenticios con flexibilidad en la fecha de entrega. *In: VI Congreso de Metaheuristicos, Algoritmos Evolutivos y Bioinspirados, Spain.*

Ammons, J. C., T. Assavapokee, D. Newton and M. J. Realff (2002). Reverse production system design for recycling under uncertainty. *Proceedings of the II manufacturing systems conference 4569*: 1–12.

Ammons, J. C., M. J. Realff and D. Newton (1997). Reverse Production System Design and Operation for Carpet Recycling, *Working Paper, Georgia Institute of Technology, Atlanta, G.A.*

Andrés C., S. Lozano and B. Adenso-Díaz (2007). Disassembly sequence planning in a disassembly cell context. *Robotics and Computer-Integrated Manufacturing* 23(6): 690–695.

Aras N., D. Aksen and A. G. Tanugur (2008). Locating collection centers for incentive-dependent returns under a pick-up policy with capacitated vehicles. *European Journal of Operational Research* 191: 1223–1240.

Archetti C., M. W. P. Savelsbergh and M. G. Speranza (2006). Worst-Case Analysis for Split Delivery Vehicle Routing Problems. *Transportation Science* 40(2): 226–234.

Askin, R. G. and J. B. Goldberg (2002). Design and Analysis of Lean Production Systems. *John Wiley & Sons, Inc. New York.*

Askin, R. G. and S. P. Subramanian (1987). A cost-based heuristic for group technology configuration. *International Journal of Production Research* 25(1): 101–113.

Askin, R. G., H. M. Selim and A. J. Vakharia (1997). A methodology for designing flexible cellular manufacturing systems. *IIE Transactions* 29(7): 599–610.

Assavapokee, T. (2004). Semi-continuous robust design for reverse production systems: a case study. *Georgia Institute of Technology, Atlanta, GA.* Ph.D. in Industrial Engineering.

Autry, C.W., P. J. Daugherty and R. G. Richey (2001). The challenge of reverse logistics in catalog retailing. I*nternational Journal of Physical Distribution & Logistics Management* 31(1): 26–37.

Azqueta, D. (2002). Introducción a la economía ambiental, *McGraw Hill, Madrid, España.*

Balakrishnan, J. and C. H. Cheng (2007). Multi-period planning and uncertainty issues in cellular manufacturing: A review and future directions. *European Journal of Operational Research* 177: 281–309.

Ballou, R.H. (1999). Business Logistics Management, *4th ed. Prentice-Hall, New York.*

Barba-Gutiérrez Y., B. Adenso-Díaz and M. Hopp (2008). An analysis of some environmental consequences of European electrical and electronic waste regulation. *Resource, Conservation and Recycling* 52(3): 481–495.

Barba-Gutierrez Y., Adenso-Diaz B. and Gupta S.M. (2008). Lot sizing in reverse MRP for scheduling disassembly. *International Journal Production Economics* 111: 741–751.

Barros, A. I., R. Dekker and V. Scholten. (1998). A two-level network for recycling sand: A case study. *European Journal of Operational Research* 110: 199–214.

Battarra, M., M. Monaci and D. Vigo (2009). An adaptive guidance approach for the heuristic solution of a minimum multiple trip vehicle routing problem. *Computer & Operations Research* 36: 3041–3050.

Bautista J. and J. Pereira (2006). Locating collection areas for municipal waste management: an application to the metropolitan area of Barcelona. *Omega: the International Journal of Management Science* 34(6): 617–629.

Beamon, B. M. and C. Fernandes (2004). Supply-chain network configuration for product recovery. *Production Planning & Control* 15: 270–281.

Beamon, B. M. (1999). Designing the green supply chain. *Logistics Information Management* 12 (4): 332–342.

Belfiore, P. and H. T. Yoshida-Yoshizaki (2009). Scatter search for a real-life heterogeneous fleet vehicle routing problem with time windows and split deliveries in Brazil. *European Journal of Operational Research* 199(3): 750–758.

