# Solving Manufacturing-Remanufacturing System Production Planning Problems Using Lagrangian Relaxation

**Pegah Abrishami Shirazi**

**A Thesis**

**in**

**The Department**

**of**

**Mechanical and Industrial Engineering**

**Presented in Partial Fulfillment of the Requirements**

for the Degree of Master of Applied Science (Industrial Engineering) at

Concordia University

Montreal, Quebec, Canada

**September 2011**

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: ____________________

Entitled: ____________________

and submitted in partial fulfillment of the requirements for the degree of

____________________

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Approved by

| | |
|---|---|
| Dr. O. Kuzgunkaya | Chair |
| Dr. Z. Tian | Examiner |
| Dr. G. Gouw | Examiner |
| Dr. M. Chen | Supervisor |

Approved by ____________________
Chair of Department or Graduate Program Director

____________________
Dean of Faculty

Date ____________________

# Abstract

In recent years, environmental legislation, societal pressure and economic opportunities have motivated many firms to integrate remanufacturing activities into the regular production environment. This presents many new challenges involving the collection, disassembly, refurbishing of used products and incorporation of remanufacturing activities into new product manufacturing. This research presents a mixed integer programming model addressing production planning problems in hybrid Manufacturing-Remanufacturing systems. The objective is optimizing an overall cost function based on an optimal number of new items to produce, number of items to be remanufactured, and number of new products to assemble in each time period of the planning horizon. A Lagrangian decomposition based method is developed to solve the problem efficiently. Numerical examples are presented to analyze the model performance and the developed solution procedure.

Keywords: Hybrid Manufacturing-Remanufacturing, Reverse Logistics, Optimum Policies.

# Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Mingyuan Chen, for his invaluable support and guidance during each stage of my graduate study and thesis preparation. His encouragement, insight, and patience made this entire effort possible.

I would like to appreciate my friend Dr Mehdi Towhidi for his help and his precious comments and critiques in fulfillment of my thesis.

I profoundly thank my parents, my brothers and my husband. I have been blessed with a family that remained dedicated and believed in me throughout the years. Their love and encouragement have enabled me to reach to this education level.

# Table of Contents

# List of Figure

# List of Table

# Chapter One

# Introduction

## 1.1 Foreword

Today’s manufacturing industries in many countries have started to develop systematic product recovery, remanufacturing and recycling procedures to reduce negative environmental impact. There are estimated more than  73,000 firms engaged in remanufacturing in the United States, directly employing over 350,000 people Remanufacturing account for total sales in excess of $53 billion per year.

Environmental friendly manufacturing activities, including product recovery, remanufacturing and recycle can substantially reduce material consumption and improve the technology and performance of manufacturing industry. Remanufacturing reserve the material and energy added in the primal manufacturing processes when the products were originally made.

## 1.2 Reverse Logistic

Fleischmann (2000) defined reverse logistics as “the process of planning, implementing, and controlling the efficient, effective inbound flow and storage of secondary goods and related information opposite to the traditional supply chain direction for the purpose of recovering value or proper disposal.” It covers activities leading recovering product values, remanufacturing, reuse, repair, recycle, etc.

### 1.2.1 Return Types

Returned Products may fill in one of the five categories based on Fleischmann (2000) definition of the different return types: end-of-use returns, commercial returns, warranty returns, production scraps and by-products, and packaging materials as explained below.

- **End-of-use returns**

  End-of-use products include products that have reached the end of their life, products that their use has been completed as well as leased product returns that can be used further.

- **Commercial returns**

  Commercial returns include product returns from costumers to sellers for refund.

- **Warranty returns**

  These types of returns are the products that are failed during use or damaged while delivered. They are returned to the manufacturer for refund or repair.

- **Production scrap and by-products**

  In many cases production scraps and by-products are of the nature of a process. However, they need to be recovered or recycled due to resource savings and economic considerations as well as environmental regulations.

- **Packaging materials**

  Returns of this type of product are desirable since they just need cleaning or minor maintenance. They can be reused directly in the same supply chain network. Examples for this category of returns can be crates, refillable bottles, pallets, and reusable boxes.

## 1.3 Green Production

Amezquita et al. (1995) and Ijomah et al. (1999) introduced the five processes of Reuse, Repair, Reconditioning, Recycling and Remanufacturing as Green Production processes. Among all, remanufacturing is highly desirable for ecological, economical and legislative considerations.

- **Reuse -** Using functional components from retired assemblies.

- **Repair -** Bringing the damaged product back to the functional conditions by fixing it.
- **Reconditioning -** Restoring components to the functional satisfactory level within original specifications. This may be achieved using resurfacing, repainting, etc.
- **Remanufacturing -** Bringing an assembly to like-new conditions through replacing and rebuilding component parts to required specifications.
- **Recycling -** Taking components and processing them to the original level or to useful degraded levels.

## 1.4 Remanufacturing

Many researchers have their own definitions for remanufacturing activities with the most comprehensive terminology proposed in Ijomah et al. (1999). In general, remanufacturing may refer to the following activities:

1. Receive the core that is the part of the product to be remanufactured. The term core is used since typical remanufactured parts are large core items of the products.
2. Strip and clean the core into individual elements as the used parts may be dirty. They are dismantled and appropriately cleaned. A visual inspection would discard badly damaged elements.
3. Estimate and quote remanufacturing costs. As many remanufacturing companies are subcontractors to the OEM (Original Equipment Manufacturer), the cost of remanufacturing is often estimated for each product to determine an appropriate rectification strategy.
4. Remanufacture. If the component is suitable, the appropriate machining/fabrication processes would be used to remanufacture the component to “as new” specifications.

5. Build, test and dispatch. Finally, the remanufactured components are assembled (together with necessary replacement components) to build the new product. After appropriate quality testing, the product would be dispatched for sale.

### 1.4.1 Motives in Remanufacturing

According to Amezquita et al. (1995), the main reasons for companies to practice remanufacturing are based on ecology, legislation and economic considerations.

- **Ecological factors**

  The amount of the waste generated in remanufacturing is noticeably less than manufacturing. Hence, remanufacturing reduces industrial waste considerably.

- **Legislative factors**

  Many government agencies have legislated strict environmental laws toward industrial waste. More policies have been generated concerning environmental attributes of manufactured products. More manufacturers are asked to take the responsibility of their product such as to take back their end-of-life products.

- **Economic factors**

  In general remanufacturing operations requires less capital investment and manufacturing operations since most of the main work has already been done during the primary manufacturing process. On the other hand, consumers constantly look for products of lower price and having the same or better quality than expensive items. Remanufactured products offer the opportunities for consumers looking for values in the products they purchase.

### 1.4.2 Remanufacturing Beneficiary

According to Giuntini and Gaudette (2003), the main beneficiaries of remanufacturing are :

- **Business enterprises**

  Capital goods remanufacturing as well as consumer goods remanufacturing are covered directly or indirectly throughout the original equipment manufacturer (OEM). Many enterprises are stakeholders in the successful expansion of remanufacturing.

- **The workforce**

  Remanufacturing environment is more dynamic in comparison to conventional manufacturing environment. Hence the workforce involved is required to have more initial training and skills. More training provides the workers with broader skills and longer term work satisfaction.

- **Consumers**

  A remanufactured product can be 40% less expensive than a similar new products.

- **Society**

  Society can be seen as the greatest beneficiary in remanufacturing era. Saving on energy and other natural resources are intrinsic social benefits.

### 1.4.3 Parties Involved

- **Third-party remanufacturing**

  Third-party remanufacturing is very common in remanufacturing industry especially in United States. One example is the automotive after-market providing consumers with replacement parts for their vehicles. Typical remanufactured components are: starters, alternators, water pumps, transmissions, and so on. The remanufacturers do disassembly, clean functional parts, add grease, paint or other material for protection, replace all worn parts, reassemble them, refurbish the exterior and test the reassemble unit. It is also common to offer warranty for the value-added remanufactured products by the third-part manufacturers separately from the original manufacturer.

- **Original-manufacturers remanufacturing**

  Original manufacturers are more capable to do remanufacturing work. They have the opportunity to use the same assembly line and other equipment for manufactured and remanufactured products.

### 1.4.3 Obstacles

Considering all the benefits of remanufacturing, yet surprisingly remanufacturing accounts for a very small portion of total production. Giuntini and Gaudette (2003) identified the following several factors.

- **Design**

  In most manufacturing industries, products are not designed for disassembly. The cost of remanufactured products may not be lower than new products.

- **Sales**

  Sales people are more willing to sell new products over remanufactured ones. They may view remanufactured product as a threat to the new ones.

- **Marketing**

  Selling remanufactured products has not been identified in marketing strategic plans in many companies. Marketing division mostly consider remanufacturing as the individual sale to the individual consumer at the time of a need.

- **Production and inventory management**

  There are more challenging issues in remanufacturing than producing new products. The required parts are (for new product production are mostly) known, while remanufacturing processes are associated with more uncertainties.

- **Workforce skill levels**

  Remanufacturing require broader technical skills from the workforce over the regular manufacturing environment. There are many techniques specialized to the remanufacturing that may not be required for regular workforce. Examples are disassembly, testing and selecting returned products.

- **Metrics**

  Businesses sometimes take revenue growth as their measure of performance. However, revenue tends to be greater in manufacturing than remanufacturing.

- **Advertising**

  To some extent, advertising tend to promote the latest and the most advanced version of a technology. This practice may not promote remanufactured products.

- **Accounting**

  Traditional accounting may provide the management with inaccurate financial performance of remanufacturing processes. This may cause the management to neglect the profitable effects of remanufacturing.

## 1.5 Scope and Objective of This Thesis

The purpose of this research is to develop a mixed integer programming model addressing production planning problems in hybrid manufacturing-remanufacturing systems (HMRS). The purposed model focuses on optimization of a cost function and determines the number of new products and remanufactured products of the considered HMRS process. An efficient solution approached is developed to solve the problem based on Lagrangian decomposition. The effectiveness and efficiency of the developed method are evaluated by computing several numerical examples and comparing the results with the generated lower bound of the objective function.

## 1.6 Thesis Organization

This thesis has six chapters. Following the introductory Chapter One, Chapter Two provides a review of the literature in remanufacturing. Chapter Three presents problem description and model formulation. Solution approach is presented and discussed in Chapter Four. Example problems are presented and solved in Chapter Five with results analysis. Finally, Chapter Six presents remarks and future research directions.

# Chapter Two

# Literature review

## 2.1 Introduction

Literature on remanufacturing system research is abundant. In this chapter, a review on the more recent and relevant literature to the work done in this thesis is presented.

## 2.2 Remanufacturing in General

Amezquita et al. (1995) discussed characteristics of products to be remanufactured in order to improve the process as a whole. They addressed some basic design features to be considered for remanufacturing. These features lead to time reduction in disassembly reassembly, and other operations of remanufacturing. They introduced a guideline in designing remanufacturable products considering the ease of disassembly, cleaning, inspection, part replacement, reassembly and the use of reusable components.

Giuntini and Gaudette (2003) discussed remanufacturing issues for represented to improved productivity. They addressed the remanufacturing beneficiaries as business enterprises, workforce, consumers and society. In this paper they identified several aspects to achieve more successful remanufacturing including product design, sales, marketing, production, inventory management, workforce skills, tax credits, among others.

King et al. (2004) discussed four approaches for waste reduction; repairing, reconditioning, remanufacturing and recycling. They argue that economic growth is the main cause of waste production in that production growth leads to consumerism followed by waste growth.

Ijomah et al. (2007) stated that long term sustainability requires a balance between economic or social development and environmental protection. Remanufacturing is the most profitable and environmental friendly approach towards sustainability together with recycling, reconditioning and repair. Some of the challenges facing remanufacturing include reluctant consumer acceptance, scarcity of remanufacturing tools and techniques, and poor remanufacturability of many products. The authors identified some influential product for improve remanufacturability features such as material, modularity and durability, part complexity, type of a fixing, joining methods, and so on.

Ijomah et al. (1999) discussed remanufacturing and differentiate it from other comparable green production alternatives. They discussed remanufacturing problems in terms of business process operations, since they are associated with high uncertainty and high risk due to difficulties in determining quality and quantity of returned products. The reasons undergoing the uncertainties attribute to variability in demand volume, core quality, core quantity, product type and availability of technical knowledge.

## 2.3 Hybrid Manufacturing-Remanufacturing Systems

Inderfurth (2004) discussed optimal policies for production control in hybrid manufacturing remanufacturing systems. The discussed problem is a single-stage and single-period problem with independent demand for two types of products. Lead times for these two types of products are deterministic and may not be the same. The objective is to maximize the expected profit to determine optimal manufacturing and remanufacturing order quantities with an arbitrary starting inventory level of serviceable products. The author discussed the problem for separate cases when manufacturing lead-time is shorter than remanufacturing lead-time and vice-versa.

Kim et al. (2006) considered a remanufacturing system where the parts can either be supplied by an external supplier or by using returned products cores. The considered remanufacturing system includes collection, disassembly, refurbishment and assembly operations. A mathematical model was developed for optimal production planning of this system. Numerical examples were presented to illustrate the considered problem and developed model.

Rubio and Corominas (2007) discussed issues related to implementing a reverse-logistics system for remanufacturing in a lean production environment. The studied problem considers a deterministic environment. A mathematical model was developed based on lean and just-in-time production. The purposed model allows manufacturing and remanufacturing capacities to be adjusted. The proposed model considers manufacturing and remanufacturing capacities, return rate and used rate for end-of-life products. The authors showed that remanufacturing is compatible with lean production practice. They also stated that combination of manufacturing, partial recovery, disposal and remanufacturing can lead to economic improvement and competitiveness.

Daniel and Guide (2000) studied industry practices and research needs of production planning (PP) and control (C) for remanufacturing. They identified seven complicating characteristics in production planning and control activities in remanufacturing.

1. Uncertain timing and quantity of returns
2. Need to balance returns with demands
3. Disassembly of returned products
4. Uncertainty in materials recovered from returned items
5. Requirement for a reverse logistics network
6. Complication of material matching restrictions

7. Problems of stochastic routings for materials for remanufacturing operations and highly variable processing time.

The authors listed threats to remanufacturing industry growth. Remanufacturing executives cited the increased pressure to reduce remanufacturing lead times continuously (60%), lack of formal systems to manage their business (38%), lack of cores to be remanufactured (50%), product designed for disposal (34%) and rapid technological changes (28%). The authors also discussed production planning and control problems in remanufacturing.

Behret and Korugan (2007) developed a mathematical model to analyze a hybrid system that meets the demand with remanufactured or new products. They classified the returned products to different quality levels for remanufacturing. Different quality levels correspond to different remanufacturing processing times, material recovery rates, remanufacturing costs and disposal costs. The authors compared their model with a benchmark model with no product classifications. They showed that classifying returned products according to quality level brings approximately 8% improvements in cost savings when the return rate is high.

Subramoniam et al. (2009) presented a literature review on research articles and future research needs in strategic decision making in remanufacturing and reverse logistic. They identified four strategic factors: product strategic planning processes, physical distribution structures, plant location and production systems, and cooperation among remanufacturing supply chain stakeholders. The authors also identified following aspects having positive impact on remanufacturing: global environmental regulations with proper incentives, needs to protect intellectual property, outside competition to remanufacture products, product design with consideration for product life cycle, increased interest to be a “green” company , good reverse logistic network , technology change and the resulting increasing disposal costs, increased product value, good core availability, a regional remanufacturing operation, eco-designed products, a well-integrated physical and non-physical organizational structure , good buyback or lease programs for products.

Kiesmuller (2003) discussed a production control problems in remanufacturing systems. The author also discussed pull and push policies in such systems. He introduced a new production control approach based on two different inventory positions of longer lead-time and shorter lead time.

Laan et al. (1999) investigated the influence of lead-time duration and lead-time variability on total expected costs in a system with manufacturing and remanufacturing operations. To control the system they applied both Push and Pull strategies. The authors used numerical examples to illustrate the effects of lead-time duration and lead-time variability on total expected costs in production/inventory systems with remanufacturing. The outcomes from the numerical examples show that:

1. Manufacturing lead-times have a larger influence on system costs than remanufacturing lead-times.
2. Increase in manufacturing lead-times may result in larger cost than equivalent increase in remanufacturing lead-times. Longer manufacturing lead-times require higher safety stocks to protect against costly stock-out events than equivalently longer remanufacturing lead-times.
3. Increase in variability of remanufacturing lead-times results in an increase in total expected costs, both under a Push and Pull strategies. The authors did not analytically prove this point.

Teunter et al. (2008) discussed multi-product economic lot scheduling problems with returns (ELSPR) where there are separate production lines for manufacturing and remanufacturing. The authors developed a mixed integer programming (MIP) model to solve the problem for a fixed cycle time which can be combined with a cycle time search to find an optimal solution. The authors used a numerical experiment to analyze the effects of switching a single production line to separate lines. these examples revealed that setting up dedicated lines for manufacturing and remanufacturing can lead to significant reduction in holding costs through lower production rate and increased scheduling flexibility. On the other hand, separate lines will require additional investment cost. A trade-off analysis is required in future research.

## 2.4 Summery

Remanufacturing has been studied for several decades and has become more importance in today's world. Much research work has been done in developing mathematical models for HMRS analysis. Various solution approaches have been developed to overcome the difficulties in solving HMRS problems and to further improve them for real world application. However, many aspects of HMRS systems still remain to be discussed. Due to the all the benefits and importance of the remanufacturing and increasing a demand for remanufactured items in this research we are going to study the hybrid manufacturing and remanufacturing systems.

