A HEURISTIC APPROACH FOR PROFIT ORIENTED DISASSEMBLY LOT-SIZING PROBLEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MELİKE KAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

FEBRUARY 2011

Approval of the thesis:

A HEURISTIC APPROACH FOR PROFIT ORIENTED DISASSEMBLY LOT-SIZING PROBLEM

submitted by **MELİKE KAYA** in partial fulfillment of the requirements for the degree of **Master of Science in Department of Industrial Engineering**, **Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Director, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Sinan Kayalıgil _____
Head of Department, **Industrial Engineering**

Assist. Prof. Dr. Z. Pelin Bayındır _____
Supervisor, **Department of Industrial Engineering, METU**

Assist. Prof. Dr. Ferda Can Çetinkaya _____
Co-Supervisor, **Industrial Engineering Dept., Çankaya University**

Examining Committee Members:

Prof. Dr. Sinan Kayalıgil _____
Department of Industrial Engineering, METU

Assist. Prof. Dr. Z. Pelin Bayındır _____
Department of Industrial Engineering, METU

Assist. Prof. Dr. Ferda Can Çetinkaya _____
Department of Industrial Engineering, Çankaya University

Prof. Dr. Meral Azizoğlu _____
Department of Industrial Engineering, METU

Assist. Prof. Dr. Sedef Meral _____
Department of Industrial Engineering, METU

Date: **11.02.2011**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Melike KAYA

Signature :

ABSTRACT

A HEURISTIC APPROACH FOR PROFIT ORIENTED DISASSEMBLY LOT-SIZING PROBLEM

Kaya, Melike

M.Sc., Department of Industrial Engineering

Supervisor: Assist. Prof. Dr. Z. Pelin Bayındır

Co-Supervisor: Assist. Prof. Dr. Ferda Can Çetinkaya

February 2011, 68 pages

In this thesis, we work on a disassembly lot-sizing problem for multiple products with parts commonality, i.e., general product structure. We assume that supply of discarded products is infinite. When a product (or a subassembly) is disassembled, all its immediate child items are obtained, i.e., complete disassembly case. Intermediate and leaf items obtained are demanded by external suppliers or remanufacturers. The maximum possible sales for each intermediate and leaf item are known. Sales of the intermediate and leaf items are the revenue sources. The discarded products are purchased at a unit purchasing cost. The disassembly operation incurs a fixed and a variable disassembly cost. Due to this cost structure, intermediate and leaf items can be stocked incurring an inventory holding cost. We develop an integer programming formulation to determine the time and quantity of the discarded products to be purchased; the time and quantity of the discarded products and the intermediate items to be disassembled; and the time and quantity of intermediate and leaf items to be sold in order to maximize the total profit over a finite planning horizon.

We state that our problem is NP-hard by refering the study of Kim et. al. (2009). We

propose a heuristic solution approach that solves the problem in a reasonable computational time and generates near optimal solutions. The solution approach is based on the idea of sequentially solving a relaxed version of the problem and one-period integer programming models. In a computational study, the performance of the heuristic approach is assessed for a number of randomly generated problem instances. The results of the computational study show that the solutions of the heuristic approach are very close to the optimal and the best feasible solutions obtained within the time limit.

Keywords: Part/Material Recovery, Disassembly Lot-Sizing, Integer Programming

ÖZ

KAR AMAÇLI DEMONTAJ PARTİ BÜYÜKLÜĞÜ PROBLEMİ İÇİN SEZGİSEL YÖNTEM

Kaya, Melike
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Yrd. Doç. Dr. Z. Pelin Bayındır
Ortak Tez Yöneticisi: Yrd. Doç. Dr. Ferda Can Çetinkaya

Şubat 2011, 68 sayfa

Bu çalışmada ortak parça içeren çoklu ürün yapıları için demontaj (söküm) parti büyüklüğü problemi ele alınmıştır. Parçalarına ayrılacak ürünlerin arzının sınırsız olduğunu varsaydık. Bir ürün parçalarına ayrıldığında, o ürünün bir alt seviyedeki tüm parçaları elde edilir. Elde edilen ara ürünler ve son seviyedeki parçalar tedarikçi ya da yeniden üreticiler tarafında talep edilir. Ara ve son seviyedeki parçaların mümkün olan en yüksek satış miktarları bilinmektedir. Ara ve son seviyedeki parçaların satışı gelir kaynaklarıdır. Parçalarına ayrılacak ürünler birim satın alma maliyeti ile alınır. Söküm operasyonu bir bağımsız sabit ve bir değişken söküm maliyetleri ile yapılır. Bu maliyet yapısından dolayı, ara ve son seviyedeki parçalar envanter taşıma maliyeti karşılığında stokta tutulabilir. Parçalarına ayrılacak ürünlerin satın alınma zamanı ve miktarını, parçalarına ayrılacak ürünlerin demonte edilme zamanı ve miktarını, ara ve son seviyedeki parçaların satış miktarlarını belirleyen, belirli planlama ufku için toplam karı da enbüyükleyen bir tam sayı programı oluşturuldu.

Problemimizin polinom olmayan zor bir problem olduğunu Kim et. al. (2009)

çalışmasına dayanarak gösterdik. Problemi makul bir sürede ve en iyi çözüme yakın çözebilecek bir sezgisel çözüm yaklaşımı önerdik. Çözüm yaklaşımı, gevşetilmiş problem yapısı ve tek dönemlik tam sayı problemlerini sırayla çözme fikrine dayanır. Deneysel çalışmada sezgisel yaklaşımın performansı rassal olarak oluşturulan örnek problemler için test edildi. Test sonuçları, sezgisel yöntemin optimal çözümlere ve zaman sınırı içinde mümkün olan en iyi tamsayı çözümlere çok yakın olduğunu göstermiştir.

Anahtar Kelimeler: Parça/Malzeme Gerikazanımı, Demontaj Parti Büyüklüğü, Tam Sayı Programlaması

To my Mum

ACKNOWLEDGEMENTS

I would like to thank all those people who have helped in the preparation of this study. I am grateful to my supervisors Assist. Prof. Dr. Z. Pelin Bayındır and Assist. Prof. Dr. Ferda Can Çetinkaya for their guidance, advice, criticism, encouragements and insight throughout this study.