Blanc, I., M. Van Krieken, H. Krikke and H. Fleuren (2006). Vehicle routing concepts in the closed-loop container network of ARN – a case study. *OR Spectrum* 28: 53–71.

Bloemhof-Ruwaard, J. M., L. N. Van Wassenhove, and H. L. Gabel (1996). An environmental life cycle optimization model for the European pulp and paper industry. *Omega: the International Journal of Management Science* 24: 615–629.

Brandão, J. and A. Mercer (1998). The multi-trip vehicle routing problem. *Journal of the Operational Research Society* 49: 799–805.

Brito, M. P. De. (2004). Managing Reverse Logistics of Reversing Logistics Management. *ERI - Erasmus University Rotterdam.* PhD Thesis.

Brito, M. P. De and S. D. P. Flapper (2003). Reverse Logistics: a review of case studies. *Erasmus Research Institute of Management (ERIM),* Erasmus University Rotterdam, Rotterdam, the Netherlands: 1–27.

Brito, M. P. De and R. Dekker (2003). A Framework for Reverse Logistics. *Erasmus Research Institute of Management (ERIM),* Erasmus University Rotterdam, Rotterdam, the Netherlands: 1–25.

Cao, D. and M. Chen (2004). Using penalty function and Tabu search to solve cell formation problems with fixed cell cost. *Computer & Operations Research* 31: 21–31.

Cao, D. and M. Chen (2005). A robust cell formation approach for varying product demands. *International Journal of Production Research* 43(8): 1587–1605.

Carter, C. R. and L. M. Ellram (1998). Reverse logistics: A review of the literature and framework for future investigation. *Journal of Business Logistics* 19(1): 85–102.

Chakravorty, S. S. and D. Hales (2008). The evolution of manufacturing cells: An action research study. *European Journal of Operational Research* 188: 153–168.

Chopra, S. and P. Meindl (2001). Supply Chain Management: Strategy, Planning, and Operation. *Prentice Hall, New Jersey.*

Clarke, G. and J. W. Wright (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operation Research* 12: 568–581.

Cordeau, J. F., G. Laporte, M. W. Savelsbergh and D. Vigo (2007). Vehicle routing. In: Barnhart C and Laporte G (eds). *Handbook in OR and MS 2007.* ELSEVIER: Philadelphia: 195–224.

Croxton, K. L., S. J. García-Dastugue, D. M. Lambert and D. S. Rogers (2001). The Supply Chain Management Process. *The International Journal of Logistics Management* 12(2): 13–36.

Dahel, N. and S. Smith (1993). Designing flexibility into cellular manufacturing systems. *International Journal of Production Research* 31: 933–945.

Daugherty, P. J., C. W. Autry and A. E. Ellinger (2001). Reverse logistics: the relationship between resources commitment and program performance. *Journal of Business Logistics* 21(1):107–124.

Daugherty P. J., M. B. Myers and R. G. Richey (2002). Information support for reverse logistics: the influence of relationship commitment. *Journal of Business Logistics* 23(1): 85–106.

Das, S.K., P. Yedlarajiah and R. Narendra (2000). An approach for estimating the end-of-life product disassembly effort and cost. *International Journal of Production Research* 38(3): 657–673.

Defersha, F. M. and M. Chen (2006). A comprehensive mathematical model for the design of cellular manufacturing systems. *International Journal of Production Economics* 103: 767–783.

Dekker, R. (2003). Return Management: An introduction and overview. available at: http://www.few.eur.nl/few/people/rdekker, visited 03/Oct/2007.

Dekker, R., J. Bloemhof-Ruwaard, M. Fleischmann, J. Van Nunen, E. Van del Laan and L. N. Van Wassenhove. (1998). Operational Research in Reverse Logistics: some recent contributions. *International Journal of Logistics: Research and Applications* 1(2): 141–154.