In the next chapter, a new mathematical model for HMRS system optimization is presented. Wagner-Within method and a new solution approach based on dynamic programing are employed to solve the proposed model.

# Chapter Three

# Problem Description and Model Formulation

## 3.1 Problem Description

This chapter presents a detailed discussion of certain production planning problems in HMRS systems. The considered HMRS system uses both manufactured and remanufactured parts to assemble products for different markets. The remanufacturing process covers disassembly, inspection as well as machining used items to produce "as good as new" items. At the same time the manufacturing process makes new components using new materials. Inventory controls are required for returned products, remanufactured items, new manufactured components and finished products.

In the considered HMRS system, the acquired returned products may enter the remanufacturing process directly or through inventory. Returned products entering the remanufacturing process will be disassembled. The parts from the disassembly process will be inspected to determine if they can be reused or should be disposed. Remanufactured items may be kept in inventory for later use or may continue in the process to the assembly stage. The process of manufacturing new parts is similar. Manufactured new items may be kept in inventory before entering the assembly stage. Finished products made from new items or remanufactured ones. They may also be kept in inventory before delivery.

## 3.2 Model Assumption

The following list presents specific assumptions in developing the model to solve the HMRS production problem.

1. The model is based on multiple time periods.

2. Deterministic demands for products made from new components and remanufactured components.
3. Deterministic demands for both new and remanufactured components.
4. Quality of remanufactured components is “as good as the new”. However, their market may be different from that of new products.
5. Deterministic recovery rate of the components from the returned products.
6. Capacity limit on available production time for producing both new and remanufactured products.
7. Remanufacturing time includes disassembly time and inspection time for returned products.
8. Inventory costs are incurred for product or items held in inventory.
9. Setup costs are incurred for disassembly, remanufacturing, as well as manufacturing and assembly.
10. There are separate manufacturing lines for each of the operation on new components, remanufactured components, returned and new products.
11. A common resource (such as labour force) is shared by manufacturing and remanufacturing operations in the system.
12. No order of manufacturing, remanufacturing or assembly will be place at the time when there is enough of inventory ( Wagner-Whitin condition ).

## 3.3 Model Notations

### Index sets

$i = \{1, \dots I\}$ Number of components in each unit of products in time period *t*

$j = \{1, \dots J\}$ Number of products produced in a time period *t*

$t = \{1, \dots T\}$ Number of time periods

**Variables**

| | |
|---|---|
| $x_{i,t}$ | Number of new component $i$ to produce in time period $t$ |
| $e_{i,t}$ | Number of new component $i$ in inventory in time period $t$ |
| $\bar{x}_{i,t}$ | Number of remanufactured component $i$ to produce in time period $t$ |
| $\bar{e}_{i,t}$ | Number of remanufactured component $i$ in inventory in time period $t$ |
| $d_{j,t}$ | Number of returned product $j$ to disassemble in time period $t$ |
| $d_{j,t}^{n}$ | Number of new product $j$ to assemble from the new components in time period *t* |
| $d_{j,t}^{R}$ | Number of new product $j$ to assemble from the remanufactured components in time period *t* |
| $r_{j,t}$ | Number of returned product $j$ to acquire in time period $t$ |
| $f_{j,t}$ | Number of returned product $j$ in inventory in time period $t$ |
| $f_{j,t}^{n}$ | Number of new product $j$ in inventory made out of new component in time period *t* |
| $f_{j,t}^{R}$ | Number of new product $j$ in inventory made out of remanufactured components in time period *t* |

$$\theta_{i,t} = \begin{cases} 1, & \text{if the system is set up to make new component } i \text{ in time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

$$\bar{\theta}_{i,t} = \begin{cases} 1, & \text{if the system is set up to remanufacture returned component } i \text{ in time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

$$\delta_{j,t} = \begin{cases} 1, & \text{if returned product } j \text{ will be disassembled in time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

$$\delta^{n}_{j,t} = \begin{cases} 1, & \text{if new product } j \text{ will be assembled from new component in time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

$$\delta^{R}_{j,t} = \begin{cases} 1, & \text{if new product } j \text{ will be assembled from remanufactured components in time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

**Parameters**

| | |
|---|---|
| $P_{i,t}$ | Unit manufacturing cost of new component $i$ in time period $t$ |
| $S_{i,t}$ | Setup cost for manufacturing new component $i$ in time period $t$ |
| $V_{i,t}$ | Unit inventory cost for new component $i$ in time period $t$ |
| $\overline{P}_{i,t}$ | Unit remanufacturing cost of recovered component $i$ in time period $t$ |
| $\overline{S}_{i,t}$ | Setup cost for remanufacturing recovered component $i$ in time period $t$ |
| $\overline{V}_{i,t}$ | Unit inventory cost for remanufactured component $i$ in time period $t$ |
| $D_{i,t}$ | Demand for new component $i$ in time period $t$ |
| $\overline{D}_{i,t}$ | Demand for remanufactured component $i$ in time period $t$ |
| $r^{n}_{j,t}$ | Demand for new product $j$ made out of new components in time period $t$ |
| $r^{R}_{j,t}$ | Demand for new product $j$ made out of remanufactured components in time period $t$ |
| $B_{i,j}$ | Number of component $i$ contained in returned product $j$ |
| $B^{n}_{i,j}$ | Number of new component $i$ contained in new product $j$ |
| $B^{R}_{i,j}$ | Number of remanufactured component $i$ contained in new product $j$ |
| $UR_{i}$ | Average recovering rate of component $i$ from all returned products |
| $AQ_{j,t}$ | Unit cost to acquire returned product $j$ in time period $t$ |
| $RD_{j,t}$ | Unit cost to disassemble returned product $j$ in time period $t$ |

| | |
|---|---|
| $RD_{j,t}^{n}$ | Unit cost of assembly for new product $j$ made out of new components in time period $t$ |
| $RD_{j,t}^{R}$ | Unit cost of assembly for new product $j$ made out of remanufactured components in time period $t$ |
| $SD_{j,t}$ | Unit setup cost for disassembling product $j$ in time period $t$ |
| $SD_{j,t}^{n}$ | Unit setup cost to assemble new product $j$ from new components in time period $t$ |
| $SD_{j,t}^{R}$ | Unit set up cost to assemble new product $j$ from remanufactured components in time period $t$ |
| $IN_{j,t}$ | Unit inventory cost for storing returned product $j$ in time period $t$ |
| $IN_{j,t}^{n}$ | Unit inventory cost for storing new product $j$ from the new components in time period $t$ |
| $IN_{j,t}^{R}$ | Unit inventory cost for storing new product $j$ from the remanufactured components in time period $t$ |
| $ACAP_t$ | Available production time in time period $t$ |
| $AST_i$ | Production time for manufacturing new component $i$ |
| $ST_i$ | Setup time for manufacturing new component $i$ |
| $ASR_i$ | Production time for remanufacturing returned component $i$ |
| $SR_i$ | Setup time for remanufacturing returned component $i$ |
| $M$ | A large positive number |

## 3.4 Model Formulation

The mathematical model to solve the production planning problem in HMRS systems is formulated as follows in this section with the variables and parameters defined in the previous section.

### 3.4.1 Objective Function

The objective function of the model is to minimize the total cost in the HMRS systems. It includes the cost of manufacturing and remanufacturing, cost of disassembly and assembly, setup costs for disassembly, manufacturing, remanufacturing and assembly, inventory holding costs of the returned products, remanufactured and new parts and holding cost of the finished products.

$$Min\ z = \sum_{t=1}^{T}\sum_{i=1}^{I}\left(P_{i,t}x_{i,t} + S_{i,t}\theta_{i,t} + V_{i,t}e_{i,t} + \bar{P}_{i,t}\bar{x}_{i,t} + \bar{S}_{i,t}\bar{\theta}_{i,t} + \bar{V}_{i,t}\bar{e}_{i,t}\right) \tag{3.1}$$

$$+ \sum_{t=1}^{T}\sum_{j=1}^{J}\left(AQ_{j,t}r_{j,t} + SD_{j,t}\delta_{j,t} + RD_{j,t}d_{j,t} + IN_{j,t}f_{j,t} + SD_{j,t}^{n}\delta_{j,t}^{n} + RD_{j,t}^{n}d_{j,t}^{n} + IN_{j,t}^{n}f_{j,t}^{n}\right.$$

$$\left. + SD_{j,t}^{R}\delta_{j,t}^{R} + RD_{j,t}^{R}d_{j,t}^{\boldsymbol{R}} + IN_{j,t}^{R}f_{j,t}^{R}\right)$$

The solution of the model is to determine the optimal number of items to be produced and remanufactured as well as optimal number of returned products to acquire in each period of time. There is also a decision to make on the optimal number of products to get assembled in each period of time based on the known demand for both new and remanufactured products.

### 3.4.2 Constraints

$$e_{i,t-1} + x_{i,t} - e_{i,t} = D_{i,t} \tag{3.2}$$

$$x_{i,t} \le M\theta_{i,t} \tag{3.3}$$

$$\bar{e}_{i,t-1} + \bar{x}_{i,t} - \bar{e}_{i,t} = \bar{D}_{i,t} \tag{3.4}$$

$$\bar{x}_{i,t} \le M\bar{\theta}_{i,t} \tag{3.5}$$

$$d_{j,t} \le M\delta_{j,t} \tag{3.6}$$

$$d_{j,t}^{n} \le M\delta_{j,t}^{n} \tag{3.7}$$

$$d_{j,t}^{R} \le M\delta_{j,t}^{R} \tag{3.8}$$

$$f_{j,t} + d_{j,t} - f_{j,t-1} = r_{j,t} \tag{3.9}$$

$$f^{n}_{j,t} + d^{n}_{j,t} - f^{n}_{j,t-1} = r^{n}_{j,t} \tag{3.10}$$

$$f^{R}_{j,t} + d^{R}_{j,t} - f^{R}_{j,t-1} = r^{R}_{j,t} \tag{3.11}$$

$$\sum_{i=1}^{I}(AST_i x_{i,t} + ST_i \theta_{i,t} + ASR_i \bar{x}_{i,t} + SR_i \bar{\theta}_{i,t}) \leq ACAP_t \tag{3.12}$$

$$\bar{x}_{i,t} \leq UR_i \sum_{j=1}^{J} B_{i,j} d_{j,t} \tag{3.13}$$

$$\sum_{j=1}^{J} B^{n}_{i,j} d^{n}_{j,t} \leq x_{i,t} \tag{3.14}$$

$$\sum_{j=1}^{J} B^{R}_{i,j} d^{R}_{j,t} \leq \bar{x}_{i,t} \tag{3.15}$$

$$e_{i,0} = \bar{e}_{i,0} = f_{j,0} = f^{n}_{j,0} = f^{R}_{j,0} = 0 \tag{3.16}$$

$$\bar{x}_{i,t}, \bar{e}_{i,t}, x_{i,t}, e_{i,t}, f_{j,t}, f^{n}_{j,t}, f^{R}_{j,t}, d_{j,t}, d^{n}_{j,t}, d^{R}_{j,t}, r_{j,t} \geq 0 \tag{3.17}$$

$$\theta_{i,t}, \bar{\theta}_{i,t}, \delta_{j,t}, \delta^{n}_{j,t}, \delta^{R}_{j,t} = \{0,1\} \tag{3.18}$$

Constraints (3.2) and (3.3) present relationship of production, inventory, setup and demand for new items. Constraints (3.4) and (3.5) present similar relationship for remanufactured items. Constraints (3.6), (3.7) and (3.8) present the relationship between setups and assembly (disassembly) of products (returned products). Constraints (3.9), (3.10) and (3.11) are inventory equations for returned and newly assembled products. Constraint (3.12) introduces a time capacity limitation for the time required for manufacturing and remanufacturing. This constraint is the common resource of the system. Constraint (3.13) represents the relationship between the number of remanufactured items and the number of recovered items from the returned products. Constraints (3.14) and (3.15) ensure that the number of new parts and remanufactured parts should always meet the demands for new products. Constraint (3.16) initialises inventories to zero at the beginning of the planning horizon. Constraint (3.17) is the non-negativity constraint and constraint (3.18) indicates binary decision variables.

# Chapter Four

# Solution Approach and Methodology

The developed model shown in chapter 3 is difficult to solve due to large number of integer variables involved. This chapter presents a solution method based on relaxation. However, the relaxation must be performed in a way that the applicability of the model is not destroyed. Details of the developed solution approach to solve the mathematical model are given in the following sections.

## 4.1 Model Relaxation

Complex constraints can be relaxed with the help of Lagrangian relaxation. The relaxed model can then be decomposed to different smaller sub-problems. These sub-problems are much simpler to solve by different methods such as Wagner-Whitin method. Sub-gradient method is used to improve Lagrangian multiplier in each iteration to find optimal or near-optimal solution of the original problem, following a standard procedure.

### 4.1.1 Lagrangian Relaxation

The specific procedure of using Lagrangian relaxation follows the steps given in Fisher (1985). In general, the idea of Lagrangian relaxation is to relax those constraints which make it hard and time consuming to solve the problem. Relaxed constraints will be added to the objective function associated with certain weights (the Lagrangian multipliers). Each Lagrangian multiplier can be considered as a penalty added to the solution not satisfying the corresponding constraint.

Mathematical presentation of the general Lagrangian relaxation can is presented below. Consider a minimization integer programming problem with $A$, $C$ and $D$ being parameter matrices. $b$ and $d$ are right hand side vectors.

$$Z_{IP} = \min C^T X$$

Subject to:

$$AX \geq b$$

$$DX \geq d$$

$$X \ are \ integers$$

Let $X := \{x \ integral | DX \geq d\}$, assuming optimization over $X$ can be done easily. After adding constraint $AX \geq b$ the problem becomes very difficult to solve. Using Lagrangian relaxation, dual variables $\lambda \geq 0$ will be used for constraints $AX \geq b$. The vector $\lambda \geq 0$ is the vector of Lagrangian multipliers of the same dimension as vector $b$. For a fixed $\lambda \geq 0$, the relaxed problem is:

$$Z(\lambda) \coloneqq \min C^T X + \lambda^T (b - AX)$$

Subject to:

$$DX \geq d, \quad X \ are \ integers$$

If the optimal solution of the relaxed problem with fixed vector λ can be found relatively easier, the value of $Z(\lambda)$ will present a upper bound on $Z_{IP}$.

### 4.1.2 Sub-gradient Method

Sub-gradient method is a technique to set the dual variables (Lagrangian multipliers) at different values in obtaining tighter bounds. Assuming that $AX \geq b$ are resource constraints and have been relaxed. Beginning with an arbitrary value of λ, we can find a solution of the relaxed problem. If we substitute the obtained solutions into the objective function of the Lagrangian problem, we have a linear objective function of λ. We should maximize the Lagrangian function in terms of λ. The sub-gradient method starts from an arbitrary set of alternative optimal Lagrangian solution and uses the vector $AX - b$ for this solution to calculate the sub-gradient of $Z(\lambda)$. The result is a procedure that defines a sequence of values for λ by beginning at an initial point $\lambda^0$ and applying the formula:

$$\lambda^{k+1} = \max\{\, 0, \lambda^k - t_k(\, Ax^k - b\,)\} \tag{4.1}$$

In this formula $t_k$ is a scalar stepsize and $x^k$ is an optimal solution to the Lagrangian problem with dual variables set to $\lambda^k$. The following formula calculates $t_k$ :

$$t_k = \frac{\mathrm{u}^k(Z^* - Z(\lambda^k))}{\sum_{i=1}^{m}(\sum_{j=1}^{n} a_{ij}x_j^k - b_i)} \tag{4.2}$$

$Z^*$ is the objective function value of the best known feasible solution to $Z_{IP}$ (known as upper bound in minimization problem) and $\mathrm{u}^k$ is a scalar chosen between 0 and 2. Frequently the sequence $\mathrm{u}^k$ is determined by starting with $\mathrm{u}^k = 2$ and reducing it by a factor of 2 whenever $Z(\lambda^k)$ fails to increase in a specific number of iterations. Value $Z^*$ can initially be set to 0 and then updated using the solution that is obtained on those iterations in which the Lagrangian problem solution turns out to be feasible in the original problem. Unless we obtain a $\lambda^k$ for which $Z(\lambda^k)$ =$Z^*$, we cannot prove the optimality in the sub-gradient method. In this case the search process usually terminates upon reaching specified number of iterations.

## 4.2 Application of Lagrangian Relaxation in Solving HMRS Model

We apply Lagrangian relaxation to solving the model developed and presented in Chapter 3.