I would like to thank jury members for their valuable contributions on the thesis.

I would also like to thank Tülin İnkaya and Banu Lokman for their valuable support and time they spared for me.

And, I would like to thank my family for their endless support, hope, care and help. I would like to thank my dear friend Ayşe Özmen for her patience and support. I would like to dedicate this study to my all family whom I am proud of.

I would like to thank TÜBİTAK for the funding they have provided during my M.S. study.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Disassembly is a methodical extraction of parts and materials from discarded products through a series of operations. Disassembly is the first step of all product recovery options. Retrieved components can be reused and materials are recycled. Value hidden in discarded products encourages producers to be engaged in product and material recovery. Therefore producers should make effective plans to recover all possible value and increase the profitability of the recovery practice.

Disassembly can be complete if the product is fully disassembled or partial if only some parts and subassemblies are removed (Güngör and Gupta, 1999). The disassembly operation can be destructive (products are destroyed/damaged because of focusing on materials recovery) or non-destructive (causing no damage on the products or the parts because of focusing on components recovery). The disassembly process can be supply or demand driven. In the demand-driven disassembly, the discarded products are disassembled to fulfill the demand with regard of cost efficiency. In the supply-driven disassembly, the discarded products are to be disassembled but it is not necessary to meet all demand.

Compared to the assembly, the disassembly has the characteristic that a product diverges into multiple demand sources of parts/components. (Note that parts/components converge into the single demand source of a product in the assembly system.) This makes the disassembly problems more complex than the ordinary production planning problems in assembly systems (Gupta and Taleb,1994).

As in the case of assembly process, planning tools are needed to manage disassembly operations more profitably.

In this study, we consider a disassembly firm that receives the discarded products and disassemble them into their components to satisfy the demand of external parties that use these components in building new products; remanufacturing and refurbishing activities. The discarded products (that are referred to as root items throughout the thesis) are purchased from external collectors incurring a unit purchasing cost. We assume that the supply is ample for them. The demand occurs for both subassemblies (intermediate items in the bill-of-material structure) and leaf items (parts that are at the lowest level of the bill-of-material structure. Note that they cannot be further disassembled.). We assume a complete disassembly structure, that is, when a product (or a subassembly) is disassembled, all its immediate child items are obtained. When a root or an intermediate item is disassembled, a fixed and a variable cost are incurred. Due to this cost structure, immediate and leaf items can be stocked incurring an inventory holding cost in the system.

We specifically study the disassembly lot-sizing problem that aims to maximize total profit over a finite planning horizon. It is assumed the the supply of root items is infinite. There is no capacity restriction on the disassembly operation. Maximum possible sales for the disassembled items in each period are known. No defective items are obtained during the disassembly operations, i.e. all parts are of perfect in quality. The demand, cost and revenue parameters are deterministic and known. Cost and revenue parameters are time invariant.

The rest of the thesis is organized as follows: in Chapter 2, we review the studies in the literature that are most related with our study. Uncapacitated problems with assembly structures, uncapacitated problems with general structure and capacitated problems with assembly structures are reviewed. We define our problem and give its mathematical formulation in Chapter 3. The assumptions in the model formulation

are provided and the mathematical formulation for the problem considered is described. We state that our problem is NP-hard by refering the study of Kim et. al. (2009). When the size of the problem is increased (by increasing the number of periods in the planning time horizon and the number of items), the solution time of the integer programming model is exponentially increased. Therefore, a heuristic algorithm is presented to find near optimal solutions to the problem within reasonable computational time. The proposed heuristic is based on the idea of sequentially solving a relaxed problem and the original integer programming model for a single period. Since the relaxed problem generates near optimal solutions, we round down the fractional solutions into integer variable and use them as bounds for the variables in the one-period integer model. Thus, we can reach near optimal solutions using the heuristic approach. In Chapter 4, we present the computational study carried out and discuss the performance of the solution approach. We conclude the study and give suggestions for future research in Chapter 5.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we explain the product structures considered, define the disassembly lot-sizing problem and review the most related studies with our study in the disassembly lot-sizing literature.

In a product structure, the root item represents the end-of-use/life product to be purchased and disassembled, and the leaf items are the parts or components not to be disassembled further. Different parents can have the same child item(s). More than one unit of a child item can be obtained when a parent is disassembled. The former situation is called commonality, and the later is called multiplicity feature of a product structure. Disassembly scheduling problems in the literature considers (i) a single product structure without parts commonality (assembly product structures); or (ii) multiple products with parts commonality (general product structures).

Figure 2.1 shows an example for assembly structure in which item 0 denote the root item. Items 1, 2, and 3 are the intermediate items and items 4-8 are the leaf items. The example does not include any common items. Figure 2.2 shows an example for general product structure, in which items 0 and 1 denote the root items and items 2–9 except item 3 denote leaf items. Item 3 is an intermediate item. The numbers in parenthesis represent the number of units (yield) of a certain child item obtained when one unit of its parent is disassembled. Items 4 and 9 are the common items: Items 0 and 1; and items 1 and 3 are the parent items of items 4 and 9, respectively.