Dell'Amico, M., G. Righini and M. Salani (2006). A branch and price algorithm for the vehicle routing problem with simultaneous pick-up and delivery. *Transportation Science* 40: 235–247.

Desrochers, M. and G. Laporte (1991). Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters* 10: 27–36.

Dethloff, J. (2001). Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-ups. *OR Spectrum* 23: 79–96.

Dowlatshahi, S. (2000). Developing a Theory of Reverse Logistics. *Interfaces* 30(3): 143–155.

Dowlatshahi, S. (2002). A framework for strategic factors in reverse logistics. *Decision Sciences Institute, Annual Meeting Proceedings:* 425–430.

Dowlatshahi, S. (2010). A cost-benefit analysis for the design and implementation of reverse logistics systems: case studies approach. *International Journal of Production Research* 48(5): 1361–1380.

Dror, M. and P. Trudeau (1989). Savings by split delivery routing. *Transportation Science* 23: 141–145.

Dror, M., G. Laporte and P. Trudeau (1994). Vehicle routing with split deliveries. *Discrete Applied Mathematics* 50: 239–254.

Dror, M. and P. Trudeau (1990). Split delivery routing. Naval Research Logistics 37: 383–402.

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

European Parliament, the Council and the Commission. Directive 2002/96/EC of 27 January 2003 on waste electrical and electronic equipment (WEEE). Official Journal of the European Union L37 13.2.2003: 24-39. http://www2.uca.es/grup-invest/cit/Union%20Europea_archivos/WEEE_ingles.pdf, accessed 15 January 2009.

Feo, T. A. and M. G. C. Resende (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters* 8: 67–71.

Feo, T. A., M. G. C. Resende (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization* 6: 109–133.

Ferguson, N. and J. Browne (2001). Issues in end-of-life product recovery and reverse logistics. *Production Planning & Control* 12(5): 534–547.

Ferrer, G. and D. C. Whybark (2001). Material planning for a remanufacturing facility. *Production and Operations Management* 10(2): 112–124.

Fleischmann, M., P. Beullens, J. M. Bloemhof-Ruwaard and L. N.Van Wassenhove (2001). The impact of product recovery on logistics network design. *Production and Operations Management* 10: 156–173.

Fleischmann, M., J. M. Bloemhof-Ruwward, R. Dekker, E. Van der Laan, J. Van Nunen and L. N. Van Wassenhove (1997). Quantitative models for reverse logistics: a review. *European Journal of Operational Research* 103: 1–17.

Fleischmann, M., H. R. Krikke, R. Dekker and S. D. P. Flapper (2000). A characterization of logistics networks for product recovery. *Omega: the International Journal of Management Science* 28: 653–666.

Fleischmann, M., J. A. E. Van Nunen and B. Van Graeve (2003). Integrating Closed-loop Supply Chains and Spare Parts Management at IBM. *Interfaces* 33(6): 44–56.

Franke, C., B. Basdere, M. Ciupek and G. Seliger (2006). Remanufacturing of mobile phones-capacity, program and facility adaption planning. *Omega: the International Journal of Management Science* 34(6): 562–570.

George, A. P., C. Rajendran and S. Ghosh (2003). An analytical-iterative clustering algorithm for cell formation in cellular manufacturing systems with ordinal-level and ratio-level data. *International Journal of Advanced Manufacturing Technology* 22: 125–133.

Golden, B. L., A. Assad, L. Levy and F. Gheysens (1984). The fleet size and mix vehicle routing problem. *Computers & Operations Research* 11: 49–65.

González, B. and B. Adenso-Díaz (2006). A scatter search approach to the optimum disassembly sequence problem. *Computers & Operations Research* 33(6): 1776–1793.

Goosey, M. (2004). End-of-life electronics legislation - an industry perspective. *Circuit World* 30: 41–45.