### 4.2.1 Lagrangian Relaxation

The model presented in Chapter 3 has four sets of complex constraints, Eqs (3-12), (3-13), (3-14) and (3-15). These constraints make solving the problem intractable. Applying Lagrangian relaxation, 4 sets of Lagrangian multipliers, λ, γ, ϕ and ψ are introduced where λ represents the Lagrangian multiplier assigned to constraint (3.12), γ represents Lagrangian multiplier assigned to the constraint (3.13), ϕ represents the Lagrangian multiplier assigned to constraint (3.14) and finally ψ represents the Lagrangian multiplier assigned to constraint (3.15). We move these constraints to the objective function with associated Lagrangian multipliers to obtain the following relaxed model:

$$\boldsymbol{Min\ z(\lambda, \gamma, \phi, \psi)}$$

$$\begin{aligned}
&= \sum_{t=1}^{T}\sum_{i=1}^{I}\left(P_{i,t}x_{i,t} + S_{i,t}\theta_{i,t} + V_{i,t}e_{i,t} + \overline{P}_{i,t}\overline{x}_{i,t} + \overline{S}_{i,t}\overline{\theta}_{i,t} + \overline{V}_{i,t}\overline{e}_{i,t}\right) \\
&+ \sum_{t=1}^{T}\sum_{j=1}^{J}\Bigg(AQ_{j,t}r_{j,t} + SD_{j,t}\delta_{j,t} + RD_{j,t}d_{j,t} + IN_{j,t}f_{j,t} + SD_{j,t}^{n}\delta_{j,t}^{n} + RD_{j,t}^{n}d_{j,t}^{n} + IN_{j,t}^{n}f_{j,t}^{n} \\
&+ SD_{j,t}^{R}\delta_{j,t}^{R} + RD_{j,t}^{R}d_{j,t}^{R} + IN_{j,t}^{R}f_{j,t}^{R}\Bigg) \\
&+ \sum_{t=1}^{T}\lambda_t\Big[\sum_{i=1}^{I}\left(AST_i x_{i,t} + ST_i\theta_{i,t} + ASR_i\overline{x}_{i,t} + SR_i\overline{\theta}_{i,t}\right) - ACAP_t\Big] \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{I}\gamma_{i,t}\Big(\overline{x}_{i,t} - UR_i\sum_{j=1}^{J}B_{i,j}d_{j,t}\Big) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{I}\phi_{i,t}\Big(\sum_{j=1}^{J}B_{i,j}^{n}d_{j,t}^{n} - x_{i,t}\Big) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{I}\psi_{i,t}\Big(\sum_{j=1}^{J}B_{i,j}^{R}d_{j,t}^{R} - \overline{x}_{i,t}\Big) \qquad (4.3)
\end{aligned}$$

Subject to:

$e_{i,t-1} + x_{i,t} - e_{i,t} = D_{i,t}$

$x_{i,t} \leq M\theta_{i,t}$

$\bar{e}_{i,t-1} + \bar{x}_{i,t} - \bar{e}_{i,t} = \bar{D}_{i,t}$

$\bar{x}_{i,t} \leq M\bar{\theta}_{i,t}$

$d_{j,t} \leq M\delta_{j,t}$

$d_{j,t}^{n} \leq M\delta_{j,t}^{n}$

$d_{j,t}^{R} \leq M\delta_{j,t}^{R}$

$f_{j,t} + d_{j,t} - f_{j,t-1} = r_{j,t}$

$f_{j,t}^{n} + d_{j,t}^{n} - f_{j,t-1}^{n} = r_{j,t}^{n}$

$f_{j,t}^{R} + d_{j,t}^{R} - f_{j,t-1}^{R} = r_{j,t}^{R}$

$e_{i,0} = \bar{e}_{i,0} = f_{j,0} = f_{j,0}^{n} = f_{j,0}^{R} = 0$

$\bar{x}_{i,t}, \bar{e}_{i,t}, x_{i,t}, e_{i,t}, f_{j,t}, f_{j,t}^{n}, f_{j,t}^{R}, d_{j,t}, d_{j,t}^{n}, d_{j,t}^{R}, r_{j,t} \geq 0$

$\theta_{i,t}, \bar{\theta}_{i,t}, \delta_{j,t}, \delta_{j,t}^{n}, \delta_{j,t}^{R} = \{0,1\}$

### 4.2.2 Decomposition Method

Above relaxed model can be decomposed to five sub-problems based on the variables and parameters for new components, remanufactured components, returned products, new products made out of new components and new products made out of remanufactured parts, respectively. The five sub-problems are discussed separately below.

### 4.2.3 Sub-Problems

Sub-problem 1 is a minimization problem representing all the decision variables and parameters contributing to manufacturing of new components including manufacturing cost, setup cost and inventory controls subject to constraints related to manufacturing, setup and inventory of new components.

SM1-i, i=1,…,I

$$\boldsymbol{MinZ_1(\lambda,\phi)} = \sum_{t=1}^{T}\left[\left(\boldsymbol{P_{i,t}} + \boldsymbol{\lambda_t AST_i} - \boldsymbol{\phi_{i,t}}\right)\boldsymbol{x_{i,t}} + \left(\boldsymbol{S_{i,t}} + \boldsymbol{\lambda_t ST_i}\right)\boldsymbol{\theta_{i,t}} + \boldsymbol{V_{i,t}e_{i,t}}\right] - \sum_{t=1}^{T}\boldsymbol{\lambda_t ACAP_t}$$

Subject to:

$$e_{i,t-1} + x_{i,t} - e_{i,t} = D_{i,t}$$

$$x_{i,t} \le M\theta_{i,t}$$

$$\boldsymbol{e_{i,0} = 0, x_{i,t}, e_{i,t} \ge 0}$$

$$\boldsymbol{\theta_{i,t} = \{0,1\}}$$

Sub-problem 2 corresponds to the parts to be remanufactured with related setup and setups and inventory variables:

SM2-i, i=1…I

$$MinZ_2(\lambda,\psi) = \sum_{t=1}^{T}\left[\left(\bar{P}_{i,t} + \lambda_t ASR_i - \psi_{i,t} + \gamma_{i,t}\right)\bar{x}_{i,t} + \left(\bar{S}_{i,t} + \lambda_t SR_i\right)\bar{\theta}_{i,t} + \bar{V}_{i,t}\bar{e}_{i,t}\right]$$

Subject to:

$$\bar{e}_{i,t-1} + \bar{x}_{i,t} - \bar{e}_{i,t} = \bar{D}_{i,t}$$

$$\bar{x}_{i,t} \le M\bar{\theta}_{i,t}$$

$\bar{\boldsymbol{e}}_{\boldsymbol{i,0}} = \mathbf{0}, \bar{\boldsymbol{x}}_{\boldsymbol{i,t}}, \bar{\boldsymbol{e}}_{\boldsymbol{i,t}} \geq \mathbf{0}$

$\bar{\boldsymbol{\theta}}_{\boldsymbol{i,t}} = \{\mathbf{0}, \mathbf{1}\}$

Sub-problem 3 determines the number of returned products to acquire, with related setups for disassembling and inventory requirement.

SM3-j, j=1…J

$$Min\ Z_3(\gamma) = \sum_{t=1}^{T} AQ_{j,t}r_{j,t} + SD_{j,t}\delta_{j,t} + \left(RD_{j,t} - \sum_{i=1}^{I} B_{i,j}\gamma_{i,t}\, UR_i\right)d_{j,t} + IN_{j,t}f_{j,t}$$

Subject to:

$f_{j,t} + d_{j,t} - f_{j,t-1} = r_{j,t}$

$d_{j,t} \leq M\delta_{j,t}$

$\boldsymbol{f}_{\boldsymbol{j,0}} = \mathbf{0}, \boldsymbol{f}_{\boldsymbol{j,t}}, \boldsymbol{r}_{\boldsymbol{j,t}}, \boldsymbol{d}_{\boldsymbol{j,t}} \geq \mathbf{0}$

$\delta_{i,t} = \{0,1\}$

Sub-problem 4 specifies similar relations for new product production with new items.

SM4-j, j=1…J

$$Min\ Z_4(\phi) = \sum_{t=1}^{T} SD_{j,t}^{n}\delta_{j,t}^{n} + \left(RD_{j,t}^{n} + \sum_{i=1}^{I} B_{i,j}^{n}\, \phi_{i,t}\right)d_{j,t}^{n} + IN_{j,t}^{n}f_{j,t}^{n}$$

Subject to:

$f_{j,t}^{n} + d_{j,t}^{n} - f_{j,t-1}^{n} = r_{j,t}^{n}$

$d_{j,t}^{n} \leq M\delta_{j,t}^{n}$

$f_{j,0}^{n} = 0, f_{j,t}^{n}, d_{j,t}^{n} \geq 0$

$\delta_{j,t}^{n} = \{0,1\}$

Sub-problem 5 determines the optimal number of new products with remanufactured items to be assembled, considering assembly setup and inventory costs.

SM5-j, j=1…J

$$Z_5(\psi) = \sum_{t=1}^{T} SD_{j,t}^{R}\delta_{j,t}^{R} + \left(RD_{j,t}^{R} + \sum_{i=1}^{I} B_{i,j}^{R}\,\psi_{i,t}\right) d_{j,t}^{R} + IN_{j,t}^{R} f_{j,t}^{R}$$

Subject to:

$f_{j,t}^{R} + d_{j,t}^{R} - f_{j,t-1}^{R} = r_{j,t}^{R}$

$d_{j,t}^{R} \leq M\delta_{j,t}^{R}$

$f_{j,0}^{R} = 0, f_{j,t}^{R}, d_{j,t}^{R} \geq 0$

$\delta_{j,t}^{R} = \{0,1\}$

## 4.3 Solving the Sub-Problems

The sub-problems presented in the previous section are much easier to solve than the original problem. Four of them can be solved by existing method. Sub-problem 3 requires developing new method to solve with details discussed next.

### 4.3.1 Wagner-Whitin

Linear presentation of the Wagner-Whitin method gives a format similar to solving sub-problems 1, 2, 4 and 5 discussed in previous section. With minor changes we can effectively solve these corresponding sub-problems using Wagner-Whitin method. The general Wagner-Whitin method can be discussed below.

Assuming a N-period planning horizon with known demands $D_t$, fixed setup cost $A_t$, unit production cost $P_t$ and holding cost $H_t$, the following dynamic lot sizing and scheduling problems can be solved by Wagner-Whitin algorithm.

To solve the problem:

$$Min\, Z = P_tQ_t + A_t\theta_t + H_tI_t$$

Subject to:

$$I_t = I_{t-1} + Q_t - D_t \tag{4.4}$$

$$I_{t-1}Q_t = 0 \tag{4.5}$$

$$Q_t \leq M\theta_t \tag{4.6}$$

$$I_0 = 0\, , I_t, Q_t \geq 0 \tag{4.7}$$

Where $I_t$ is the inventory level at time period *t,* $Q_t$ is the number of products to be manufactured at *t*, $\theta_t$ is the binary variables to allocate the setup cost of production at *t.* Constraint Eq (4.4) is the inventory balance equation. Wagner-Whitin algorithm will force zero inventories at the time of production as shown in Eq (4.5). Eq (4.6) is the setup requirement for manufacturing.

We follow the steps of Wagner-Whitin algorithm in solving the dynamic lot sizing problems, as presented in sub-problems 1, 2, 4 and 5. Detailed steps of Wagner-Whitin method can be found in Evans (1985) and are omitted in this thesis.

### 4.3.2 Dynamic Programming

Sub-problem 3 presented in section 4.2.3 has the same linear functions as other four sub-problems. However, it does not have fixed demand and cannot be solved by Wagner-Whitin algorithm. A heuristic method based on dynamic programing is developed to solve this sub-problem.

As one can see, the number of remanufactured parts required in each period can be obtained by solving sub-problem 2. This solution can be used in constraint (3.13) to obtain an estimated number of the returned products. We will acquire the product when the cost of purchasing is less. Our algorithm also implies the same inventory role as Wagner-Within in that whenever there is an order to be placed in a certain period, inventory at the beginning of that period must be zero. Specific steps of calculation are given below:

**Step 1.** Let $\tilde{\tilde{x}}_{i,t}$ be the current solution of Sub-problem 2. Let $\tilde{d}_{j,t}$ be the minimum values

satisfying $\tilde{\tilde{x}}_{i,t} \le UR_i \sum_{j=1}^{J} B_{i,j}\tilde{d}_{j,t}$. Define $\tilde{R}_{j,t} = RD_{j,t} - (\sum_{i=1}^{I} B_{i,j}\gamma_{i,t}UR_i)$. Let $\tilde{A}_{j,1} = AQ_{j,1}$.

Let $d_{j,1} = r_{j,1} = \tilde{d}_{j,1}$, $\delta_{j,1} = 1$ and let $t = 2$ Goto Step 2.

**Step 2.** Recursively calculate: $\tilde{A}_{j,t} = \min\{AQ_{j,t}, IN_{j,t-1} + \tilde{A}_{j,t-1}\}$.

Step 2.1. If $\tilde{R}_{j,t} \ge -(\tilde{A}_{j,t} \times \tilde{d}_{j,t} + SD_{j,t})$ let $d_{j,t} = \delta_{j,t} = r_{j,t} = f_{j,t} = 0$, Goto Step 3;

otherwise, let $\delta_{j,t} = 1$ and $d_{j,t} = \tilde{d}_{j,t}$, Goto Step 2.2.

Step 2.2. If: $AQ_{j,t} \le IN_{j,t-1} + \widetilde{A}_{j,t-1}$, let $r_{j,t} = \widetilde{d}_{j,t}$, Goto Step 3.

Otherwise, let $s$ be the last time period when $r_{j,t} \ne 0$ and let

$$r_{j,s} = r_{j,s} + \widetilde{d}_{j,t}$$

$$f_{j,k} = f_{j,k} + \widetilde{d}_{j,t}, k = s,...,t,$$

$$d_{j,t} = \delta_{j,t} = 0$$

Goto Step 3.

**Step 3.** If $t = T$, stop; otherwise, let $t = t + 1$, Goto Step 2.

## 4.4 Complete Solution Procedure to Solve the HMRS Problem

The complete procedure of Lagrangian decomposition and sub-gradient method to solve the HMRS production planning problem is provided below.

**Step 1.** Find a feasible solution of the original problem given in equation number (3.1) using any optimization software or by decomposition.

**Step 2.** Use the objective function of the current feasible solution to calculate the upper bound ( UB) for solving the original problem.

**Step 3.** Set iteration number It=1. Starting from an arbitrary set of Lagrangian multipliers $\lambda, \gamma, \phi$ and $\psi$ apply Lagrangian relaxation to the original main model as described in Section 4.2.1.

**Step 4.** Decompose the original problem to the 5 sub-problems as described in Section 4.2.3.

**Step 5.** Solve the 5 sub-problems as discussed in Section 4.3.

**Step 6.** Compose the solutions of the 5 sub-problems obtained in Step 5 and calculate the value of $Min\ Z(\lambda, \gamma, \phi, \psi)$ shown in Eq (4.3).

**Step 7.** If the solution composed from the solutions of the 5 sub-problems obtained in step 5 is feasible to the original problem and the corresponding value of the original objective function Eq (3.1) is less than the UB ($Z <$ UB), then let $UB = Z$ and go to step 8. Otherwise, go to step 8 without updating UB.

**Step 8.** If the stopping criterion is reached, stop. Otherwise, let It=It+1 and update Lagrangian multipliers based on sub-gradient search method shown in Eq (4.1) and Eq (4.2) Go to Step 4.

**Step 9.** When the search process stops, take the best feasible solution found in the process as the final solution of the problem.

The algorithm has been coded in Python 2.6 and implemented in a PC computer. Several example problems were calculated to test the model and solution method it will be discussed in the next chapter.

# Chapter Five

# Numerical Example and Analysis

In this chapter we present several numerical examples to illustrate the developed model and the algorithm presented in previous chapters and following the solution procedure presented in Section 4.4. The algorithm is coded in Python-2.6 to solve the example problems. Some of the smaller size examples were solved to optimality by Lingo 10. All the computer work was performed on a laptop with Intel Core i5 CPU and 4 GB RAM.

## 5.1 Example Problems and Data

The first example, Example 1, is to solve a production planning problem with 3 time periods. The considered HMRS system produces 4 different types of components from new materials or from returned products. The cores to be remanufactured are from disassembly of 3 different types of returned products. Remanufactured and manufactured components are assembled to make 3 different products. Tables 5.1 and 5.2 present the costs and demands of new and remanufactured components, respectively. Table 5.3 presents similar data for the returned products.  Tables 5.4 and 5.5 give the cost and demands for new products with new components and with remanufactured components. The numbers and types of components contained in each of the products are shown in Tables 5.6, 5.7 and 5.8. Table 5.9 gives the time requirements to produce the new and remanufactured components requiring the shared resources. It also gives the empirical ratio of each type component that is of a good quality and can be remanufactured, out of the total acquired returned products. Table 5.10 is the available time for setting up the system and producing the components by the shared resource in the 3 time periods.