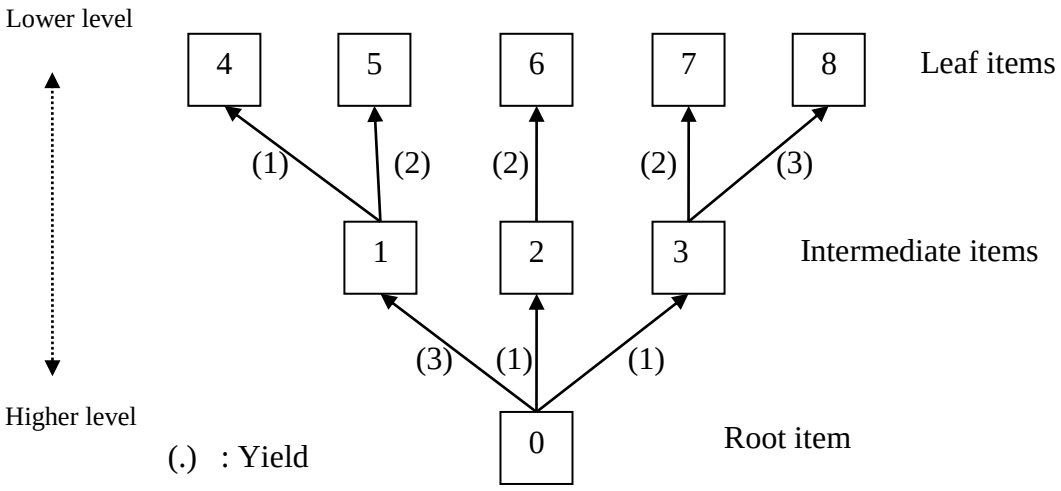

Figure 2.1. An example for assembly structure

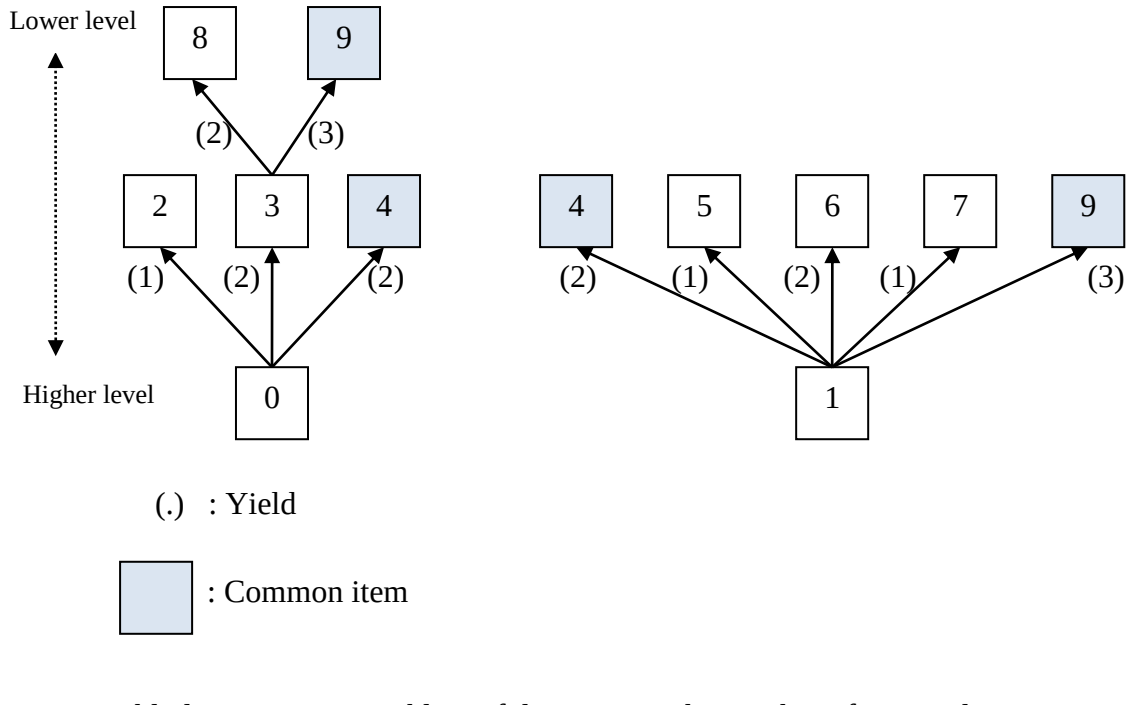

Figure 2.2. An example for general product structure

Disassembly lot-sizing is a problem of determining the number of root and intermediate items to be disassembled to fulfill the demand for leaf items over a given finite planning horizon with discrete time periods. The possible objectives are minimizing total cost, minimizing the quantity of discarded products to be disassembled and maximizing total profit. Cost minimization objective is more extensively studied in the literature.

A recent literature review on disassembly-lot sizing problem is presented by Kim et. al. (2007). Problem characteristics of all studies related to the disassembly lot-sizing and solution approaches are discussed. Problem types are categorized and some possible extensions and integrated decision problems are described with future research directions. In the review, disassembly lot-sizing problem are classified as follows:

- Uncapacitated problem with assembly structure (basic problem)
- Uncapaciteted problems with general structure
- Capacitated problem with assembly structure
- Problems with uncertainity

We review the studies that are in the first three classes above which are most related to our study.

2.1. Uncapacitated problem with the assembly structure

The basic problem considers a single product without parts commonality. Capacity issue is not considered in the basic problem. It is first studied by Taleb and Gupta in 1994. A reverse material requirement planning (MRP) problem is presented to determine the quantity and schedule of the root item to be disassembled in order to fulfill the demand for components over a finite planning horizon. The assumptions employed are similar to the assumptions of MRP:

(i) gross requirements for the leaf items, scheduled receipts from external sources are known with certainty,
(ii) lead times are constant and irrespective of the lot-size,
(iii) perfect disassembly is assumed, i.e. no defective parts are generated,
(iv) capacity is not considered explicitly.

The algorithm suggested starts with calculations for leaf item for the first period. Its net requirement, on hand before disassembly, gross requirement disassembled for its parent item, schedule receipts from disassembly and on hand after disassembly for its brother items are calculated. Then time index is increased by one. If it is not the last planning period, the procedure is applied to the remaining brother items. If it reaches the last planning period, disassembly schedule is calculated for the parent item. Then calculation is made for the other end items. Therefore, schedules for all parent items are calculated over the planning horizon. The disassembly schedule for the root item is determined by using disassembly schedule of its child items. The algorithm is applied to a disassembly structure by a spreadsheet implementation.

Adenso-Diaza et.al. (2008) address the lot sizing problem in reverse Material Requirement Planning (MRP) for scheduling disassembly. The data needed for reverse MRP problem are information about the product structure, disassembly lead times for each parent item, ordering lead time for the root item, and the planning horizon. Gross requirement for leaf items, external scheduled receipts and inventories at the beginning of the planning horizon, set-up cost for parent items and ordering cost for root items are known. They apply the period order quantity (POQ) lot-sizing technique to the items from the bottom level (leaf item) to the upper level (root item). Developed scenarios are experimented using three additional lot-sizing methods: lot for lot (LFL), best disassembly schedule in each subassembly (BIES) and best combination (BC). BIES corresponds to choosing the best disassembly option (based on cost) at each subassembly and at the root item independently. BC refers to choosing the best disassembly schedules by considering all the possible

combinations of lot sizes and choosing the combination with the lowest global cost. In their study, lot-sizing techniques are determined for disassembly levels. Different lot-sizing rules behave differently when applied at various levels in the Bill-of-Material (BOM) structure. They design full factorial experiments for randomly generated product structures. They divide the BOM into four parts, viz., lot-sizing policy in the level just below the components with external demand (factor LS1), the intermediate subassemblies (factor LS2), in the disassembly process of the final product (factor LS3), and in the ordering of the final product from suppliers (factor LS4). They consider two disassembly levels, three set-up cost levels, three inventory cost levels and one ordering cost for the root item. Number of levels, set-up cost, total cost (TC) and LS factors are the factors tested and their interactions are analyzed.