Grunow, M. and C. Gobbi (2009). Designing the reverse network for WEEE in Denmark. *CIRP Annals - Manufacturing Technology* 58: 391–394

Guide Jr. V. D. R., V. Jayaraman and J. D. Linton (2003). Building contingency planning for closed-loop supply chains with product recovery. *Journal of Operations Management* 21(3): 259–79.

Guide Jr., V. D. R., R. Srivastava and M. E. Kraus (1997). Product structure complexity and scheduling operations in recovery manufacturing. *International Journal of Production Research* 35(11): 3179–3199.

Guide Jr., V. D. R. and L. N. Van Wassenhove (2001). Managing product returns for remanufacturing. *Production and Operations Management* 10(2): 142–155.

Guide Jr., V. D. R., T. P. Harrison and L. N. Van Wassenhove (2003). The Challenge of Closed-Loop Supply Chains. *Interfaces* 33(6): 3–6.

Guide Jr., V. D. R., V. Jayaraman, R. Srivastava and W. C. Benton (2000). Supply-Chain Management for Recoverable Manufacturing Systems. *Interfaces* 30(3): 125–142.

Guide Jr., V. D. R. and L. N. Van Wassenhove (2009). The Evolution of Closed-Loop Supply Chain Research. *Operations Research* 57(1): 10–18.

Guide Jr., V. D. R. V. Jayaraman and R. Srivastava (1999). The effect of lead time variation on the performance of disassembly release mechanisms. *Computers & Industrial Engineering* 36: 759–779.

Gungor, A. and S. M. Gupta (1998). Disassembly sequence planning for products with defective parts in product recovery. *Computers & Industrial Engineering* 35(1): 161–164.

Gungor, A. and S. M. Gupta (2001). A solution approach to the disassembly line balancing problem in the presence of task failures. *International Journal of Production Research* 39(7): 1427–1467.

Gungor, A. and S. M. Gupta (2002). Disassembly line in product recovery. *International Journal of Production Research* 40(11): 2569–2589.

Hansen, P., N. Mladenović and J. A. Moreno-Perez (2010). Variable neighborhood search: methods and applications. *Annals Operations Research* 175: 367 – 407.

Harhalakis, G., G. Ioannou, I. Minis and R. Nagi (1994). Manufacturing cell formation under random product demand. *International Journal of Production Research* 32: 47–64.

Harrington, L. (1994). The art of reverse logistics. *Inbound Logistics* 14: 29–36.

Hillegersberg, J. V., R. Zuidwijk, J. V. Nunen and D. V. Eijk (2001). Supporting returns flows in the supply chain. *Communications of the ACM* 44(6): 74–79.

Hu, T. L., J. B. Sheu and K. H. Huang (2002). A reverse logistics cost minimization model for the treatment of hazardous waste. *Transportation Research* Part E 38: 457–473.

Hughes, D. (2003). Reverse Thinking in the Supply-Chain. *EBSCO Business Review*: 30–36.

Imran, A., S. Salhi and N. Wassan (2009). A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research* 197: 509–518.

Jayaswal, S. and G. K. Adilz (2004). Efficient algorithm for cell formation with sequence data, machine replications and alternative process routings. *International Journal of Production Research* 42(12): 2419–2433.

Jeung-Ko, H. and G. W. Evans (2007). A genetic algorithm-based heuristic for the dynamic integrated forward/reverse logistics network for 3PLs. *Computers & Operations Research* 34(2): 346–366.

Johnson, M. R. and M. H. Wang. (1998). Economical evaluation of disassembly operations for recycling, remanufacturing, and reuse. *International Journal of Production Research* 36(12): 3227–3252.

Kara I., G. Laporte and T. Bektas (2004). A note on the lifted Miller–Tucker–Zemlin subtour elimination constraints for the capacitated vehicle routing problem. *European Journal Operation Research* 158: 793–795.

Kara, S., H. Kaebernick and P. Magnusson (2003) Modelling of product collection networks in Sydney Metropolitan Area: A simulation approach. In: *Proceedings of the CIRP Seminar on Life Cycle Engineering*, May 21–23, Copenhagen, Denmark.