**Table 5.1: Cost and Demand for Component from New Material [Example 1]**

| Time Period | Cost and Demand | Component Type | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Production Cost/item | 100 | 90 | 110 | 95 |
| | Setup Cost | 600 | 700 | 600 | 550 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 |
| | Demand | 180 | 270 | 235 | 250 |
| 2 | Production Cost/item | 80 | 100 | 80 | 80 |
| | Setup Cost | 600 | 700 | 600 | 550 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 |
| | Demand | 200 | 305 | 260 | 260 |
| 3 | Production Cost/item | 110 | 80 | 95 | 70 |
| | Setup Cost | 600 | 700 | 600 | 550 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 |
| | Demand | 225 | 320 | 275 | 265 |

**Table 5.2: Cost and Demand for Component to be Remanufactured [Example 1]**

| Time Period | Cost and Demand | Component Type | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Production Cost/item | 30 | 40 | 20 | 15 |
| | Setup Cost | 400 | 500 | 450 | 350 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 |
| | Demand | 80 | 88 | 83 | 80 |
| 2 | Production Cost/item | 30 | 40 | 15 | 30 |
| | Setup Cost | 400 | 500 | 450 | 350 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 |
| | Demand | 95 | 96 | 100 | 95 |
| 3 | Production Cost/item | 30 | 30 | 18 | 25 |
| | Setup Cost | 400 | 500 | 450 | 350 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 |
| | Demand | 75 | 85 | 80 | 81 |

**Table 5.3: Cost Corresponds to the Returned Products [Example 1]**

| Product Type | Cost | Time Periods | | |
|---|---|---|---|---|
| | | 1 | 2 | 3 |
| 1 | Disassembly /item | 30 | 35 | 20 |
| | Setup | 22 | 30 | 33 |
| | Inventory/item | 240 | 240 | 240 |
| | Acquiring/item | 25 | 15 | 20 |
| 2 | Disassembly /item | 25 | 30 | 15 |
| | Setup | 35 | 25 | 35 |
| | Inventory/item | 250 | 250 | 250 |
| | Acquiring/item | 35 | 20 | 30 |
| 3 | Disassembly /item | 20 | 18 | 30 |
| | Setup | 30 | 28 | 30 |
| | Inventory/item | 230 | 230 | 230 |
| | Acquiring/item | 25 | 28 | 30 |

**Table 5.4: Cost and Demand for New Products of New Components [Example 1]**

| Product Type | Cost | Time Periods | | |
|---|---|---|---|---|
| | | 1 | 2 | 3 |
| 1 | Assembly/item | 32 | 35 | 22 |
| | Setup | 26 | 31 | 34 |
| | Inventory/item | 100 | 100 | 100 |
| | Demand | 10 | 15 | 16 |
| 2 | Assembly/item | 30 | 31 | 18 |
| | Setup | 38 | 26 | 36 |
| | Inventory/item | 90 | 90 | 90 |
| | Demand | 13 | 14 | 16 |
| 3 | Assembly/item | 22 | 24 | 28 |
| | Setup | 32 | 30 | 31 |
| | Inventory/item | 95 | 95 | 95 |
| | Demand | 15 | 14 | 13 |

**Table 5.5: Cost and Demand for New Products of Remanuf. Components [Example 1]**

| Product Type | Cost | Time Periods | | |
|---|---|---|---|---|
| | | 1 | 2 | 3 |
| 1 | Assembly/item | 15 | 12 | 10 |
| | Setup | 15 | 18 | 20 |
| | Inventory/item | 90 | 90 | 90 |
| | Demand | 6 | 5 | 7 |
| 2 | Assembly/item | 16 | 16 | 11 |
| | Setup | 26 | 20 | 22 |
| | Inventory/item | 60 | 60 | 60 |
| | Demand | 15 | 22 | 20 |
| 3 | Assembly/item | 14 | 11 | 12 |
| | Setup | 22 | 20 | 25 |
| | Inventory/item | 85 | 85 | 85 |
| | Demand | 5 | 6 | 7 |

**Table 5.6: Number of Parts in the Returned Products [Example 1]**

| Component / Type Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 10 | 10 | 8 | 13 |
| 2 | 12 | 12 | 10 | 12 |
| 3 | 15 | 11 | 3 | 11 |

**Table 5.7: Number of New Parts in the Assembled New Products [Example 1]**

| Component / Type Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 5 | 7 | 4 | 3 |
| 2 | 6 | 5 | 7 | 6 |
| 3 | 2 | 4 | 5 | 7 |

**Table 5.8: Number of Remanuf. Parts in the Assembled New Products [Example 1]**

| Component Type / Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 4 | 3 | 3 | 4 |
| 2 | 3 | 5 | 4 | 4 |
| 3 | 4 | 5 | 2 | 3 |

**Table 5.9: Resource Time Required and Quality Ratio [Example 1]**

| | Component Type | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| New Component Production Time | 100 | 150 | 100 | 120 |
| New Component Setup Time | 50 | 50 | 40 | 30 |
| Remanuf. Component Production Time | 80 | 80 | 75 | 80 |
| Remanuf. Component Setup Time | 30 | 30 | 35 | 30 |
| Quality Ratio | 0.5 | 0.5 | 0.6 | 0.2 |

**Table 5.10: Resource Availability [Example 1]**

| Time Period | 1 | 2 | 3 |
|---|---|---|---|
| Resource Available Time | 600000 | 150000 | 250000 |

## 5.2 Computational Results and Analysis

In this section we will discuss the results obtained from solving the Example 1. In calculating this example, we used two stopping criteria. The process will stop if the Lagrangian search reaches 300 iterations (It=300) or the difference between the value of Lagrangian function in Eq (4.3) and value of the original objective function in Eq (3.1) is less than 0.01 ($Z^* - Z\_l \leq 0.01$). As discussed in the previous chapter, in each iteration, the Lagrangian multipliers $\lambda, \gamma, \phi$ and ψ will be updated so is the value of the upper bound, if required.

Corresponding input data for the first example has been shown in Tables 5.1 to 5.10. The initial upper bound, the value of the first feasible solution to the original objective function Eq (3.1), was set at 334291. This value will be used in the beginning of the sub-gradient method to update Lagrangian multipliers. The value of the first set of the Lagrangian multipliers was set at 0.1. They got updated at every iteration following the described sub-gradient method in Section (4.1.2). In the process of updating the Lagrangian multiplier using Eqs (4.1) and (4.2), the scalar $u^k$ is reduced by a factor of 2.0 after every 5 consecutive iterations that the Lagrangian function fails to increase. Upper bound stays the same the search process if it cannot find a better feasible solution to substitute for the current upper bound. The convergence of the Lagrangian function $Z(\lambda, \gamma, \phi, \psi)$ obtained from Eq (4.3) can be seen in Figure 5.1 by the solid line. Figure 5.2 presents the closer capture of the convergence of the Lagrangian function. The search process took 12.47 seconds to find the best solution in 300 iterations. Corresponding solution is near-optimal and is quite close to the upper bound.

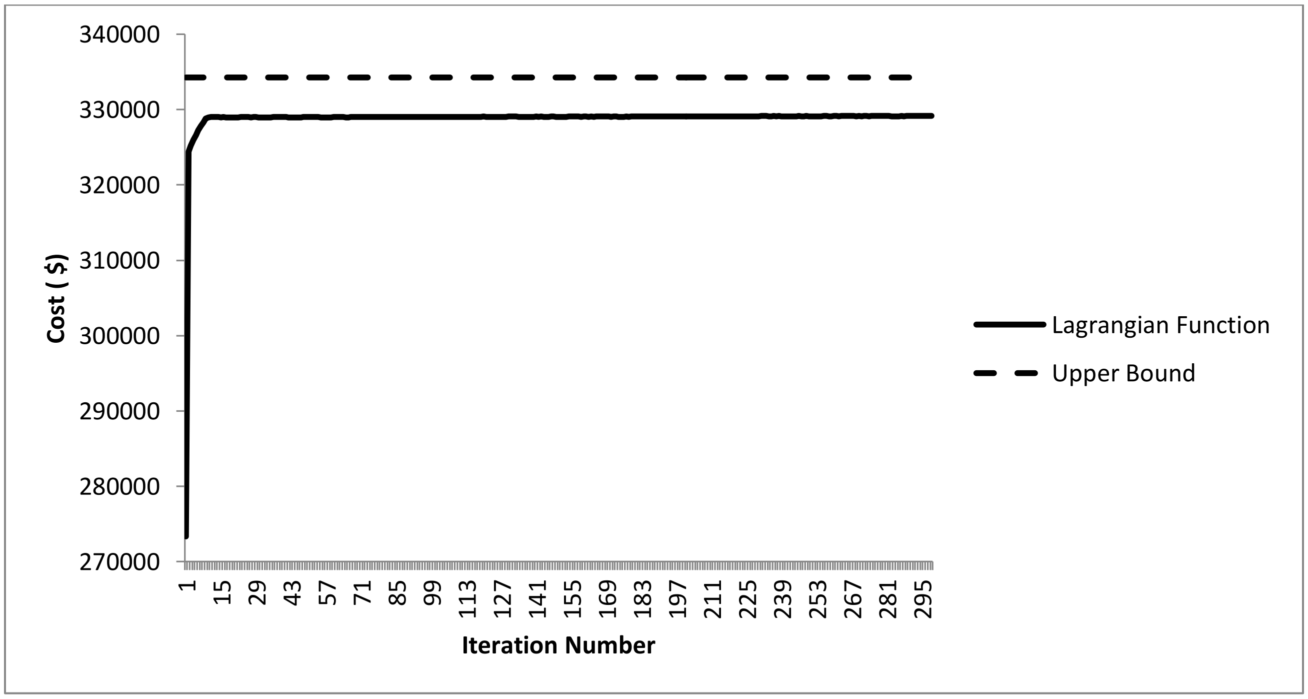

Figure 5.1.Convergence Behaviour of the Lagrangian Function [Example 1]

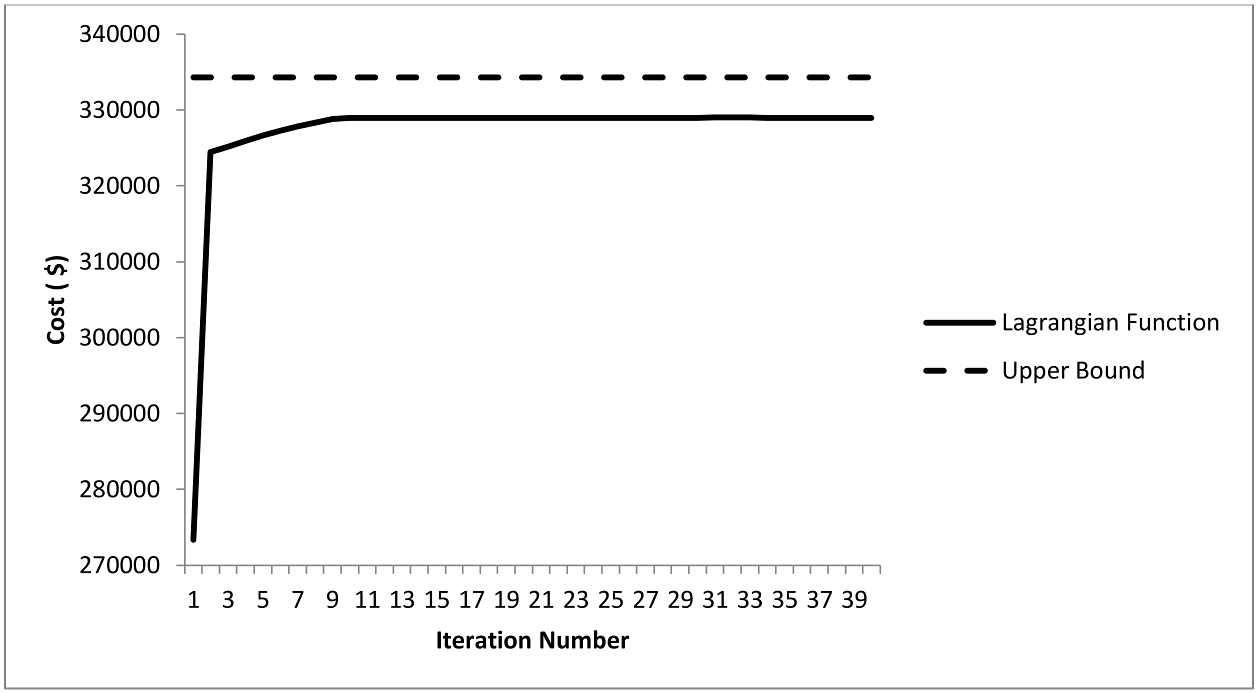

Figure 5.2.Behaviour of the Lagrangian Function in Close Capture [Example 1]

The value of the original objective function (total cost), after 300 iterations is 328471. The best solutions are presented in Tables 5.11 and 5.12 with specific values of the decision variables. The first four rows in the Table 5.11 present the values associated with new manufactured components. The second four rows present the values associated with the remanufactured components. Respectively, data for the inventory of new components, inventory of remanufactured components and finally setups for the new and remanufactured components are presented. Table 5.12 presents the values for inventory of the returned products, number of returned products to acquire, number of returned products to disassemble, setups correspond to disassembling, inventory of new and remanufactured products, number of new and remanufactured products to assemble and finally setups associate with assembly of new and remanufactured products. Table 5.13 presents the values to the Lagrangian multipliers when the search process stopped.

**Table 5.11: Results Corresponding to the Components [Example 1]**

| Decision Variables | Period / Component | 1 | 2 | 3 |
|---|---|---|---|---|
| $x$ | 1 | 180 | 200 | 225 |
| | 2 | 270 | 305 | 320 |
| | 3 | 235 | 260 | 275 |
| | 4 | 250 | 260 | 265 |
| $\bar{x}$ | 1 | 80 | 95 | 75 |
| | 2 | 88 | 96 | 85 |
| | 3 | 83 | 100 | 80 |
| | 4 | 80 | 95 | 81 |
| $e$ | 1 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| | 4 | 0 | 0 | 0 |
| $\bar{e}$ | 1 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| | 4 | 0 | 0 | 0 |
| $\theta$ | 1 | 1 | 1 | 1 |
| | 2 | 1 | 1 | 1 |
| | 3 | 1 | 1 | 1 |
| | 4 | 1 | 1 | 1 |
| $\bar{\theta}$ | 1 | 1 | 1 | 1 |
| | 2 | 1 | 1 | 1 |
| | 3 | 1 | 1 | 1 |
| | 4 | 1 | 1 | 1 |

**Table 5.12: Results Corresponds to the Products [Example 1]**

| Decision Variables | Period / Product | 1 | 2 | 3 |
|---|---|---|---|---|
| $f$ | 1 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $r$ | 1 | 31 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $d$ | 1 | 31 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $\delta$ | 1 | 1 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $f^n$ | 1 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $f^R$ | 1 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 0 |
| | 3 | 0 | 0 | 0 |
| $d^n$ | 1 | 10 | 15 | 16 |
| | 2 | 13 | 14 | 16 |
| | 3 | 15 | 14 | 13 |
| $d^R$ | 1 | 6 | 5 | 7 |
| | 2 | 6 | 7 | 5 |
| | 3 | 5 | 6 | 7 |
| $\delta^n$ | 1 | 1 | 1 | 1 |
| | 2 | 1 | 1 | 1 |
| | 3 | 1 | 1 | 1 |
| $\delta^R$ | 1 | 1 | 1 | 1 |
| | 2 | 1 | 1 | 1 |
| | 3 | 1 | 1 | 1 |

**Table 5.13: Lagrangian Multipliers [Example 1]**

| Decision Variables | Period / Product | 1 | 2 | 3 |
|---|---|---|---|---|
| $\gamma$ | 1 | 0.09992 | 0.10009 | 0.10007 |
| | 2 | 0.09993 | 0.10009 | 0.10008 |
| | 3 | 0.09993 | 0.10010 | 0.10008 |
| | 4 | 0.09999 | 0.10009 | 0.10008 |
| $\phi$ | 1 | 0.10002 | 0.10001 | 0.10002 |
| | 2 | 0.10007 | 0.10007 | 0.10007 |
| | 3 | 0.10002 | 0.10003 | 0.10003 |
| | 4 | 0.10003 | 0.10003 | 0.10003 |
| $\psi$ | 1 | 0.10001 | 0.10003 | 0.10000 |
| | 2 | 0.10001 | 0.10001 | 0.10000 |
| | 3 | 0.10003 | 0.10004 | 0.10001 |
| | 4 | 0.10001 | 0.10002 | 0.10001 |
| $\lambda$ | $t$ | 0.00000 | 0.10365 | 0.00459 |

## 5.3 Other Examples

Several other example problems were used to test model and solution method developed in this research. Some features for two of them are introduced below. Computations of these examples follow the same stopping criteria as in Example 1.

Example 2 is a HMRS production planning problem with 5 time periods, 4 products and 5 components, other information and data are presented in Tables 5.14 to 5.23.