The results are summarized as follows:

- LFL, Economic Order Quantity (EOQ) and POQ show significant differences relative to TC in factors LS1 and LS3. However, for intermediate subassemblies (LS2), no significant differences between EOQ and LFL or POQ exist.
- For the upper level of the BOM (LS1), the interaction between the three lot-sizing rules (LFL, EOQ, POQ) and the number of levels in BOM show that as the number of levels increases, so does the TC for all the lot-sizing rules tested, especially LFL, which is the worst alternative, independently of the complexity of the structure. POQ turns out to be the best option, followed by EOQ.
- Lot-sizing rules POQ and EOQ are cost-effective in planning disassembly operations at upper levels, while LFL is the worst performer.
- At the intermediate levels, the lot-sizing rule chosen to organize the production plan does not play a significant role to reduce the system costs.
- At the highest level (final product disassembly), LFL turns out to be the most effective rule, especially in simple structures. When the complexity of the structure increases, LFL or POQ are both economically viable.

- The way the final products are ordered (LS4) has no significant effect on cost effectiveness.

Lee et. al. (2004a) present on three integer programming models that are for the case of a single product type without parts commonality, the case of a single product type with parts commonality and the case of multiple product types with parts commonality. This study starts with the basic problem formulation and the remaining two problems are formulated considering multiple products and the commonality feature. Following assumptions are employed.

- Disassembly structure is given in advance from the corresponding disassembly process plan that specifies all parts/subassemblies.
- There is no shortage of the root items; i.e. products can be obtained whenever they are ordered.
- The demand for the leaf items is given and deterministic.
- Disassembly lead times with discrete time scale are given and deterministic.
- Backlogging is not allowed and hence demands are satisfied on time.
- Parts/components are perfect in quality, i.e. no defective parts are considered.

The objective of all models is to minimize the sum of root purchasing cost, set-up cost, and inventory holding cost for all items and disassembly operation costs. In order to test the performance of the integer programming models, firstly the problem instances given by Taleb and Gupta (1997) are used. Solution of integer programming models is compared with the solution of heuristic approach presented by Taleb and Gupta (1997). Optimal solutions for the existing problems are obtained in reasonable computational times. Secondly, randomly generated problem instances from small to medium sizes are used to test the performance of integer programming models more generally. Performance measures are the number of problems solved optimally in 3600 second time limit, CPU seconds and percentage deviation from the lower bound (for problems that cannot be solved in 3600 seconds). 825 problem

instances out of 900 problems are solved optimally by CPLEX 6.5 in 3600 seconds time limit. As the number of items and/or the number of periods increase, per cent deviation from the lower bound and the computational times increase. The percentage deviations from the lower bounds are within 1 % on the average.

Lee et.al. (2004b) develop a two-stage heuristic algorithm for the problem considered. Assumptions and the objective of the integer programming model are the same with the study of Lee et. al. (2004a). They develop a two-stage heuristic which finds the initial feasible solution using Gupta and Taleb (1997) (GT) algorithm at the first stage. In the second stage, backward and extreme (pairwise) move improves the initial solution by considering trade-offs among different cost factors. In this method, a later disassembly operation is moved to an earlier period. It is extreme because the all disassembly quantity of a later period is moved to an earlier period. After all, an improvement procedure is applied in order to find best backward and extreme movement. Backward and extereme move can result in backlogging for items. Therefore, feasibility of the new disassembly schedule is checked. New feasible disassembly schedule gives new ordering schedule and new inventory levels. Random problem instances are generated to measure the performance of the two-stage heuristic algorithm. Time variant and time invariant purchasing costs are considered in the computational experiments. Results of the heuristic algorithm are very close to optimal solutions within very short computational times. Solution quality for the case of time-invariant purchasing cost is better (very close to the optimal solution) than the case of time-variant purchasing cost.

Kim et. al. (2009) consider the problem that minimizes the total cost that is sum of set-up cost and inventory holding cost. It is proved that the problem is NP-hard by reducing it to the Joint Replenishment Problem (JRP). Two properties of the optimal solution are derived. The first one is the extended zero inventory property that if at least one of the children of a parent has positive inventory on hand, then the parent cannot be disassembled for the following period. The second property describes the condition that the disassembly of a parent item in a period implies the disassembly

(or demand) of its child items in that period. It is an extension of the nestedness property. An integer programming model is reformulated by eliminating the inventory variables and adding a constraint, that is, the disassembly quantity of a parent item should at least be equal to the demand of its succeeding leaf items.

They suggest a branch and bound algorithm that incorporates the Lagrangean relaxation technique to obtain good lower and upper bounds. As in the ordinary branch and bound algorithms, each node of the branch and bound tree is fathomed, if the lower bound at the node is greater than or equal to the incumbent solution value. In order to test the performance of the algorithm, problem instances are randomly generated and solved. Branch and bound algorithm provides optimal solution up to moderate-sized problems within a reasonable amount of computational time. It requires less computational time than a standard solver (CPLEX). CPU time increases as the problem size increases and also lower bound gets tighter. When set-up cost is high, the subproblems may have tighter lower bounds and so can be fathomed earlier. Lagrangean heuristic is suggested as an alternative for large-sized problems. Computational results show the Lagrangean heuristic outperforms other heuristics which are Taleb and Gupta (1994)-reverse MRP and Lee et. al. (2004b)-two stage heuristic.

2.2. Uncapacitated problems with general structure

In this case, multiple products with parts commonality are considered. Capacity restrictions are still ignored.