Kara, S., F. Rugrungruang and H. Kaebernick (2007) Simulation modelling of reverse logistics networks. *International Journal of Production Economics* 106(1): 61–69.

Khoo, H. H., T. A. Spedding, I. Bainbridge and D. M. Taplin (2001). Creating a Green Supply Chain. *Greenleaf Publishing* autumn: 78–88.

Kim, H., J. Jaehwan and K. D. Lee (2009). Vehicle routing in reverse logistics for recycling end-of-life consumer electronic goods in South Korea. *Transportation Research* 14(5): 291–299.

Kizilkaya, E. A. and S. M. Gupta (1998). Material Flow Control and Scheduling in a Disassembly Environment. Computers & Industrial Engineering 35(1-2): 93–96.

Kokkinaki, A. I., R. Dekker, J. Van Nunen and C. P. Pappis (1999). An exploratory study on Electronic commerce for Reverse Logistics. *Econometric institute report*. Erasmus University of Rotterdam.

Kouvelis, P. and G. Yu (1997). Robust discrete optimization and its applications. *Kluwer Academic Publishers, Norwell, MA.*

Krikke, H.R. (1998). Recovery strategies and reverse logistic network design. *Institute for Business Engineering and Technology Application.* University of Twente, Enschede. The Netherlands. PhD. in Finance.

Krikke, H., J. Bloemhof-Ruwaard and J. N. Wassenhove (2001a). Design of Closed Loop Supply Chains: A Production and Return Network for Refrigerators. *Erasmus Research Institute of Management (ERIM),* Erasmus University Rotterdam, Rotterdam, the Netherlands: 1–33.

Krikke, H., C. P. Pappis, G. T. Tsoulfas and J. Bloemhof-Ruwaard (2001b). Design Principles for Closed Loop Supply Chains: Optimizing Economic, Logistic and Environmental Performance. *Erasmus Research Institute of Management (ERIM),* Erasmus University Rotterdam, Rotterdam, the Netherlands: 1–14.

Kroon, L. and G. Vrijens (1995). Returnable containers: an example of reverse logistics. *International Journal of Physical Distribution & Logistics Management* 25(2): 56–68.

Krumwiede, D. W. and C. Sheu (2002). A model for reverse logistics entry by third –party providers. *Omega: the International Journal of Management Science* 30(5): 325–333.

Lee, J. E., M. Gen and K. G. Rhee (2009). Network model and optimization of reverse logistics by hybrid genetic algorithm. *Computers & Industrial Engineering* 56: 951–964.

Li, Z., A. Kumar and Y. G. Lim. (2002). Supply chain modeling – a co-coordination approach. *Integrated Manufacturing Systems* 160: 268–287.

Lieckens, K. and N. Vandaele (2007). Reverse logistics network design with stochastic lead times. *Computers & Operation Research* 34: 395–416.

Lin, S. and B. W. Kernighan (1973). An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Operations Research* 21: 498–516.

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

Mandl, C. (1979). *Applied Network Optimization. Academic Press: New York.*

Melo, M. T., S. Nickel and F. Shaldanha-da-Gama (2009). Facility location and supply chain management – a review. *European Journal Operation Research* 196: 401–412.

Min, H., V. Jayaraman and R. Srivastava (1998). Combined location-routing problems: A research directions synthesis and future. *European Journal Operation Research* 108: 1–15.

Miner, S. (2001). Strategic safety stocks in reverse logistics supply chains. *International Journal of Production Economics* 71: 417–428.

Mitra, S. (2008). A Parallel Clustering Technique for the Vehicle Routing Problem with Split Deliveries and Pickups. *Journal of the Operational Research Society* 59: 1532–1546.

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

Moore, K. E., A. Gungor and S. M. Gupta (1998). A Petri net approach to disassembly process planning. *Computers & Industrial Engineering* 35(1-2): 165–168.