**Table 5.14: Cost and Demand for Component from New Material [Example 2]**

| Time Period | Cost and Demand | Component Type | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 1 | Production Cost/item | 100 | 90 | 110 | 95 | 85 |
| | Setup Cost | 600 | 700 | 600 | 550 | 600 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 | 100 |
| | Demand | 240 | 370 | 285 | 250 | 300 |
| 2 | Production Cost/item | 80 | 100 | 80 | 80 | 100 |
| | Setup Cost | 600 | 700 | 600 | 550 | 600 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 | 100 |
| | Demand | 200 | 300 | 230 | 212 | 240 |
| 3 | Production Cost/item | 90 | 90 | 95 | 100 | 100 |
| | Setup Cost | 600 | 700 | 600 | 550 | 600 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 | 100 |
| | Demand | 210 | 345 | 260 | 205 | 280 |
| 4 | Production Cost/item | 90 | 85 | 70 | 80 | 90 |
| | Setup Cost | 600 | 700 | 600 | 550 | 600 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 | 100 |
| | Demand | 220 | 385 | 310 | 260 | 370 |
| 5 | Production Cost/item | 75 | 110 | 85 | 80 | 110 |
| | Setup Cost | 600 | 700 | 600 | 550 | 600 |
| | Inventory Cost/item | 100 | 120 | 105 | 115 | 100 |
| | Demand | 140 | 235 | 162 | 170 | 120 |

**Table 5.15: Cost and Demand for Component to be Remanufactured [Example 2]**

| Time Period | Cost and Demand | Component Type | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 1 | Production Cost/item | 30 | 40 | 20 | 15 | 10 |
| | Setup Cost | 400 | 500 | 450 | 350 | 400 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 | 65 |
| | Demand | 130 | 150 | 155 | 155 | 135 |
| 2 | Production Cost/item | 20 | 50 | 15 | 30 | 25 |
| | Setup Cost | 400 | 500 | 450 | 350 | 400 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 | 65 |
| | Demand | 126 | 142 | 150 | 155 | 130 |
| 3 | Production Cost/item | 40 | 30 | 18 | 25 | 15 |
| | Setup Cost | 400 | 500 | 450 | 350 | 400 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 | 65 |
| | Demand | 119 | 120 | 135 | 135 | 120 |
| 4 | Production Cost/item | 15 | 30 | 18 | 22 | 15 |
| | Setup Cost | 400 | 500 | 450 | 350 | 400 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 | 65 |
| | Demand | 90 | 100 | 100 | 95 | 84 |
| 5 | Production Cost/item | 20 | 55 | 20 | 32 | 20 |
| | Setup Cost | 400 | 500 | 450 | 350 | 400 |
| | Inventory Cost/item | 50 | 80 | 60 | 75 | 65 |
| | Demand | 114 | 125 | 130 | 140 | 114 |

**Table 5.16: Cost Corresponds to the Returned Products [Example 2]**

| Product Type | Cost | Time Periods | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 1 | Disassembly/item | 30 | 35 | 20 | 15 | 27 |
| | Setup | 22 | 30 | 33 | 22 | 33 |
| | Inventory/item | 40 | 40 | 40 | 40 | 40 |
| | Acquiring/item | 25 | 15 | 20 | 21 | 18 |
| 2 | Disassembly/item | 25 | 30 | 15 | 22 | 28 |
| | Setup | 35 | 25 | 35 | 30 | 27 |
| | Inventory/item | 50 | 50 | 50 | 50 | 50 |
| | Acquiring/item | 35 | 20 | 30 | 30 | 22 |
| 3 | Disassembly/item | 20 | 18 | 30 | 18 | 14 |
| | Setup | 30 | 28 | 30 | 32 | 27 |
| | Inventory/item | 30 | 30 | 30 | 30 | 30 |
| | Acquiring/item | 25 | 28 | 30 | 28 | 30 |
| 4 | Disassembly/item | 17 | 22 | 26 | 21 | 19 |
| | Setup | 31 | 29 | 32 | 33 | 29 |
| | Inventory/item | 35 | 35 | 35 | 35 | 35 |
| | Acquiring/item | 22 | 26 | 31 | 20 | 32 |

**Table 5.17: Cost and Demand for New Products of New Components [Example 2]**

| Product Type | Cost | Time Periods | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 1 | Assembly/item | 32 | 35 | 22 | 20 | 29 |
| | Setup | 26 | 31 | 34 | 23 | 34 |
| | Inventory/item | 60 | 60 | 60 | 60 | 60 |
| | Demand | 20 | 15 | 20 | 14 | 17 |
| 2 | Assembly/item | 30 | 31 | 18 | 32 | 26 |
| | Setup | 38 | 26 | 36 | 31 | 28 |
| | Inventory/item | 70 | 70 | 70 | 70 | 70 |
| | Demand | 18 | 16 | 14 | 12 | 17 |
| 3 | Assembly/item | 22 | 24 | 28 | 26 | 22 |
| | Setup | 32 | 30 | 31 | 33 | 28 |
| | Inventory/item | 45 | 45 | 45 | 45 | 45 |
| | Demand | 16 | 11 | 16 | 12 | 14 |
| 4 | Assembly/item | 32 | 28 | 26 | 30 | 25 |
| | Setup | 30 | 28 | 30 | 30 | 29 |
| | Inventory/item | 65 | 65 | 65 | 65 | 65 |
| | Demand | 14 | 15 | 10 | 17 | 11 |

**Table 5.18: Cost and Demand for New Products of Remanuf. Components [Example 2]**

| Product Type | Cost | Time Periods | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| 1 | Assembly/item | 30 | 33 | 20 | 19 | 25 |
| | Setup | 25 | 31 | 34 | 23 | 34 |
| | Inventory/item | 55 | 55 | 55 | 55 | 55 |
| | Demand | 9 | 9 | 9 | 8 | 6 |
| 2 | Assembly/item | 30 | 28 | 18 | 22 | 24 |
| | Setup | 36 | 30 | 36 | 31 | 30 |
| | Inventory/item | 60 | 60 | 60 | 60 | 60 |
| | Demand | 7 | 4 | 8 | 5 | 6 |
| 3 | Assembly/item | 25 | 22 | 25 | 23 | 26 |
| | Setup | 32 | 30 | 32 | 35 | 30 |
| | Inventory/item | 44 | 44 | 44 | 44 | 44 |
| | Demand | 7 | 8 | 7 | 3 | 7 |
| 4 | Assembly/item | 28 | 30 | 25 | 27 | 26 |
| | Setup | 34 | 34 | 30 | 28 | 30 |
| | Inventory/item | 50 | 50 | 50 | 50 | 50 |
| | Demand | 9 | 10 | 5 | 4 | 9 |

**Table 5.19: Number of Parts in the Returned Products [Example 2]**

| Component / Type Product Type | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 10 | 10 | 8 | 13 | 8 |
| 2 | 12 | 12 | 10 | 12 | 15 |
| 3 | 15 | 11 | 3 | 11 | 2 |
| 4 | 11 | 15 | 9 | 10 | 13 |

**Table 5.20: Number of New Parts in the Assembled New Products [Example 2]**

| Component Type / Product Type | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 3 | 6 | 5 | 3 | 8 |
| 2 | 2 | 6 | 3 | 4 | 2 |
| 3 | 5 | 7 | 5 | 2 | 3 |
| 4 | 2 | 2 | 3 | 5 | 3 |

**Table 5.21: Number of Remanuf. Parts in the Assembled New Products [Example 2]**

| Component Type / Product Type | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 3 | 5 | 5 | 3 | 4 |
| 2 | 4 | 4 | 3 | 4 | 3 |
| 3 | 5 | 6 | 5 | 6 | 5 |
| 4 | 3 | 2 | 4 | 5 | 3 |

**Table 5.22: Resource Time Required and Quality Ratio [Example 2]**

| | Component Type | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| New Component Production Time | 120 | 150 | 130 | 120 | 110 |
| New Component Setup Time | 50 | 50 | 40 | 45 | 50 |
| Remanuf. Component Production Time | 80 | 80 | 75 | 75 | 75 |
| Remanuf. Component Setup Time | 30 | 30 | 30 | 30 | 30 |
| Quality Ratio | 0.5 | 0.5 | 0.6 | 0.4 | 0.7 |

**Table 5.23: Resource Availability [Example 2]**

| Time Period | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Resource Available Time | 250000 | 230000 | 230000 | 230000 | 230000 |

The computation results show convergence behavior of the procedure. Figure 5.3 presents the graph corresponds to the convergence of the Lagrangian function as well as the upper bound. In the first 80 iterations, the upper bound remained the same at 700926 since search process did not find a better feasible solution to substitute for upper bound. The first feasible solution obtained in iteration 81 causes the significant drop in the value of the upper bound. The value of the objective function for the original model (total cost) obtained from Eq (3.1) after 300 iterations is 697999 which corresponds to the feasible near optimal solutions. The computation took 61 seconds.

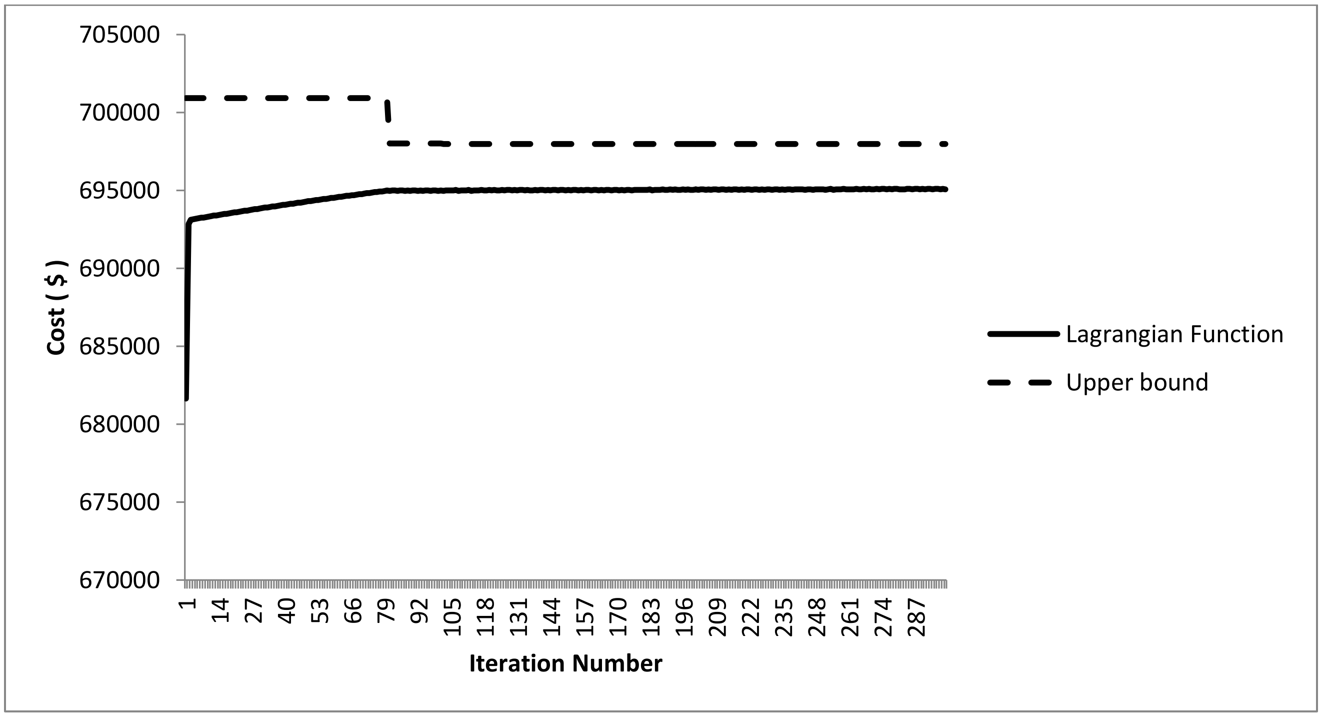

Figure 5.3.Convergence Behaviour of the Lagrangian Function [Example 2]

Example 3 is a HMRS production planning problem with 4 time periods, 3 products and 4 components, other information and data are presented in Tables 5.24 to 5.33.

**Table 5.24: Cost and Demand for Component from New Material [Example 3]**

| Time Period | Cost and Demand | Component Type | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Production Cost/item | 93 | 90 | 104 | 95 |
| | Setup Cost | 555 | 723 | 618 | 550 |
| | Inventory Cost/item | 105 | 112 | 115 | 105 |
| | Demand | 180 | 270 | 235 | 250 |
| 2 | Production Cost/item | 74 | 101 | 88 | 83 |
| | Setup Cost | 555 | 723 | 618 | 550 |
| | Inventory Cost/item | 105 | 112 | 115 | 105 |
| | Demand | 200 | 305 | 260 | 260 |
| 3 | Production Cost/item | 91 | 93 | 90 | 100 |
| | Setup Cost | 555 | 723 | 618 | 550 |
| | Inventory Cost/item | 105 | 112 | 115 | 105 |
| | Demand | 225 | 320 | 275 | 265 |
| 4 | Production Cost/item | 102 | 87 | 83 | 98 |
| | Setup Cost | 555 | 723 | 618 | 550 |
| | Inventory Cost/item | 105 | 112 | 115 | 105 |
| | Demand | 150 | 200 | 210 | 200 |

**Table 5.25: Cost and Demand for Component to be Remanufactured [Example 3]**

| Time Period | Cost and Demand | Component Type | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Production Cost/item | 28 | 41 | 22 | 15 |
| | Setup Cost | 420 | 512 | 468 | 421 |
| | Inventory Cost/item | 58 | 83 | 62 | 75 |
| | Demand | 80 | 88 | 83 | 80 |
| 2 | Production Cost/item | 28 | 40 | 15 | 30 |
| | Setup Cost | 420 | 512 | 468 | 421 |
| | Inventory Cost/item | 58 | 83 | 62 | 75 |
| | Demand | 95 | 96 | 100 | 95 |
| 3 | Production Cost/item | 29 | 33 | 18 | 25 |
| | Setup Cost | 420 | 512 | 468 | 421 |
| | Inventory Cost/item | 58 | 83 | 62 | 75 |
| | Demand | 75 | 85 | 80 | 81 |
| 4 | Production Cost/item | 30 | 35 | 18 | 30 |
| | Setup Cost | 420 | 512 | 468 | 421 |
| | Inventory Cost/item | 58 | 83 | 62 | 75 |
| | Demand | 80 | 85 | 82 | 80 |

**Table 5.26: Cost Corresponds to the Returned Products [Example 3]**

| Product Type | Cost | Time Periods | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Disassembly/item | 28 | 34 | 21 | 22 |
| | Setup | 23 | 31 | 33 | 35 |
| | Inventory/item | 180 | 180 | 180 | 180 |
| | Acquiring/item | 24 | 17 | 20 | 22 |
| 2 | Disassembly/item | 25 | 33 | 15 | 18 |
| | Setup | 35 | 25 | 34 | 32 |
| | Inventory/item | 220 | 220 | 220 | 220 |
| | Acquiring/item | 33 | 20 | 31 | 15 |
| 3 | Disassembly/item | 20 | 18 | 30 | 15 |
| | Setup | 31 | 27 | 30 | 31 |
| | Inventory/item | 200 | 200 | 200 | 200 |
| | Acquiring/item | 25 | 28 | 30 | 20 |

**Table 5.27: Cost and Demand for New Products of New Components [Example 3]**

| Product Type | Cost | Time Periods | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Assembly/item | 32 | 31 | 22 | 20 |
| | Setup | 28 | 31 | 34 | 28 |
| | Inventory/item | 103 | 103 | 103 | 103 |
| | Demand/item | 9 | 13 | 16 | 15 |
| 2 | Assembly/item | 28 | 31 | 18 | 22 |
| | Setup | 34 | 26 | 35 | 32 |
| | Inventory/item | 92 | 92 | 92 | 92 |
| | Demand/item | 12 | 14 | 15 | 14 |
| 3 | Assembly/item | 22 | 23 | 26 | 20 |
| | Setup | 32 | 30 | 31 | 28 |
| | Inventory/item | 95 | 95 | 95 | 95 |
| | Demand/item | 13 | 13 | 13 | 16 |

**Table 5.28: Cost and Demand for New Products of Remanuf. Components [Example 3]**

| Product Type | Cost | Time Periods | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| 1 | Assembly/item | 16 | 12 | 13 | 11 |
| | Setup | 16 | 18 | 21 | 20 |
| | Inventory/item | 93 | 93 | 93 | 93 |
| | Demand | 5 | 6 | 5 | 4 |
| 2 | Assembly/item | 13 | 15 | 11 | 14 |
| | Setup | 24 | 20 | 21 | 25 |
| | Inventory/item | 74 | 74 | 74 | 74 |
| | Demand | 6 | 5 | 5 | 5 |
| 3 | Assembly/item | 14 | 11 | 13 | 14 |
| | Setup | 28 | 19 | 23 | 22 |
| | Inventory/item | 85 | 85 | 85 | 85 |
| | Demand | 4 | 6 | 6 | 4 |

**Table 5.29: Number of Parts in the Returned Products [Example 3]**

| Component Type / Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 10 | 10 | 8 | 13 |
| 2 | 12 | 12 | 10 | 12 |
| 3 | 15 | 11 | 3 | 11 |

**Table 5.30 Number of New Parts in the Assembled New Products [Example 3]**

| Component Type / Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 5 | 7 | 4 | 3 |
| 2 | 6 | 5 | 7 | 6 |
| 3 | 2 | 4 | 5 | 7 |

**Table 5.31: Number of Remanuf. Parts in the Assembled New Products [Example 3]**

| Component Type / Product Type | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 4 |
| 2 | 3 | 3 | 4 | 2 |
| 3 | 4 | 1 | 2 | 3 |

**Table 5.32: Resource Time Required and Quality Ratio [Example 3]**

| | Component Type | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| New Component Production Time | 102 | 100 | 110 | 110 |
| New Component Setup Time | 52 | 48 | 43 | 44 |
| Remanuf. Component Production Time | 77 | 80 | 75 | 83 |
| Remanuf. Component Setup Time | 32 | 31 | 35 | 30 |
| Quality Ratio | 0.5 | 0.5 | 0.6 | 0.2 |

**Table 5.33: Resource Availability [Example 3]**

| Time Period | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Resource Available Time | 600000 | 200000 | 250000 | 200000 |

The computation results show convergence behavior of the procedure. Figure 5.4 presents the graph corresponds to the convergence of the Lagrangian function as well as the upper bound. Upper bound remained the same since search process did not find a better solution to substitute for upper bound. The value of the objective function for the original model (total cost) obtained from Eq (3.1) after 300 iterations is 412012 which corresponds to the feasible near optimal solutions. The computation took 12.92 seconds.