Taleb and Gupta (1997) extend the study of Taleb and Gupta (1994) by considering cost issues and more complex product structures. The demand for each leaf item is known. Ordering lead time for root items and disassembly lead time for parent items are given. Cost parameters are item cost (acquisition cost), separation cost for parent items, and disposal cost for the leaf items. Total cost is to be minimized over a finite planning horizon. Two consecutive algorithms, core and allocation algorithms, are

developed to solve the problem. Core Algorithm determines the number of root items to be disassembled by minimizing the total cost. It determines the minimal disassembly order for each root items based on Gross Requirements of non-common leaves. Then, it determines unfulfilled requirements for all common leaves. Parent items are decided to be disassembled to satisfy common leaves by considering cost benefit ratio that is the cost over total number of yields obtained. Then, a feasible schedule is determined. In order to find a better solution, the feasible solution is improved by decreasing disassembly order by using improvement factor which is based on improvement in objective function. Release of root items are scheduled over the planning horizon. Allocation Algorithm determines a disassembly schedule by delaying disassembly as much as possible. Therefore it implicitly minimizes the holding cost. The algorithm first determines the nearest time period with net requirements for the leaf items. Next, most attractive root items which result in the most decrease in requirements of leaf items are selected. After that, the requirements of leaves are updated. The procedure continues until all requirements are fulfilled for that time period. The entire procedure is repeated for all periods. Finally, it provides a disassembly schedule for the subassemblies. In order to evaluate the performance of the heuristic algorithm, they randomly generate 25 problem instances. The proposed algorithm solves 19 cases optimally. The deviations from the optimal solution for the remaining problem instances are between 1.4% and 3.8%.

Langella (2007) considers the problem given in Gupta and Taleb (1997). Since Core and Allocation Algorithm by Taleb and Gupta (1997) can sometimes give infeasible solutions, additional parameters and options are considered by Langella (2007). Demand for leaf items can be satisfied by the procured leaf items in addition to the disassembled parts. In this study, it is also considered that all items can be held in inventory and all items except the root items can be disposed of. An integer programming model is formulated to determine the quantity of discarded products procured, disassembled, the quantity of leaf item procured and the quantity of intermediate and leaf items held or disposed of. The model aims to minimize total

cost that includes root procurement cost, leaf procurement cost, disassembly cost, holding cost and disposal cost over a finite planning horizon. Constraints are as follows:

- Initial inventory levels
- Inventory balance constraints
- Limited supply of root items
- Upper bounds on ending inventory of intermediate and leaf items

Langella (2007) presents Integral Algorithm that incorporates holding cost, external leaf procurement and leaf holding versus disposal. Leaf procurement is preferred if the sum of the purchasing cost of leaves is less than the additional cost of disassembly. Holding cost versus disposal of a leaf is recommended as a remedy. The algorithm starts with calculations of the net requirements for the first period. A ratio that is amount of demand satisfied by root over additional costs is calculated for each feasible root (available in time that is met demand). The quantity of the root item to be disassembled with the highest ratio is increased by one. Then, it is decided whether procuring leaves externally is considered or not. Net requirements of the items are updated for the following time period. The algorithm continues until all requirements are satisfied. Performance of the heuristic is tested on randomly generated problem instances. Average cost deviation from the optimal solution is around 4.5 %. More specifically, in 147 out of 250 problem instances the deviation from the optimal solution is less than 5%; with an average of 1.2%. However the heuristic exhibits poor performance in some problem instances. In 41 problem instances, the deviation from the optimal solution is more than 10% with an average of 15%.

Kim et. al. (2006a) develop a two-phase heuristic for the problem of disassembly scheduling. Integer programming model for the problem aims at minimizing the sum of set-up, disassembly operation and inventory-holding costs. In the proposed

solution approach, linear programming relaxation gives the initial solution in the first stage. Continuous variables in the relaxed solution are rounded down. The rounded down solution may violate the inventory balance constraints. In order to satisfy the feasibility, values of the decision variables are increased or decreased by considering the cost changes. The procedure starts with the last item and finishes when the root item is reached. In the second stage, a dynamic programming algorithm with look-ahead check is used to improve the initial solution. Dynamic programming model improves the current feasible solution by solving v-period sub problem in the forward direction (starting from period v=1 and ending in period v=T), and find the new disassembly schedule of an item. Possible cost changes are calculated for the possible disassembly schedules and the schedule which provides more improvement on the current feasible solution is chosen. This procedure is applied to all parent items, starting from the root item. If there is no further improvement to be made for all parent items, the second stage is terminated. Randomly generated problem instances from small to large sizes are used to test the performance of the solution approach. The second phase of the solution approch improves the initial feasible solution by 6.6%, 4.8% and 3.6% on the average for the cases of 10, 20, and 30 periods for high set-up cost case, respectively. The initial solution is significantly improved in the second phase. For the problem instances with 30-period, overall average deviations from the optimal solution (or lower bound) is 0%, 0.3% and 1.6% for the case of low, medium and high set-up cost, respectively. Computation time for the initial phase is between 0.01 and 0.17 seconds, while the second phase requires time between 0.01 and 7.43 seconds. Computational experiments show that the second phase solution time is slightly longer than the first phase solution time.

2.3. Capacitated problem with the assembly structure

The basic problem is extended by considering resource capacity constraint. The capacity restriction is considered in the form of time limit for the disassembly operations performed in that period. There is an upper limit on the available time in

each period of the planning horizon and each disassembly operation assigned to a period consumes a portion of the available time.

Lee et. al. (2006) develop an integer programming model considering capacity restriction explicitly. The objective is to minimize the number of products disassembled. Total operation time for a planning period is limited with processing time capacity in each period of the planning horizon. A two stage solution approach is proposed. In the first stage, uncapacitated disassembly scheduling problem is solved by relaxing the capacity constraint. Initial solution is obtained by the Taleb and Gupta (1994) algorithm. An initial solution is called minimal latest disassembly schedule since it satisfies the demands of leaf items as late as possible with the minimum disassembly quantities. In the second stage, current disassembly schedule is changed by considering the capacity constraint. Disassembly quantities are moved to earlier time periods without increasing initial objective function value. Quantity of moved items is chosen without violating time capacity for that period. Computational experiment consists of a case study and randomly generated problem instances. Loose and tight capacity options are considered in the computational study. Solution time of the algorithm is very short for the case study, i.e. within 0.005 seconds on average. Results show that the tight capacity option requires more computation time than the loose capacity option. All problem instances are solved to optimality. Heuristic algorithm also requires much smaller computation time -from 0.3 to 11 seconds- to solve the randomly generated.