Mulvey, J. M., R. J. Vanderbei and S. A. Zenios (1995). Robust optimization of large-scale systems. *Operations Research* 43(2): 264–281.

Mungwattana, A. (2000). Design of cellular manufacturing systems for dynamic and uncertain production requirements with presence of routing flexibility. PhD. Thesis in Industrial and Systems Engineering, *Virginia Polytechnic Institute and State University.*

Nag, B., B. L. Golden and A. Assad (1988). Vehicle routing with site dependencies. In: Golden, B.L., Assad, A. (eds.). Vehicle Routing: Methods and Studies. *Studies in Management Science and Systems* 16: 149–159. North-Holland: Amsterdam, The Netherlands.

Nagy, G. and S. Salhi (2007). Location-routing: Issues, models and methods. *European Journal Operation Research* 177: 649–672.

Newton, J. D. (2000). A robust approach for planning the strategic infrastructure of reverse production systems. PhD. Thesis in Industrial Engineering. *Georgia Tech.*

Nsakanda, A. L., M. Diaby and W. L. Price (2006). Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings. *European Journal of Operational Research* 171: 1051–1070.

Nowak, M. A. (2005).The Pickup and Delivery Problem with Split Loads. PhD. Thesis in Industrial and Systems Engineering. *Georgia Institute of Technology.*

Pandian, R. S. and S. S. Mahapatra (2009).Manufacturing cell formation with production data using neural networks. *Computers & Industrial Engineering* 56: 1340–1347.

Papaioannou, G. and J. M. Wilson (2009). The evolution of cell formation problem methodologies based on recent studies (1997–2008): Review and directions for future research. *European Journal of Operational Research*, doi:10.1016/j.ejor.2009.10.020.

Peng-Kuan, H. and P. Jossef (1995). Warehouse location under service-sensitive demand. *Journal of Business Logistics* 16(1): 133–162.

Pogorelec, J. (2000). Reverse logistics is doable, important. *Frontline Solutions* 1 (10): 68–69.

Pohlen, T. L. and M. T. Farris (1992). Reverse logistics in plastics recycling. *International Journal of Physical Distribution and Logistics Management* 22(7): 35–47.

Prahinskia, C. and C. Kocabasoglub. (2006). Empirical research opportunities in reverse supply chains. *Omega: International Journal of Management Science* 34(6): 519–532.

Queiruga, D., G. Walther, J. Gonzalez-Benito and T. Spengler (2008). Evaluation of sites for the location of WEEE recycling plants in Spain. *Waste Management* 28(1):181–190.

Ramabhatta, V. and R. Nagi (1998). An integrated formulation of manufacturing cell formation with capacity planning and multiple routings. *Annals of Operations Research* 77: 79–95.

Ritchie, L., B. Burnes, P. Whittle and R. Hey (2000). The benefits of reverse logistics: the case of Manchester Royal Infirmary Pharmacy. *Supply Chain Management: An International Journal* 5(5): 226–233.

Robbins-Gentry, C. (1999). Reducing the cost of returns. *Chain Store Age* 75 (10), 124–126.

Rodrigue, J. P., B. Slack and C. Comtois (2001). Green Logistics. *Handbook of Logistics and Supply-Chain Management, Edited by: Brewer, A.M., Button, K.J. and Hensher, New York:* 339–350.

Rogers, D. S. and R. S. Tibben-Lembke (1998). Going Backwards: reverse logistics trends and practices. *Reverse Logistics Executive Council, Pittsburgh, PA.*

Rubio, S., A. Chamorro and F. J. Miranda (2008). Characteristics of the research on reverse logistics (1995-2005). *International Journal of Production Research* 46(4): 1099–1120.

Safaei, N., M. Saidi-Mehrabad and M. Babakhani (2007). Designing cellular manufacturing systems under dynamic and uncertain conditions. *Journal of Intelligent Manufacturing* 18: 383–399.