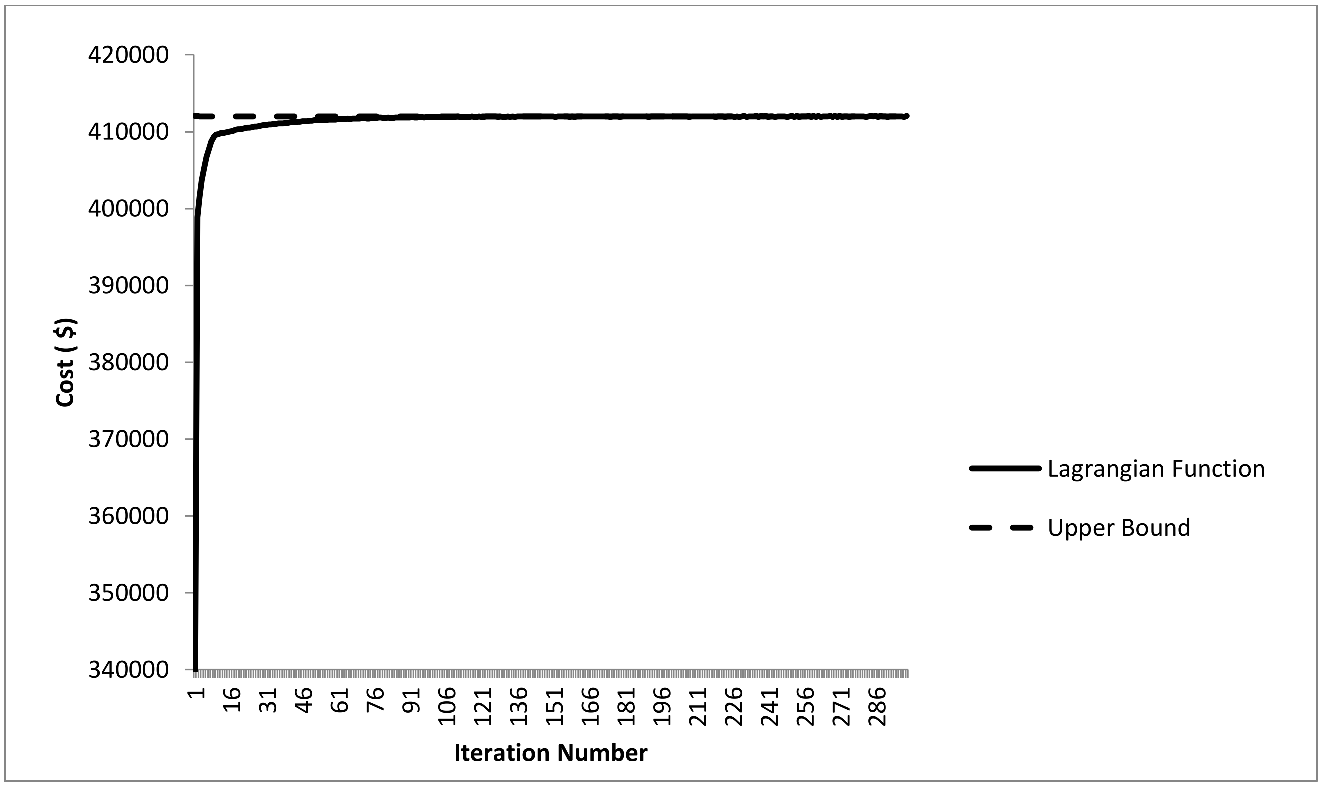

Figure 5.4.Convergence Behaviour of the Lagrangian Function [Example 3]

## 5.4 Summary

In testing the developed model and solution method, we used several example problems of different sizes with randomly generated data. These example problems have 3 to 5 different types of products with 3 to 5 components. The planning time horizon has 3 to 6 periods. The computational results show that the developed solution method can reach optimal or near optimal solution for all of the tested problems within very short computational time. As can be observed, the convergence of the Lagrangian function in Figures (5.1) to (5.4) attributes to the same trend which corresponds to the fact that upper bound stays the same during the whole searching process.

# Chapter Six

# Conclusion and Future Research

In this chapter we present a summary of the research carried out in this thesis. It also includes several concluding remarks based on the problem modeling. Future research directions will be discussed.

## 6.1 Conclusion

Production planning problems in hybrid manufacturing remanufacturing systems (HMRS) are studied. Optimal or near-optimal decisions on system setup, production, inventory to produce new components and remanufactured components are required in solving such problems. The optimal production decisions are to be coordinated with decisions for purchasing, disassembly, assembly and inventory of a both returned products to retrieve the components for remanufacturing and new products to be assembled out of new and remanufactured components.

A mixed integer linear programming (MILP) is developed to obtain optimal solution of the considered problem. To efficiently solve the MILP model, a solution procedure based on Lagrangian decomposition is developed so that the original model can be solved through solving a set of much smaller sub-problems.

The model and solution procedure were tested using several numerical examples. The testing results show that the proposed solution procedure can reach optimal or very close to optimal solutions in short computational time.

## 6.2 Future Research

There are several options to extend the model framework presented in this thesis. Our suggestions for future research in this area could be:

- Considering results application in the real case.
- Considering non deterministic demands for new products and consequently both new components and remanufactured components.
- Considering the combined assembly of new products from new and remanufactured components at the same time.
- Considering more detailed inventory control strategies with back orders and other aspects.

# References

1. Amezquita, T, Hammond, R, Salazor, M, Bras, B. (1995), ”Characterizing the Remanufacturability of Engineering Systems”, Proceeding 1995 ASME Advances in Design Automation Conference, Boston, Massachusetts, USA, pp. 271-278.
2. Behret, H and Korugan, A. (2009), “Performance Analysis of a Hybrid System under Quality Impact of Returns”, Computers and Industrial Engineering, Vol.56, pp.507-520.
3. Chen, M. (1996), “A Mathematical Programming Model for AGVS Planning and Control in Manufacturing Systems”, Computers and Industrial Engineering, Vol.30, pp.647-658.
4. Guide, VDR. (2000), “Production Planning and Control for Remanufacturing: Industry Practice and Research Needs”, Journal of Operations Management, Vol.18, pp.467-483.
5. Evans, JR. (1985), “An Efficient Implementation of Wagner-Whitin Algorithm for Dynamic Lot-Sizing”, Journal of Operations Management, Vol.5, pp.229-235.
6. Fisher, ML. (1985), “An ApplicationS Oriented Guide to Lagrangian Relaxation”, Interfaces, Vol.15, pp.10-21.
7. Fisher, ML. (2004), “The Lagrangian Relaxation Method for Solving Integer Programming Problems”, Management Science, Vol.27, pp.1-18.
8. Fleischmann, M, Krikke, HR, Dekker, R, Flapper, SDP. (2000), “A Characterisation of Logistics Networks for Product Recovery”, Omega, Vol.28, pp.653-666.
9. Giuntini, R and Gaudette, K. (2003), “Remanufacturing: The Next Great Opportunity for Boosting US Productivity”, Business Horizons, Vol.46, pp.41-46.
10. Ijomah, WL, Bennett, JP, Pearce, J. (1999), “Remanufacturing: Evidence of Environmentally Conscious Business Practice in the UK”, Proceeding of the first International Conference on Environmentally Conscious Design and Inverse Manufacturing, Tokyo, Japan.

11. Ijomah, WL, Mcmahon, CA, Hammond, GP, Newman, ST. (2007), "Development of Design for Remanufacturing Guidelines to Support Sustainable Manufacturing", Robotics and Computer-Integrated Manufacturing, Vol.23, pp.712-719.
12. Inderfurth, K. (2004), "Optimal Policies in Hybrid Manufacturing/Remanufacturing Systems with Product Substitution", International Journal of Production Economics, Vol.90, pp.325-343.
13. Kiesmuller, GP. (2003), "A New Approach for Controlling a Hybrid Stochastic Manufacturing/Remanufacturing System with Inventories and Different Lead Times", European Journal of Operation Research, Vol.147, pp. 62-71.
14. Kim, K, Song, I, Kim, J, Jeong, B. (2006), "Supply Planning Model for Remanufacturing System in Reverse Logistics Environment", Computers and Industrial Engineering, Vol.51, pp.279-287.
15. King, AM, Burgess, SC, Ijomah, W, McMahon, CA. (2006), "Reducing Waste: Repair, Recondition, Remanufacture or Recycle?", Sustainable Development, Vol. 14, pp. 257-267.
16. Laan, E van der, Salomon, M, Dekker, R. (1999), "An Investigation of a Lead-Time Effects in Manufacturing/Remanufacturing Systems under Simple PUSH and PULL Control Strategies", European Journal of Operational Research, Vol.115, pp.195-214.
17. Richter, K and Weber, J. (2001), "The Reverse Wagner/Whitin Model with Variable Manufacturing and Remanufacturing Cost", International Journal of Production Economics, Vol. 71, pp. 447-456.
18. Rubio, S and Corominas, A. (2008), "Optimal Manufacturing –Remanufacturing Policies in a Lean Production Environment", Computers and Industrial Engineering, Vol. 55, pp. 234-242.
19. Subramoniam, R, Huisingh, D, Chinnam, RB. (2009), "Remanufacturing for the Automotive Aftermarket-Strategic Factors: Literature Review and Future Research Needs", Journal of Cleaner Production, Vol.17, pp. 1163-1174.

20. Teunter, RH, Bayindir, ZP, Heuvel, W van der. (2006), "Dynamic Lot Sizing with Product Returns and Remanufacturing", International Journal of Production Research, Vol.44, pp.4377-4400.

21. Teunter, R, Kaparis, K, Tang, O. (2008), "Multi-Product Economic Lot Scheduling Problems with Separate Production Lines for Manufacturing and Remanufacturing", European Journal of Operations Research, Vol.191, pp.1241-1253.