Kim et. al. (2006b) consider the same environment studied by Lee et. al. (2006) with the objective of minimizing sum of set-up, disassembly operation, and inventory holding costs. A two stage heuristic approach is proposed. In the first step, Lagrangean relaxation technique is used. Capacity constraint and demand constraint are relaxed. Then, the model is reformulated by rewritten demand constraint to make a stronger formulation. After the relaxation, the model turns into several single-item lot sizing problems. Lower bounds are obtained by solving single-item lot sizing

problems. Subgradient optimization technique improves the solution. However, the solution may not be feasible due to demand and capacity constraint violation. In order to get a feasible solution backward and forward moves are applied to the initial solution. In backward and forward move, overloaded disassembly quantities are moved to earlier periods or to later periods, respectively. In both movements the best movement is chosen by considering minimum cost increase. It is an iterative procedure repeated until a feasible solution is obtained. Performance of the algorithm is tested by solving randomly generated problem instances. Two capacity cases, loose capacity and tight capacity, are analyzed. Results show that solution quality of the algorithm is affected a little by the capacity tightness. Overall percentage deviations from the optimal solutions are 0.18% and 0.46% for the cases of loose and tight capacity, respectively. The algorithm gives better solutions for longer planning horizon problem instances. Computation time of the heuristic algorithm is very short (4-5 seconds on the average) while many problem instances require 7200 seconds to reach optimal solutions.

Up to now, we have reviewed the most related studies to our study. The studies assume yield of the products known with certainty. Quality uncertainty, results in the fact that when we disassemble a certain core, we are not sure how many "good quality" leaves we will obtain. It is an important issue for remanufacturing and other recovery operations. Recently, Inderfurth et. al. (2006) consider product structure with multiple roots with parts commonality for complete disassembly case with stochastic yield.

Profit related issues are rarely studied in literature. Clegg et. al. (1995) and Spengler et. al. (2003) study profit-based disassembly lot-sizing problems. These studies consider integrated sytems. Clegg et. al. (1995) consider a remanufacturing facility where discarded products are totally or partially disassembled or disposed of. Disassembled parts are recovered and used as components and materials can be recycled. Disassembled components are sent to the assembly line for

remanufacturing. In addition to remanufacturing, new products are also produced on the assembly line.

Spengler et. al. (2003) presents a case study on electronic scrap recovery. A short term (daily) integrated recovery planning problem is worked on. They consider a system limited to a recovery center, not including the storage center. The first step is the disassembly operation which is composed of manual and partially automated processes. The second step is bulk recycling designed to recover precious fractions from mixed electronic scrap. The scrap disassembled is either used internally in bulk recycling or marketed externally. A mixed-integer linear programming (MILP) model is formulated to determine the amount of scrap to be recovered, disassembled and used in internal or external operations to maximize marginal income.

In addition to profit-based studies, multi objective issue is also studied in the disassembly lot-sizing literature. Kongar and Gupta (2002) consider a disassembly-to-order (DTO) system where end-of-life products are procured from the final users and collected products are disassembled in a disassembly plant. Destructive and non-destructive disassembly processes are both considered. After the destructive disassembly, components are either recycled or disposed of. On the other hand, components are either reused, stored, recycled or disposed of after the non-destructive disassembly operation. The aim is to determine the best combination of multiple products to selectively disassemble to meet the demand for items and materials under a variety of physical, financial and environmental constraints and goals. Preemptive goals to be achieved are the followings:

- maximum total profit,
- maximum sales from materials,
- minimum number of disposed items,
- minimum number of stored items,
- minimum cost of disposal

- minimum cost of preparation, in that order.

In our study, we consider multiple products with parts commonality, i.e. general product structure. The most closely related study to ours is the study by Kim et. al. (2006a). Our study differs from their study in the followings.

- We consider demand for intermediate items in addition to the demand for leaf items.
- Demand for intermediate and leaf item is the maximum possible sales quantity of these items. It is not required to satisfy the demand fully.
- There is no explicit cost of not satisfying demand other than lost profit margin.
- Kim et. al. (2006a) aim to minimize the total cost which is sum of set-up cost, disassembly operation cost and inventory-holding costs over a finite planning time horizon. In addition to these parameters, we consider unit purchasing cost of root items and unit sales price for intermediate and leaf items.
- Our study aims to maximize total profit over a finite planning time horizon.

CHAPTER 3

PROBLEM DEFINITION AND SOLUTION APPROACH

In this chapter, we describe our problem environment and define the problem. In Section 3.1, we give the assumptions of the disassembly lot-sizing problem considered. In Section 3.2, an integer programming formulation for the problem is provided. In Section 3.3, the proposed heuristic solution approach for the problem is given and discussed. Finally, we illustrate our solution algorithm on a numerical example in Section 3.4.

In this study, we consider a disassembly firm that receives collected used products and disassembles them to satisfy the demand for their components which are the items that occur in the lower levels of BOM structure of used products. The root items are purchased from external collectors incurring a unit purchasing cost. We assume that the supply of them is ample. The root items are not kept in stock; they are disassembled at the time that they are purchased since their supply is ample and there is no fixed purchasing cost. These items have general structures, i.e. both parts commonality and multiplicity which are the features of a product structure defined at the beginning of Chapter 2. When an item (a root or an intermediate one) is disassembled, all of its immediate child items are obtained. Intermediate items can be either directly sold to the customers or disassembled further to obtain the items that are at the lower levels of BOM structure. The disassembly operation incurs a fixed and a unit variable cost. Fixed disassembly operation costs are independent. Due to this cost structure, it is allowed to stock intermediate items incurring an inventory holding cost per unit per period basis. Intermediate and leaf items are demanded by external suppliers or remanufacturers for building new products; remanufacturing

and refurbising activities. Sales of the intermediate and leaf items are the sources of revenue. Unit price of the disassembled items are set by the market. The demand forecasts for intermediate and leaf items over a finite planning horizon are available. In this study, these forecasts are treated as the maximum possible sales quantities. There is no penalty cost of not satisfying maximum demand other than the lost profit. Demand, cost and revenue parameters are known and deterministic. The cost and revenue parameters are time invariant.