Sastry, S. Y. (2006). A methodology for evaluating the performance of reverse supply chains in consumer electronics industry. PhD. Thesis in Industrial Engineering. *The University of Texas at Arlington.*

Sato-Yamashita, D., V. Amaral-Armentano and M. Laguna (2007). Robust optimization models for project scheduling with resource availability cost. *Journal of Scheduling* 10: 67–76.

Savaskan, R. C., S. Bhattacharya and L. N. Van Wassenhove (2004). Closed-Loop Supply Chains Models with Product Remanufacturing. *Management Science* 50(2): 239–252.

Schaller, J. (2005). Tabu search procedures for the cell formation problem with intra-cell transfer costs as a function of cell size. *Computers & Industrial Engineering* 49: 449–462.

Schaller, J. (2007). Designing and redesigning cellular manufacturing systems to handle demand changes. *Computers & Industrial Engineering* 53: 478–490.

Schaller, J. (2008). Incorporating cellular manufacturing into supply chain design. *International Journal of Production Research* 46(17): 4925–4925.

Schultmann, F., M. Zumkeller and O. Rentz (2006). Modeling reverse logistic tasks within closed-loop supply chains: An example from the automotive industry. *European Journal Operation Research* 171: 1033–1050.

Secretaría del Medio Ambiente y Recursos Naturales (SEMARNAT). Programa Nacional para la Prevención y Gestión Integral de los Residuos 2009 – 2012. http://www.semarnat.gob.mx/informacionambiental/Publicacion/SEMARNAT%20Residuos%2009.pdf , accessed 18 November 2009.

Seifoddini, H. (1990). A probabilistic model for machine cell formation. *Journal of Manufactory System*: 69–75.

Seitz, M. A. and K. Peattie (2004). Meeting the closed-loop challenge: the case of remanufacturing. *California Management Review* 46(2): 74–89.

Selim, H. M. (2002). Manufacturing cell formation problem: a graph partitioning approach. *Industrial Management and Data Systems* 102(6): 341–352.

Selim, H. M., R. G. Askin and A. J. Vakharia (1998). Cell formation in group technology: review, evaluation and directions for future research. *Computers and Industrial Engineering* 34(1): 3–20.

Serrato-Garcia M. A. (2006). Outsourcing analysis for Reverse Logistics systems: a qualitative study and a Markov decision model. PhD. Thesis in Industrial Engineering. *Iowa State University.*

Sharma M. (2004). Reverse Logistics and environmental considerations in equipment leasing and asset management. PhD. Thesis in Industrial Engineering. *Georgia Institute of Technology.*

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

Singh, N. (1993). Design of cellular manufacturing systems: An invited review. *European Journal of Operational Research* 69(3): 281–511.

Soto, J. P. (2005). Reverse Logistics: Models and Applications. PhD. Thesis in Economics Management and Finance. *Universitat Pompeu Fraba.*

Spiliopoulos, K. and S. Sofianopoulou (2008). An efficient ant colony optimization system for the manufacturing cells formation problem. *International Journal of Advanced Manufacturing Technology* 36: 589–597.

Stock, J. R. (1992). Reverse Logistics. *Council of Logistics Management Oak Brook, IL.*

Stock, J. R. and C. J. Broadus (2006). Doctoral research in supply chain management and/or logistics related areas: 1999-2004. *Journal of Business Logistics* 21(1): 139 –151.

Stock, J. R. (2001). The 7 deadly sins of reverse logistics. *Material Handling Management* 56 (3): MHS5–MHS11.

Strom, J. R. (1997). Application of a reverse logistics model for optimizing scrap tire processing. MSc. Thesis in Mechanical and Industrial Engineering. *The University of Texas at El Paso.*

Taillard, E. (1993). Parallel iterative search methods for vehicle routing problems. *Networks* 23: 661–673.