# Appendix A

## Python codes for Production Planning Problems of Hybrid Manufacturing-Remanufacturing Systems

```
import numpy as np
import math
import xlrd
import sys
import time
PEGAH=xlrd.open_workbook('C:\\Data\\Desktop\\example two.xls')
PEGAH.sheet_names()
sh=PEGAH.sheet_by_index(0)
def readMatrix(sheet, fromLine, toLine, fromcol, tocol):
    s = [sheet.row_values(row,fromcol,tocol) for row in range(fromLine,toLine+1)]
    return np.matrix(s)
def readArray(sheet, row , fromcol, tocol):
     w=[sh.row_values(row,fromcol, tocol+1)]
     return np.array(w[0])
V=np.matrix(readMatrix(sh,49,52,2,5))
V_bar=np.matrix(readMatrix(sh,56,59,2,5))
P=np.matrix(readMatrix(sh,1,4,1,4))
P_bar=np.matrix(readMatrix(sh,8,11,1,4))
AQ=np.matrix(readMatrix(sh,68,70,2,5))
SD=np.matrix(readMatrix(sh,73,75,2,5))
RD=np.matrix(readMatrix(sh,63,65,2,5))
IN=np.matrix(readMatrix(sh,15,17,1,4))
S=np.matrix(readMatrix(sh,34,37,1,4))
S_bar=np.matrix(readMatrix(sh,41,44,1,4))
D=np.matrix(readMatrix(sh,20,23,1,4))
```

```
D_bar=np.matrix(readMatrix(sh,27,30,1,4))

AST=np.array(readArray(sh,87,2,5))

ST=np.array(readArray(sh,89,2,5))

ASR=np.array(readArray(sh,91,2,5))

SR=np.array(readArray(sh,93,2,5))

ACAP=np.array(readArray(sh, 85, 2, 4))

B=np.matrix(readMatrix(sh,78,81,2,5))

UR=np.array(readArray(sh,95,2,5))

RR=np.matrix(readMatrix(sh,233,235,2,5))

RN=np.matrix(readMatrix(sh,228,230,2,5))

SDN=np.matrix(readMatrix(sh,184,186,2,5))

SDR=np.matrix(readMatrix(sh,189,191,2,5))

RDN=np.matrix(readMatrix(sh,194,196,2,5))

RDR=np.matrix(readMatrix(sh,199,201,2,5))

INR=np.matrix(readMatrix(sh,174,176,2,5))

INN=np.matrix(readMatrix(sh,179,181,2,5))

BN=np.matrix(readMatrix(sh,214,217,2,5))

BR=np.matrix(readMatrix(sh,221,224,2,5))

nPeriods = P.shape[1]

nComponents = P.shape[0]

nProducts = AQ.shape[0]

print nPeriods, nComponents, nProducts

#Computes the cost of producing 'd' of component 'i'  in period [t]

def cost(i, d, t, lmda, phi):

   return d * (P[i , t] + lmda[t] * AST[i]-phi[i,t]) + S[i, t] + lmda[t] * ST[i]

def InvCost(i, fromPeriod, toPeriod, v):

   s = 0

   for t1 in range(fromPeriod + 1, toPeriod+1):

      sumv=0

      for t2 in range(fromPeriod, t1):
           sumv+=v[i,t2]
      s+=D[i,t1]*sumv
   return s
def ww(i, lmda, gmma, phi,khi):
   x = nPeriods * [0]
   LastProductingPeriod = 0
   x = nPeriods * [0]
   teta = nPeriods * [0]
   e = nPeriods * [0]
   x[0] = D[i, 0]
   teta[0] = 1
   e[0] = 0
   minCost = nPeriods * [0]
   minCost[0] = cost(i, D[i, 0], 0, lmda,phi)
   for t in range(1, nPeriods):
      cppc = minCost[t-1] + \
              cost(i, D[i, t], t, lmda, phi)
      lpppc = minCost[LastProductingPeriod] + \
              cost(i, D[i, t], LastProductingPeriod, lmda, phi ) + \
              InvCost(i, LastProductingPeriod, t, V) - \
               S[i, t] + lmda[t] * ST[i]
      if lpppc > cppc:
           x[t] = D[i, t]
           teta[t] = 1
           LastProductingPeriod = t
           minCost[t] = cppc
      else:
           x[LastProductingPeriod] += D[i,t]
           teta[t] = 0
           x[t] = 0
           for period in range(LastProductingPeriod, t):
               e[period] += D[i, t]
           minCost[t] = lpppc
   return x, e, teta
def cost2(i, d_bar, t, lmda, gmma, khi ):
   return d_bar * (P_bar[i , t] + lmda[t] * ASR[i] + gmma[i, t]-khi[i,t]) + S_bar[i, t] + lmda[t] * SR[i]
def InvCost2(i, fromPeriod, toPeriod, v):
   s = 0
   for t1 in range(fromPeriod + 1, toPeriod+1):
      sumv = 0
      for t2 in range(fromPeriod, t1):
           sumv += v[i,t2]
      s += D_bar[i,t1] * sumv
   return s
def ww2(i, lmda, gmma,phi,khi):
   x = nPeriods * [0]
   LastProductingPeriod = 0
   x_bar = nPeriods * [0]
   teta_bar = nPeriods * [0]
   e_bar = nPeriods * [0]
   x_bar[0] = D_bar[i, 0]
   teta_bar[0] = 1
   e_bar[0] = 0
   minCost = nPeriods * [0]
   minCost[0] = cost2(i, D_bar[i, 0], 0, lmda, gmma, khi)
   for t in range(1, nPeriods):
      cppc = minCost[t-1] + \
              cost2(i, D_bar[i, t], t, lmda, gmma, khi)
      lpppc = minCost[LastProductingPeriod] + \
              cost2(i, D_bar[i, t], LastProductingPeriod, lmda, gmma, khi) + \
              InvCost2(i, LastProductingPeriod, t, V_bar) - \
               S_bar[i, t] + lmda[t] * SR[i]

      if lpppc > cppc:
           x_bar[t] = D_bar[i, t]
           teta_bar[t] = 1
           LastProductingPeriod = t
           minCost[t] = cppc
      else:
           x_bar[LastProductingPeriod] += D_bar[i,t]
           teta_bar[t] = 0
           x_bar[t] = 0
           for period in range(LastProductingPeriod, t):
               e_bar[period] += D_bar[i, t]
           minCost[t] = lpppc
   return x_bar, e_bar, teta_bar
#**************************************************************************************************
def cost3(j, RN, t, lmda, gmma, phi ):
   vv=0
   for i in range(1,nComponents):
      vv+=BN[i,j]* phi[i,t]
   return RN * (RDN[j , t] +vv)+SDN[j,t]
def InvCost3(j, fromPeriod, toPeriod, INN):
   s = 0
   for t1 in range(fromPeriod + 1, toPeriod+1):
      sumINN = 0
      for t2 in range(fromPeriod, t1):
           sumINN += INN[j,t2]
      s += RN[j,t1] * sumINN
   return s
```

```
def ww3(j, lmda, gmma,phi,khi):
    x = nPeriods * [0]
    LastProductingPeriod = 0
    dn = nPeriods * [0]
    deln = nPeriods * [0]
    fn = nPeriods * [0]
    dn[0] = RN[j, 0]
    deln[0] = 1
    fn[0] = 0
    minCost = nPeriods * [0]
    minCost[0] = cost3(j, RN[j, 0], 0, lmda, gmma, phi)
    for t in range(1, nPeriods):
        cppc = minCost[t-1] + \
               cost3(j, RN[j, 0], 0, lmda, gmma, phi)
        lpppc = minCost[LastProductingPeriod] + \
               cost3(j, RN[j, t], LastProductingPeriod, lmda, gmma, phi) + \
               InvCost3(j, LastProductingPeriod, t, INN) - \
                SDN[j,t]
        if lpppc > cppc:
            dn[t] = RN[j, t]
            deln[t] = 1
            LastProductingPeriod = t
            minCost[t] = cppc
        else:
            dn[LastProductingPeriod] += RN[j,t]
            deln[t] = 0
            dn[t] = 0
            for period in range(LastProductingPeriod, t):
                fn[period] += RN[j, t]
            minCost[t] = lpppc
    return fn, dn, deln
#***************************************************************************************************
def cost4(j, RR, t, lmda, gmma, khi ):
    ll=0
    for i in range (1,nComponents):
        ll+=BR[i,j]* khi[i,t]
    return RR * (RDR[j , t] +ll)+SDR[j,t]
def InvCost4(j, fromPeriod, toPeriod, INR):
    s = 0
    for t1 in range(fromPeriod + 1, toPeriod+1):
        sumINR = 0
        for t2 in range(fromPeriod, t1):
            sumINR += INR[j,t2]
        s += RR[j,t1] * sumINR
    return s
def ww4(j, lmda, gmma,phi,khi):
    x = nPeriods * [0]
    LastProductingPeriod = 0
    dr = nPeriods * [0]
    delr = nPeriods * [0]
    fr = nPeriods * [0]
    dr[0] = RR[j, 0]
    delr[0] = 1
    fr[0] = 0
    minCost = nPeriods * [0]
    minCost[0] = cost4(j, RR[j, 0], 0, lmda, gmma, khi)
    for t in range(1, nPeriods):
        cppc = minCost[t-1] + \
               cost4(j, RR[j, t], t, lmda, gmma, khi)
        lpppc = minCost[LastProductingPeriod] + \
               cost4(j, RR[j, t], LastProductingPeriod, lmda, gmma, khi) + \
               InvCost4(j, LastProductingPeriod, t, INR) - \
                SDR[j,t]
        if lpppc > cppc:
            dr[t] = RR[j, t]
            delr[t] = 1
            LastProductingPeriod = t
            minCost[t] = cppc
        else:
            dr[LastProductingPeriod] += RR[j,t]
            delr[t] = 0
            dr[t] = 0
            for period in range(LastProductingPeriod, t):
               fr[period] += RR[j, t]
            minCost[t] = lpppc
    return fr, dr, delr
#***************************************************************************************
import pulp
def getD_tilde(x_bar):
    prob = pulp.LpProblem("Sub3 LP",pulp.LpMinimize)
    periods = [str(i) for i in range(nPeriods)]
    products=[str(k) for k in range(nProducts)]
    varNames = [str(k) + str(i)  for i in range(nPeriods) for k in range(nProducts)]
    d_tilde=pulp.LpVariable.dict('d_tilde', varNames,lowBound =0, cat = pulp.LpInteger)
    print varNames
    prob += sum(d_tilde[str(j) + str(t)] for j in range(nProducts) for t in range(nPeriods))
    for i in range(nComponents):
        for t in range (nPeriods):
            prob +=   UR[i] * sum(B[i,j] * d_tilde[str(j)+str(t)] for j in range(nProducts)) >= x_bar[i,t]
    prob.solve(pulp.COIN(msg=0))
    ret_d_tilde = np.matrix(np.zeros((nProducts, nPeriods)))
    for v in prob.variables():
        n = v.name
        ret_d_tilde[n[-2], n[-1]] = v.varValue
    return ret_d_tilde
def PegHeur(x_bar, gmma):
    #Initializing the return variables
    d = np.matrix(np.zeros((nProducts, nPeriods)))
    delta = np.matrix(np.zeros((nProducts, nPeriods)))
    r = np.matrix(np.zeros((nProducts, nPeriods)))
    f = np.matrix(np.zeros((nProducts, nPeriods)))
    #Setting the values of d_tilde
    d_tilde = getD_tilde(x_bar)
    print "d_tilde"
    print d_tilde
    #Initializeing R_tilde
    R_tilde = np.matrix(np.zeros((nProducts, nPeriods)))
    for t in range(nPeriods):
        for j in range(nProducts):
            s = sum(B[i,j] * gmma[i,t] * UR[i] for i in range(nComponents))
            R_tilde[j,t] = RD[j,t] - s
    A_tilde = np.matrix(np.zeros((nProducts, nPeriods)))
    for j in range(nProducts):
        A_tilde[j, 0] = AQ[j,0]
    for j in range(nProducts):
        for t in range(1, nPeriods):
            A_tilde[j,t] = min(AQ[j,t], IN[j, t-1] + A_tilde[j, t-1])
    #step2
    for j in range(nProducts):
        d[j,0] = r[j,0] = d_tilde[j,0]
        if d[j,0]>0:
            delta[j,0] = 1
    #step3
    maxper=0
    for j in range(nProducts):
        for t in range(1,nPeriods):
            if R_tilde[j, t] >= - (A_tilde[j,t] * d_tilde[j, t] + SD[j,t]):
                d[j,t]= r[j,t] = d_tilde[j,t]=d[j,t]=0
            else:
                if AQ[j,t] <= IN[j, t-1] + A_tilde[j,t-1]:
                    r[j,t] = d_tilde[j,t]
                    f[j,t]=0
                    delta[j,t]=1
                    maxper=t
                else:
                    r[j, maxper] += d_tilde[j,t]
                    delta=[j,maxper]=1
                    for k in range(maxper,t-1):
                        f[j,k]+=d_tilde[j,t]
    return  d,delta, r,f
def getSlacksOfFirstCommonConstraint(x, teta, x_bar, teta_bar):
    slack = []
    for t in range(nPeriods):
        s = sum(AST[i]*x[i,t] + ST[i] * teta[i,t] + ASR[i]*x_bar[i,t] + SR[i] * teta_bar[i,t] for i in range(nComponents))
        slack.append(s - ACAP[t])
    return slack
    slackone =sum(slack for i in range (nComponents) for t in range (nPeriods))
    #print slackone
def getSlacksOfSecondCommonConstraint(x_bar, d):
    slack = []
    for i in range(nComponents):
        slack.append([])
        for t in range(nPeriods):
            s = sum(B[i,j] * d[j,t] for j in range(nProducts))
            s *= UR[i]
            slack[-1].append(x_bar[i, t]- s)
    return slack
#***************************************************************************
def getSlacksOfThirdCommonConstraint(x, dn):
    slack = []
    for i in range(nComponents):
        slack.append([])
        for t in range(nPeriods):
            s = sum(BN[i,j] * dn[j,t] for j in range(nProducts))
            slack[-1].append(x[i, t]- s)
    return slack
def getSlacksOfForthCommonConstraint(x_bar, dr):
    slack = []
    for i in range(nComponents):
        slack.append([])
        for t in range(nPeriods):
            s = sum(BR[i,j] * dr[j,t] for j in range(nProducts))
            slack[-1].append(x_bar[i, t]- s)
    return slack
#***************************************************************************
def Z_function(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d,fr,fn,deln,delr,dr,dn):
        gmma = np.matrix(np.zeros((nComponents, nPeriods)))
        lmda = np.array(np.zeros(nPeriods))
        phi= np.matrix(np.zeros((nComponents, nPeriods)))
        khi= np.matrix(np.zeros((nComponents, nPeriods)))
        return  Z_lag(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d, lmda, gmma,fr,fn,deln,delr,dr,dn,phi,khi)
def Z_lag(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d, lmda, gmma,fr,fn,deln,delr,dr,dn,phi,khi):
    sum_sub1 =0
    for i in range (nComponents):
        sum_sub1 += sum([(P[i,t] +lmda[t]*AST[i]+phi[i,t])* x[i,t] + (S[i, t]+lmda[t]*ST[i])*teta[i, t] \
                    + V[i, t]* e[i, t] for t in range(nPeriods)])
    sum_sub2 =0
    for i in range (nComponents):
        sum_sub2 += sum((P_bar[i,t]+lmda[t]*ASR[i]+gmma[i,t]+khi[i,t])* x_bar[i,t] + \
                   (S_bar[i, t]+lmda[t]*SR[i])* teta_bar[i, t] + V_bar[i, t]* e_bar[i, t]for t in range(nPeriods))
    sum_sub3 =0
    for j in range (nProducts):
        sum_sub3 += sum([AQ[j, t]* r[j, t] +SD[j, t]* delta[j, t]+ (RD[j, t] - \
                   sum([B[i,j]*gmma[i,t]*UR[i] for i in range(nComponents)]))*d[j,t] +IN[j, t]* f[j, t] \
                    for t in range(nPeriods)])
    sum_sub4 =0
    for j in range (nProducts):
        sum_sub4 += sum([SDN[j, t]* deln[j, t]+ (RDN[j, t] - \
                   sum([BN[i,j]*phi[i,t] for i in range(nComponents)]))*dn[j,t] +INN[j, t]* fn[j, t] \
                    for t in range(nPeriods)])
    sum_sub5 =0
    for j in range (nProducts):
        sum_sub5 += sum([SDR[j, t]* delr[j, t]+ (RDR[j, t] - \
                   sum([BR[i,j]*khi[i,t] for i in range(nComponents)]))*dr[j,t] +INR[j, t]* fr[j, t] \
                    for t in range(nPeriods)])
          sum_sub1=sum_sub1- sum([lmda[t]*ACAP[t] for t in range(nPeriods)])
    return sum_sub1 + sum_sub2 + sum_sub3 +sum_sub4 +sum_sub5
from xlwt import Workbook
def initializeXL(filename):
    workbook = Workbook()
    sheet1 = workbook.add_sheet('Lagranigian')
    sheet1.write(0,0,'Iteration')
    sheet1.write(0,1,'z_l')
    sheet1.write(0,2,'z')
    sheet1.write(0,3,'Feasibility')
    sheet1.write(0,4,'ub')
    ind = 5
    for t in range(nPeriods):
        sheet1.write(0,ind,'lamda_'+str(t))
        ind += 1
    for i in range(nComponents):
        for t in range(nPeriods):
            sheet1.write(0,ind,'gamma_'+str(i) + str(t))
            ind += 1
    return workbook,sheet1
def finalizeXL(book, filename):
    book.save(filename + '.xls')
def writeToXL(sheet, iteration, feasible, z_l,ub, z,x, x_bar, teta, teta_bar, e, e_bar, f, r, d, delta, lmda, gmma):
    sheet.write(iteration + 1,0,iteration)
    sheet.write(iteration + 1,1,z_l)
    sheet.write(iteration + 1,2,z)
    sheet.write(iteration + 1,3,feasible)
    sheet.write(iteration + 1,4,ub)
    ind = 5
    for t in range(nPeriods):
        sheet.write(iteration + 1,ind,lmda[t])
        ind += 1
    for i in range(nComponents):
       for t in range(nPeriods):
            sheet.write(iteration + 1,ind,gmma[i,t])
            ind += 1
    sheet.flush_row_data()
import time
def lagSolve(lmda, gmma,phi,khi, maxIterations):
    ub =334291.0
    noUpdateIterations = 0
    stoppingIteration = 0
    start1= time.time()
    print "start1=", start1
    f = open('res.txt', 'w')
    f.write('hello')
    f.close()
    book, sheet = initializeXL('expanded results')
    x = np.matrix(np.zeros((nComponents, nPeriods)))
    e = np.matrix(np.zeros((nComponents, nPeriods)))
    teta = np.matrix(np.zeros((nComponents, nPeriods)))
    x_bar = np.matrix(np.zeros((nComponents, nPeriods)))
    e_bar = np.matrix(np.zeros((nComponents, nPeriods)))
    teta_bar = np.matrix(np.zeros((nComponents, nPeriods)))
    dn = np.matrix(np.zeros((nProducts,nPeriods)))
    deln = np.matrix(np.zeros((nProducts,nPeriods)))
    fn = np.matrix(np.zeros((nProducts,nPeriods)))
    dr = np.matrix(np.zeros((nProducts,nPeriods)))
    delr = np.matrix(np.zeros((nProducts,nPeriods)))
    fr = np.matrix(np.zeros((nProducts,nPeriods)))


    step = 2
    z_best = 10**10
    x_best= x_bar_best= e_best= e_bar_best= f_best=fn_best=fr_best= r_best= d_best=dn_best=dr_best=
delta_best=delr_best=deln_best= \
            teta_best = teta_bar_best = best_lmda = best_gmma =best_khi=best_phi= -1
    k=2.0
    z_l = -1
    for iteration in range(1, maxIterations):
        for i in range(nComponents):
            x_i,e_i,teta_i = ww(i, lmda, gmma,phi,khi)
            x[i, :] = x_i
            e[i, :] = e_i
            teta[i, :] = teta_i
        for i in range(nComponents):
            x_bar_i, e_bar_i, teta_bar_i = ww2(i, lmda, gmma,phi,khi)
            x_bar[i, :] = x_bar_i
            e_bar[i, :] = e_bar_i
            teta_bar[i, :] = teta_bar_i
        for j in range (nProducts):
            fn_j, dn_j, deln_j=ww3(j, lmda, gmma,phi,khi)
            fn[j,:]=fn_j
            dn[j,:]=dn_j
            deln[j,:]=deln_j
        for j in range (nProducts):
            fr_j, dr_j, delr_j=ww4(j, lmda, gmma,phi,khi)
            fr[j,:]=fr_j
            dr[j,:]=dr_j
            delr[j,:]=delr_j
        d,delta, r,f = PegHeur(x_bar, gmma)
             if iteration == 299:
             print "gmma=",gmma
             print "lmda=",lmda
             print "phi=",phi
             print "khi=",khi
        print "dr=",dr
        print "dn=",dn
        print "fn=",fn
        print "fr=",fr
        print "deln=",deln
        print "delr=",delr
        print "x="
        print x
        print "x_bar="
        print x_bar
        print "e="
        print e
        print "e_bar="
        print e_bar
        print "teta="
        print teta
        print "teta_bar="
        print teta_bar
        print "r="
        print r
        print "f="
        print f
        print "delta="
        print delta
        print "d="
        print d
        print 'Primal Objective Value:'
        print Z_function(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d,fr,fn,deln,delr,dr,dn)
    s1 = getSlacksOfFirstCommonConstraint(x, teta, x_bar, teta_bar)
    s2 = getSlacksOfSecondCommonConstraint(x_bar, d)
```

```
s3 = getSlacksOfThirdCommonConstraint(x, dn)
s4 = getSlacksOfForthCommonConstraint(x_bar, dr)
feasible = '-'
print "number of infeasible constraints = " , [s>0.0000000001 for s in s1].count(True)
if [s>0.0000000001 for s in s1].count(True)>0:
   print [s>0.0000000001 for s in s1].index(True)
else:
   feasible='feasible'
if [s>0.0000000001 for s in s1].count(True) == 0 and (gmma>=0).all():# and Z_lag < z_best:
   feasible = 'feasible'
z_lold=z_l
z_l = Z_lag(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d, lmda, gmma,fr,fn,deln,delr,dr,dn,phi,khi)
oo=z_l-z_lold
if (oo <0.00001).all():
   noUpdateIterations += 1
else:
   noUpdateIterations = 0
z= Z_function(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d,fr,fn,deln,delr,dr,dn)
if feasible == 'feasible' and z < ub:
   ub = z
     writeToXL(sheet,iteration, feasible, z_l, ub,z, x, x_bar, teta, teta_bar, e, e_bar, f, r, d, delta, lmda, gmma)
if noUpdateIterations == 15:
   noUpdateIterations = 0
   k=k/2
qp=z-z_l
if (0<qp and qp <0.01).all():
   print "best soulution so far", z
   duration_if=time.time()-start1
  #print 'Problem took ' + '%6.2f'%duration_if + ' seconds'
   print "stoppingIteration",stoppingIteration
print "gmma=",gmma

print "lmda=",lmda

print "phi=",phi

print "khi=",khi

print "dr=",dr

print "dn=",dn

print "fn=",fn

print "fr=",fr

print "deln=",deln

print "delr=",delr

print "x="

print x

print "x_bar="

print x_bar

print "e="

print e

print "e_bar="

print e_bar

print "teta="

print teta

print "teta_bar="

print teta_bar

print "r="

print r

print "f="

print f

print "delta="

print delta

print "d="

print d
```

```
            print 'Primal Objective Value:'
            print Z_function(x, x_bar, e, e_bar, teta, teta_bar, r, f, delta, d,fr,fn,deln,delr,dr,dn)
            break
           #iteration = stoppingIteration
        else:
            stoppingIteration += 1
        denom=sum([h**2 for h in s1])
        for s in s2:
            denom +=sum ([h**2 for h in s])
        step= k*(ub-z_l)/denom
        lmda = [max(0, lmda[i] + step * s1[i]) for i in range(nPeriods)]
        for i in range(nComponents):
            for t in range(nPeriods):
                gmma[i, t] = max(0, gmma[i, t] + step * s2[i][t])
        for i in range(nComponents):
            for t in range(nPeriods):
                phi[i, t] = max(0, phi[i, t] + step * s3[i][t])
        for i in range(nComponents):
            for t in range(nPeriods):
                khi[i, t] = max(0, khi[i, t] + step * s4[i][t])
    finalizeXL(book, 'expanded results')
g = np.matrix(np.zeros((nComponents, nPeriods)))
g += 0.1
l=np.array(np.zeros(nPeriods))
l+= 0.1
p =np.matrix(np.zeros((nComponents, nPeriods)))
p+= 0.1
k =np.matrix(np.zeros((nComponents, nPeriods)))
k+= 0.1
start = time.time()
```

```
lagSolve(l,g,p,k ,300)

duration = time.time()-start

print 'Problem took ' + '%6.2f'%duration + ' seconds'
```