We consider the problem that maximizes the total profit which is the difference between total revenue and total cost, over a given planning horizon with discrete periods. The basic decisions at each planning period are as follows:

- The quantity of the root items purchased and disassembled
- The quantity of intermediate items disassembled
- The quantity of all items except the root items kept in inventory
- The quantity of the intermediate and leaf items sold

3.1. Assumptions

The following assumptions are employed in the model formulation.

A1. The supply of root items is infinite.

A2. Complete disassembly case is considered. When an item is disassembled, all of its immediate child items are obtained.

A3. The demand quantities, cost and revenue parameters are deterministic and known. Cost and revenue parameters are time invariant, i.e., they don’t change from one period to another.

A4. Backlogging is not allowed. Unsatisfied demand is lost.

A5. Root items are purchased and immediately disassembled. Since the supply of the root items is infinite and there is no fixed purchasing cost, they are not kept in inventory. Only intermediate and leaf items are kept in inventory due to assumed disassembly cost structure.

A6. There is no capacity constraint on disassembly operations.

A7**.** No defective intermediate and leaf items are obtained during the disassembly operations, i.e. parts are of perfect in quality.

A8. There is no yield loss, i.e. we release all intermediate and leaf items which exist in BOM of the root items.

A9. All disassembly lead times are less than a period.

3.2. Mathematical Model

In this section, we first define the indices, parameters and decision variables. Then, we give the mathematical programming model of the problem.

Indices and sets

t = index for periods, $t = 1, 2, \ldots, T$, where T is the length of the planning horizon

i_r = index for the last root item

i_l = index for the first leaf item

i = index for items, $i = 1, 2, ..., i_r, i_r+1, ..., i_l-1, i_l, i_l+1, ..., N$, where N is the total number of items

$\Phi(i)$ = set of immediate-parents of item i

$H(i)$ = set of immediate-children of item i

Parameters

I_{i0} = initial inventory of item i

λ_{ki} = quantity of item i obtained from disassembling one unit of item k

d_{it} = demand for item i in period t

h_i = unit inventory holding cost per period for item i

p_i = unit disassembly cost for item i

a_i = unit purchasing/acquisition cost of root item i

f_i = fixed cost of disassembly for (parent) item i

r_i = unit sales price of item $\boldsymbol{i}$

M = a large positive number

Decision variables

X_{it} = quantity of item i disassembled in period t

I_{it} = inventory of item i at the end of period t

S_{it} = quantity of item i sold in period t

$$Y_{it} = \begin{cases} 1 \text{ if a disassembly set-up is done for item } i \text{ in period } t \text{ i.e., if } X_{it} > 0. \\ 0 \text{ otherwise} \end{cases}$$

Integer programming model (P) of the problem is given below:

$$(\mathbf{P}) \text{ Max } \sum_{i=i_r+1}^{N}\sum_{t=1}^{T} r_i S_{it} - \left(\sum_{i=1}^{i_r}\sum_{t=1}^{T} a_i X_{it} + \sum_{i=1}^{i_l-1}\sum_{t=1}^{T} f_i Y_{it} + \sum_{i=1}^{i_l-1}\sum_{t=1}^{T} p_i X_{it} + \sum_{i=i_r+1}^{N}\sum_{t=1}^{T} h_i I_{it} \right) \quad (3.1)$$

Subject to

$$I_{it} = I_{i,t-1} + \sum_{k\in\Phi(i)} \lambda_{ki} X_{kt} - X_{it} - S_{it} \text{ for } i = i_r+1, i_r+2, \ldots, i_l-1 \text{ and } t = 1, 2, \ldots, T \quad (3.2)$$

$$I_{it} = I_{i,t-1} + \sum_{k \in \Phi(i)} \lambda_{ki} X_{kt} - S_{it} \qquad \text{for } i = i_l, i_l+1, ..., N \text{ and } t = 1, 2, ..., T \qquad (3.3)$$

$$X_{it} \leq M Y_{it} \qquad \text{for } i = 1, 2, ..., i_l\text{-}1 \text{ and } t = 1, 2, ..., T \qquad (3.4)$$

$$S_{it} \leq d_{it} \qquad \text{for } i = i_r+1, i_r+2, ..., N \text{ and } t = 1, 2, ..., T \qquad (3.5)$$

$$X_{it} \geq 0 \quad \text{and integer} \qquad \text{for } i = 1, 2, ..., i_l\text{-}1 \text{ and } t = 1, 2, ..., T \qquad (3.6)$$

$$I_{it} \geq 0 \quad \text{and integer} \qquad \text{for } i = i_r+1, i_r+2, ..., N \text{ and } t = 1, 2, ..., T \qquad (3.7)$$

$$S_{it} \geq 0 \quad \text{and integer} \qquad \text{for } i = i_r+1, i_r+2, ..., N \text{ and } t = 1, 2, ..., T \qquad (3.8)$$

$$Y_{it} \in \{0,1\} \qquad \text{for } i = 1, 2, ..., i_l\text{-}1 \text{ and } t = 1, 2, ..., T \qquad (3.9)$$

The objective function (3.1) maximizes total profit which is the difference between total revenue and total cost, over the planning horizon. The total revenue is obtained from the sales of the intermediate and leaf items. The total cost consists of purchasing cost of the root items, fixed and variable disassembly costs of root and intermediate items, and inventory holding cost of the intermediate and leaf items. Constraint sets (3.2) and (3.3) represent the inventory balance equations for the intermediate and leaf items, respectively. As it is stated in equation (3.2), the inventory level for each intermediate item is increased by the disassembly quantity of its parent(s) multiplied by the associated yield, and decreased by the sales quantity and disassembly quantity of the item further. The inventory balance equation (3.3) for each leaf item is the same with the equation (3.2) except the disassembled quantity of the item since the leaf items cannot be disassembled further. Constraint set (3.4) guarantees that a disassembly set up is made in a period when there is a disassembly operation in that period. Constraint set (3.5) represents that the limit on

the sales quantity of the intermediate and leaf items cannot exceed the corresponding demand. Finally, constraint sets (3.6) through (3.9) represent non-negativity and integrality restrictions on the decision variables. The quantity of disassembled items, stocked items and, sold items are the integer variables and, set-up variables are binary.