Taillard, E., G. Laporte and M. Gendreau (1996). Vehicle routing problem with multiple use of vehicles. *Journal of the Operational Research Society* 47: 1065–1070

Tang O. and R. Teunter (2006). Economic lot scheduling problem with returns. *Department of Management Science. Lancaster University.*

Tang Y., M. Zhou, E. Zussman and R. Caudill (2002). Disassembly modeling, planning, and application. *Journal of Manufacturing Systems* 21(3): 200–217.

Tavakkoli-Moghaddam R., M. B. Aryanezhad, N. Safaei and A. Azaron (2005). Solving a dynamic cell formation problem using metaheuristics. *Applied Mathematics and Computation* 170: 761–780.

Teunter, R., E. Van der Laan. (2004). Valuation of inventories in product recovery systems in Reverse logistics, quantitative models for closed-loop supply chains, Dekker Fleischmann, Inderfurth, Wassenhove eds. 2004, *Springer.* 275–291.

Teunter, R., E. Van der Laan and K. Inderfurth (2000). How to set the holding cost rates in average cost inventory models with Reverse Logistics. *Omega: the International Journal of Management Science* 28: 409–415.

Teunter, R., Z. P. Bayyndyr and W. Van den Heuvel (2006). Dynamic lot sizing with product returns. *Department of Management Science. Lancaster University.*

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

Toffel, M. (2003). The growing strategic importance of end-of-life product management. *California Management Review* 45(3): 102–129.

Toktay, B., E. Van der Laan and M. P. de Brito (2003). Managing Product Returns: The Role of Forecasting. *Erasmus Research Institute of Management (ERIM),* Erasmus University Rotterdam, Rotterdam, the Netherlands: 1–27.

Topcu, A. (2009). A heuristic approach based on golden section simulation-optimization for reconfigurable remanufacturing inventory space planning. Ph.D. Thesis. *Northeastern University.*

Toth, P. and D. Vigo (Eds.) (2002). The Vehicle Routing Problem. *SIAM Monographs on Discrete Mathematics and Applications: Philadelphia.*

Van Beukering, P. J. H. and J. C. Van den Bergh (2006). Modeling and analysis of international recycling between developed and developing countries. *Resources, Conservation and Recycling* 46(1): 1–26.

Van der Laan, E. and M. Salomon (1997). Production planning and inventory control with remanufacturing and disposal. *European Journal of Operational Research* 102(2): 264–278.

Van der Laan, E., M. Salomon, R. Dekker and L. N. Van Wassenhove (1999). Inventory Control in Hybrid systems with remanufacturing. *Management Science* 45(5): 733–747.

Van Hoek, R. I. (1999). From reversed logistics to green supply chains. *Supply Chain Management* 4(3): 129–134.

Vuk, D. and B. Kozelj (1991). Application of multicriterial analysis on the selection of the location for disposal of communal waste. *European Journal of Operational Management* 55: 211–217.

Walther, G., E. Schmid and T. Spengler (2008). Negotiation based coordination in product recovery networks. *International Journal of Production Economics* 111(2): 334–350.

Walther, G. and T. Spengler (2005). Impact of the WEEE-directive on reverse logistics in Germany. *International Journal of Physical Distribution & Logistics Management* 35(5): 337–361.

Walther G., T. Spengler and D. Queiruga (2008). Facility location planning for treatment of large household appliances in Spain. *International Journal of Environmental Technology and Management* 8(4): 405–425.

Wemmerlöv U. and N. Hyer (1989). Cellular manufacturing in the US industry: A survey of users. *International Journal of Production Research* 27(9): 1511–1530.

Yaman, H. (2006). Formulations and valid inequalities for the heterogeneous vehicle routing problem. *Mathematical Programming* 106: 365–390.

Yen-Chin J. W. and C. Wei-Ping (2006). Reverse logistics in the publishing industry: China, Hon Kong, and Taiwan. *International Journal of Physical Distribution and Logistics Management*, 36(7): 507–523.

Zussman, E. (1995). Planning of disassembly systems. *Assembly Automation* 15(4): 20–23.