# Appendix B

**Lingo codes for finding the first feasible solution in solving HMRS problem**

**Sub model 1**

```
SETS:
Component/1..4/:AST,URST,ASR,SR,ST,UR;                                  !i;
Product/1..3/;                                                    !j;
Period/1..3/:ACAP,Landa;                                              !t;
ComPer(Component,Period):X,X_bar,P,P_bar,V,V_bar,S,S_bar,Tet,Tet_bar,e,e_bar,D,D_ba
r; !i,t;
ProPer(Product,Period):dtilda,f,r,del,AQ,SD,RD,IN,INN,SDN,FN,deln,dn,RN,RDN;
ComPro(Component,Product): B,BN;


ENDSETS


DATA:


P = @OLE('C:\Data\Desktop\example tow.xls','P') ;
P_bar = @OLE('C:\Data\Desktop\example tow.xls','P_bar ') ;
IN = @OLE('C:\Data\Desktop\example tow.xls','IN') ;
D = @OLE('C:\Data\Desktop\example tow.xls','D') ;
D_bar = @OLE('C:\Data\Desktop\example tow.xls','D_bar ') ;
S = @OLE('C:\Data\Desktop\example tow.xls','S') ;
S_bar = @OLE('C:\Data\Desktop\example tow.xls','S_bar ') ;
V = @OLE('C:\Data\Desktop\example tow.xls','V') ;
V_bar = @OLE('C:\Data\Desktop\example tow.xls','V_bar ') ;
RD = @OLE('C:\Data\Desktop\example tow.xls','RD') ;
AQ = @OLE('C:\Data\Desktop\example tow.xls','AQ ') ;
SD = @OLE('C:\Data\Desktop\example tow.xls','SD') ;
B = @OLE('C:\Data\Desktop\example tow.xls','B') ;
ACAP = @OLE('C:\Data\Desktop\example tow.xls','ACAP') ;
SDN=@OLE('C:\Data\Desktop\example tow.xls','SDN') ;
BN=@OLE('C:\Data\Desktop\example tow.xls','BN') ;
RN=@OLE('C:\Data\Desktop\example tow.xls','RN') ;
RDN=@OLE('C:\Data\Desktop\example tow.xls','RDN') ;
INN=@OLE('C:\Data\Desktop\example tow.xls','INN') ;
AST = @OLE('C:\Data\Desktop\example tow.xls','AST') ;
ST = @OLE('C:\Data\Desktop\example tow.xls','ST') ;
ASR = @OLE('C:\Data\Desktop\example tow.xls','ASR') ;
SR = @OLE('C:\Data\Desktop\example tow.xls','SR') ;
UR = @OLE('C:\Data\Desktop\example tow.xls','UR') ;
M= @OLE('C:\Data\Desktop\example tow.xls','M') ;
@OLE('C:\Data\Desktop\example tow.xls','X') = X;
@OLE('C:\Data\Desktop\example tow.xls','e') = e;
@OLE('C:\Data\Desktop\example tow.xls','Tet') = Tet;

END DATA

!OBJECTIVE FUNCTION;

Min= @sum(ComPer(i,t): P(i,t)*X(i,t) + S(i,t)*Tet(i,t) + V(i,t)*e(i,t));


!SUBJECT TO;
@for(ComPer(i,t):@BIN(Tet(i,t)));
@for(ComPer(i,t):@GIN(e(i,t)));
```

```
@for(ComPer(i,t):@GIN(X(i,t)));


!1;
@for(ComPer(i,t)| t #GT# 1 : e(i,t-1)+ X(i,t) - e(i,t)= D(i,t));
@for(Component(i) : X(i,1) - e(i,1)= D(i,1));
!2;
@for(ComPer(i,t): X(i,t) <= 100000* Tet(i,t));

!3;
@for(Period(t):@sum(Component(i): AST(i)* X(i,t) + ST(i)*Tet(i,t)) <= ACAP(t) );
```

**Submodel 2**

```
SETS:
Component/1..4/:AST,URST,ASR,SR,ST,UR;                              !i;
Product/1..3/;                                               !j;
Period/1..3/:ACAP,Landa,RAMA;                                       !t;
ComPer(Component,Period):X,X_bar,P,P_bar,V,V_bar,S,S_bar,Tet,Tet_bar,e,e_bar,D,D_ba
r; !i,t;
ProPer(Product,Period):dtilda,f,r,del,delr,deln,FN,FR,SDN,SDR,RDR,RDN,INN,INR,RR,RN
,dn,dr,AQ,SD,RD,IN;
ComPro(Component,Product): B,BN,BR;


ENDSETS


DATA:
P = @OLE('C:\Data\Desktop\example tow.xls','P') ;
P_bar = @OLE('C:\Data\Desktop\example tow.xls','P_bar ') ;
IN = @OLE('C:\Data\Desktop\example tow.xls','IN') ;
D = @OLE('C:\Data\Desktop\example tow.xls','D') ;
D_bar = @OLE('C:\Data\Desktop\example tow.xls','D_bar ') ;
S = @OLE('C:\Data\Desktop\example tow.xls','S') ;
S_bar = @OLE('C:\Data\Desktop\example tow.xls','S_bar ') ;
V = @OLE('C:\Data\Desktop\example tow.xls','V') ;
V_bar = @OLE('C:\Data\Desktop\example tow.xls','V_bar ') ;
RD = @OLE('C:\Data\Desktop\example tow.xls','RD') ;
AQ = @OLE('C:\Data\Desktop\example tow.xls','AQ ') ;
SD = @OLE('C:\Data\Desktop\example tow.xls','SD') ;
B = @OLE('C:\Data\Desktop\example tow.xls','B') ;
ACAP = @OLE('C:\Data\Desktop\example tow.xls','ACAP') ;
RAMA = @OLE('C:\Data\Dropbox\thesis\EXAMPLE TWO\remaining.xls','RAMA') ;
AST = @OLE('C:\Data\Desktop\example tow.xls','AST') ;
ST = @OLE('C:\Data\Desktop\example tow.xls','ST') ;
ASR = @OLE('C:\Data\Desktop\example tow.xls','ASR') ;
SR = @OLE('C:\Data\Desktop\example tow.xls','SR') ;
UR = @OLE('C:\Data\Desktop\example tow.xls','UR') ;
M= @OLE('C:\Data\Desktop\example tow.xls','M') ;
SDR=@OLE('C:\Data\Desktop\example tow.xls','SDR') ;
SDN=@OLE('C:\Data\Desktop\example tow.xls','SDN') ;
```

```
BR=@OLE('C:\Data\Desktop\example tow.xls','BR') ;
BN=@OLE('C:\Data\Desktop\example tow.xls','BN') ;
RDR=@OLE('C:\Data\Desktop\example tow.xls','RDR') ;
RDN=@OLE('C:\Data\Desktop\example tow.xls','RDN') ;
INN=@OLE('C:\Data\Desktop\example tow.xls','INN') ;
INR=@OLE('C:\Data\Desktop\example tow.xls','INR') ;
RR=@OLE('C:\Data\Desktop\example tow.xls','RR') ;
RN=@OLE('C:\Data\Desktop\example tow.xls','RN') ;
@OLE('C:\Data\Desktop\example tow.xls','X_bar') = X_bar;
@OLE('C:\Data\Desktop\example tow.xls','Tet_bar') = Tet_bar;
@OLE('C:\Data\Desktop\example tow.xls','dtilda') = dtilda;
@OLE('C:\Data\Desktop\example tow.xls','RO') = r;
@OLE('C:\Data\Desktop\example tow.xls','F') = f;
@OLE('C:\Data\Desktop\example tow.xls','del') = del;
@OLE('C:\Data\Desktop\example tow.xls','e_bar') = e_bar;

END DATA

!OBJECTIVE FUNCTION;

Min= @sum(ComPer(i,t): P_bar(i,t)*X_bar(i,t) + S_bar(i,t)*Tet_bar(i,t) +
V_bar(i,t)*e_bar(i,t))+
@sum(ProPer(j,t):AQ(j,t)*r(j,t)+SD(j,t)*del(j,t)+RD(j,t)*dtilda(j,t)+IN(j,t)*f(j,t)
);


!SUBJECT TO;
@for(ComPer(i,t):@BIN(Tet(i,t)));
@for(ComPer(i,t):@BIN(Tet_bar(i,t)));
@for(ProPer(j,t):@BIN(del(j,t)));
@for(ComPer(i,t):@GIN(X_bar(i,t)));
@for(ComPer(i,t):@GIN(e_bar(i,t)));
@for(ProPer(j,t):@GIN(r(j,t)));
@for(ProPer(j,t):@GIN(f(j,t)));
@for(ProPer(j,t):@GIN(dtilda(j,t)));


!1;
@for(ComPer(i,t)| t #GT# 1 : e_bar(i,t-1)+ X_bar(i,t) - e_bar(i,t)= D_bar(i,t));
@for(Component(i): X_bar(i,1) - e_bar(i,1)= D_bar(i,1));

!2;
@for(ComPer(i,t): X_bar(i,t) <= M* Tet_bar(i,t));



!3;
@for(Period(t):@sum(Component(i):ASR(i)*X_bar(i,t) + SR(i)*Tet_bar(i,t) ) <=
RAMA(t) );


!4;
@for(ProPer(j,t)| t #GT# 1 : f(j,t) + dtilda(j,t) - f(j,t-1) = r(j,t));
@for(Product(j) : dtilda(j,1)+f(j,1)= r(j,1));


!5;
```

```
@for(ProPer(j,t): dtilda(j,t)  <= M*del(j,t));
!6;
@for(ComPer(i,t): X_bar(i,t)<= UR(i)* @SUM(Product(j):B(i,j)*dtilda(j,t)));
```

**Sub model 3**

```
SETS:
Component/1..4/:AST,URST,ASR,SR,ST,UR;                                !i;
Product/1..3/;                                                   !j;
Period/1..3/:ACAP,Landa;                                               !t;
ComPer(Component,Period):X,X_bar,P,P_bar,V,V_bar,S,S_bar,Tet,Tet_bar,e,e_bar,D,D_ba
r; !i,t;
ProPer(Product,Period):dtilda,f,r,del,AQ,SD,RD,IN,INN,SDN,FN,deln,dn,RN,RDN;
ComPro(Component,Product): B,BN;


ENDSETS


DATA:

X=@OLE('C:\Data\Desktop\example tow.xls','X') ;

P = @OLE('C:\Data\Desktop\example tow.xls','P') ;
P_bar = @OLE('C:\Data\Desktop\example tow.xls','P_bar ') ;
IN = @OLE('C:\Data\Desktop\example tow.xls','IN') ;
D = @OLE('C:\Data\Desktop\example tow.xls','D') ;
D_bar = @OLE('C:\Data\Desktop\example tow.xls','D_bar ') ;
S = @OLE('C:\Data\Desktop\example tow.xls','S') ;
S_bar = @OLE('C:\Data\Desktop\example tow.xls','S_bar ') ;
V = @OLE('C:\Data\Desktop\example tow.xls','V') ;
V_bar = @OLE('C:\Data\Desktop\example tow.xls','V_bar ') ;
RD = @OLE('C:\Data\Desktop\example tow.xls','RD') ;
AQ = @OLE('C:\Data\Desktop\example tow.xls','AQ ') ;
SD = @OLE('C:\Data\Desktop\example tow.xls','SD') ;
B = @OLE('C:\Data\Desktop\example tow.xls','B') ;
ACAP = @OLE('C:\Data\Desktop\example tow.xls','ACAP') ;
SDN=@OLE('C:\Data\Desktop\example tow.xls','SDN') ;
BN=@OLE('C:\Data\Desktop\example tow.xls','BN') ;
RN=@OLE('C:\Data\Desktop\example tow.xls','RN') ;
RDN=@OLE('C:\Data\Desktop\example tow.xls','RDN') ;
INN=@OLE('C:\Data\Desktop\example tow.xls','INN') ;
AST = @OLE('C:\Data\Desktop\example tow.xls','AST') ;
ST = @OLE('C:\Data\Desktop\example tow.xls','ST') ;
ASR = @OLE('C:\Data\Desktop\example tow.xls','ASR') ;
SR = @OLE('C:\Data\Desktop\example tow.xls','SR') ;
UR = @OLE('C:\Data\Desktop\example tow.xls','UR') ;
M= @OLE('C:\Data\Desktop\example tow.xls','M') ;
@OLE('C:\Data\Desktop\example tow.xls','deln') = deln;
@OLE('C:\Data\Desktop\example tow.xls','FN') = FN;
@OLE('C:\Data\Desktop\example tow.xls','dn') = dn;
```

```
END DATA

!OBJECTIVE FUNCTION;

Min= @sum(ProPer(j,t):SDN(j,t)*deln(j,t)+RDN(j,t)*dn(j,t)+INN(j,t)*FN(j,t));


!SUBJECT TO;
@for(ProPer(j,t):@BIN(deln(j,t)));

@for(ProPer(j,t):@GIN(FN(j,t)));
@for(ProPer(j,t):@GIN(dn(j,t)));

!1;
@for(ProPer(j,t)| t #GT# 1 : FN(j,t-1) + dn(j,t) - FN(j,t) = RN(j,t));
@for(Product(j) : dn(j,1)- FN(j,1) = RN(j,1));
!2;
@for(ProPer(j,t): dn(j,t) <= 100000*deln(j,t));
!3;
@for(ComPer(i,t): X(i,t)>= @SUM(Product(j):BN(i,j)*dn(j,t)));
```

**Sub model 4**

```
SETS:
Component/1..4/:AST,URST,ASR,SR,ST,UR;                                    !i;
Product/1..3/;                                                      !j;
Period/1..3/:ACAP,Landa,RAMA;                                             !t;
ComPer(Component,Period):X,X_bar,P,P_bar,V,V_bar,S,S_bar,Tet,Tet_bar,e,e_bar,D,D_ba
r; !i,t;
ProPer(Product,Period):dtilda,f,r,del,delr,deln,FN,FR,SDN,SDR,RDR,RDN,INN,INR,RR,RN
,dn,dr,AQ,SD,RD,IN;
ComPro(Component,Product): B,BN,BR;


ENDSETS


DATA:
X_bar=@OLE('C:\Data\Desktop\example tow.xls','X_bar') ;
P = @OLE('C:\Data\Desktop\example tow.xls','P') ;
P_bar = @OLE('C:\Data\Desktop\example tow.xls','P_bar ') ;
IN = @OLE('C:\Data\Desktop\example tow.xls','IN') ;
D = @OLE('C:\Data\Desktop\example tow.xls','D') ;
D_bar = @OLE('C:\Data\Desktop\example tow.xls','D_bar ') ;
S = @OLE('C:\Data\Desktop\example tow.xls','S') ;
S_bar = @OLE('C:\Data\Desktop\example tow.xls','S_bar ') ;
V = @OLE('C:\Data\Desktop\example tow.xls','V') ;
V_bar = @OLE('C:\Data\Desktop\example tow.xls','V_bar ') ;
RD = @OLE('C:\Data\Desktop\example tow.xls','RD') ;
AQ = @OLE('C:\Data\Desktop\example tow.xls','AQ ') ;
SD = @OLE('C:\Data\Desktop\example tow.xls','SD') ;
B = @OLE('C:\Data\Desktop\example tow.xls','B') ;
ACAP = @OLE('C:\Data\Desktop\example tow.xls','ACAP') ;
AST = @OLE('C:\Data\Desktop\example tow.xls','AST') ;
```

```
ST = @OLE('C:\Data\Desktop\example tow.xls','ST') ;
ASR = @OLE('C:\Data\Desktop\example tow.xls','ASR') ;
SR = @OLE('C:\Data\Desktop\example tow.xls','SR') ;
UR = @OLE('C:\Data\Desktop\example tow.xls','UR') ;
M= @OLE('C:\Data\Desktop\example tow.xls','M') ;
SDR=@OLE('C:\Data\Desktop\example tow.xls','SDR') ;
SDN=@OLE('C:\Data\Desktop\example tow.xls','SDN') ;
BR=@OLE('C:\Data\Desktop\example tow.xls','BR') ;
BN=@OLE('C:\Data\Desktop\example tow.xls','BN') ;
RDR=@OLE('C:\Data\Desktop\example tow.xls','RDR') ;
RDN=@OLE('C:\Data\Desktop\example tow.xls','RDN') ;
INN=@OLE('C:\Data\Desktop\example tow.xls','INN') ;
INR=@OLE('C:\Data\Desktop\example tow.xls','INR') ;
RR=@OLE('C:\Data\Desktop\example tow.xls','RR') ;
RN=@OLE('C:\Data\Desktop\example tow.xls','RN') ;
@OLE('C:\Data\Desktop\example tow.xls','FR') = FR;
@OLE('C:\Data\Desktop\example tow.xls','delr') = delr;
@OLE('C:\Data\Desktop\example tow.xls','dr') = dr;

END DATA

!OBJECTIVE FUNCTION;

Min=@sum(ProPer(j,t):SDR(j,t)*delr(j,t)+RDR(j,t)*dr(j,t)+INR(j,t)*FR(j,t));


!SUBJECT TO;
@for(ProPer(j,t):@BIN(delr(j,t)));
@for(ProPer(j,t):@GIN(FR(j,t)));
@for(ProPer(j,t):@GIN(dr(j,t)));

!1;

@for(ProPer(j,t)| t #GT#1 : FR(j,t-1) + dr(j,t) - FR(j,t) = RR(j,t));
@for(Product(j) : dr(j,1)- FR(j,1) = RR(j,1));

!2;
@for(ProPer(j,t): dr(j,t) <= M*delr(j,t));


!3;
@for(ComPer(i,t): X_bar(i,t)>= @SUM(Product(j):BR(i,j)*dr(j,t)));
```