The integer programming model includes $(i_l - 1)T + 2(N - i_r)T$ integer variables and $(i_l - 1)T$ binary variables; and in total $2(N - i_r + i_l - 1)T$ decision variables. The model consists of $(i_l - 1)T + 2(N - i_r)T$ constraints.

Due to the fact that, the parameters, λ_{ki} are integer and the inventory balance equations, it is possible to relax the integrality constraints in 7-8. Under these relaxations, the solution to (P) is integer. In other words, representing the disassembly quantities, X_{it},as integer variable is enough to obtain an integer solution to (P).

Since the disassembly variable has still integrality restriction, it may require more computational time to solve the problems optimally. We do not choose this case in the computational study since we cannot guarantee to get benefits in terms of CPU time for the case. In the computational study, we solve (P) model that includes all integrality restrictions on all decision variables.

Selecting the value of parameter M in the constraint set (3.4) affects the computational burden for the model, since it defines the feasible region. One should restrict the value of parameter M with the smallest possible number in order to reduce the feasible region.

Let c_{it} be the cumulative future demand for item i in period t as given in Equation 3.10.

$$c_{it} = \sum_{t}^{T} d_{it} \quad \text{for } i = i_r + 1,\ i_r + 2,\ ...,\ N \text{ and } t = 1,\ 2,\ \ldots, T \tag{3.10}$$

Let M_{it} be a big-*M* value for item *i* at period *t*. M_{it} is calculated using cumulative future demand and yield information.

$$M_{it} = \max_{j \in H(i)} \left\{ \left\lceil \frac{c_{jt} + M_{jt}}{\lambda_{ij}} \right\rceil \right\} \quad \text{for } i = 1,\ 2,\ ...,\ i_l - 1 \text{ and } t = 1, 2, ..., T. \tag{3.11}$$

where $M_{it} = 0$ for $i = i_l,\ i_l + 1,\ ...,\ N$ and $t = 1, 2, ..., T$.

In the computational study, we take M as M_{it} that is calculated for the disassembled item i in period *t* by the Equation (3.11). Here, the calculations start from the parent items of the leaf items and reaches to the first root item at the end.

3.3. Problem Complexity

Kim et. al. (2009) consider the disassembly lot-sizing problem for single product type without parts commonality whereas we consider the problem for multiple product types with parts commonality. They prove that their problem is NP-hard by reducing it to the Joint Replenishment Problem (JRP). We can reduce our problem to the problem studied by Kim et. al.(2009) by considering single product without parts commonality and a case of constant sales revenue of our problem. Thus, our problem is reduced to cost minimization problem. As a result, our disassembly lot-sizing problem is also NP-hard. While commonality and multiplicity make very good sense from an economic and environmental standpoint, they certainly complicate the planning decisions. In our problem, intermediate items are also demanded in addition to the leaf items. The demand is just an upper bound for the sales quantity. It generates a larger feasible region since we cannot satisfy the demand in full. Objective function of our problem includes sales revenue, purchasing cost of root items and disassembly operation cost in addition to inventory holding cost and set-up cost. Our aim is to maximize total profit by selling obtained items from the

disassembled items. As a result, our problem is more complex than the problem considered by Kim et. al. (2009).

Complexity of integer programming (P) formulation increases exponentially with increasing the number of items (root, intermediate, and leaf) and the number of periods in the planning horizon. Therefore, it discourages us using the exact methods for solving the problem optimally. Thus, we provide a heuristic solution approach in order to find near optimal solutions in reasonable computational time.

3.4. Solution Approach

In this section, we propose a heuristic solution algorithm for the problem introduced in Section 3.2. At first, we present relaxed problem used in the solution algorithm. Before the desciption of the solution algorithm, we give the additional parameters and variables used in the algorithm. Then, the proposed solution algorithm is described.

We note that the problem (P) is a pure integer programming model that requires high computational time since it includes integer and binary decision variables. In order to reduce the computational time for the problem, we relax the integer variables which are disassembly variables (X_{it}), inventory variables (I_{it}), sales variables (S_{it}), and keep the set-up variables (Y_{it}) as binary. Specifically, we remove the integrality restrictions in (3.6), (3.7) and (3.8) of the model (P) and keep the non-negativity restrictions in (3.6), (3.7) and (3.8) as of the relaxed problem (RP). The non-negativity restrictions of the problem (RP) are represented by (3.11), (3.12) and (3.13).

$(\mathbf{RP})$ Max (3.1)

s.t. (3.2), (3.3), (3.4), (3.5), (3.9) and

$$X_{it} \geq 0 \quad \text{for } i = 1, 2, ..., i_l\text{-}1 \quad \text{and } t = 1, 2, ..., T \tag{3.12}$$

$$I_{it} \geq 0 \quad \text{for } i = i_r+1, i_r+2, ..., N \quad \text{and } t = 1, 2, ..., T \tag{3.13}$$

$$S_{it} \geq 0 \quad \text{for } i = i_r+1, i_r+2, ..., N \quad \text{and } t = 1, 2, ..., T \tag{3.14}$$

Our solution algorithm consists of T many (RP) problems and T many one-period (P) problems with additional restrictions on the decision variables which are the quantity of disassembly (X_{it}) and quantity of sales (S_{it}). The algorithm is based on the idea of sequentially solving the problem (RP) and one-period problem (P). The problem (RP), a relaxed version of the problem (P), is solved from period t to T. The solution of RP is an upper bound on the problem (P). We round down the fractional values of the solution to the problem (RP) in period t. The problem (P) is solved for that period (one-period) with the additional constraints which are obtained from the rounded-down solution to the problem (RP). We keep and save the solutions to T one-period (P) problems in order to obtain our heuristic solution.

We define the following additional parameters and variables used in the proposed heuristic solution.

S'_{it} = rounded-down sales quantity of the problem (RP) for item i in time t

X'_{it} = rounded-down disassembly quantity of the problem (RP) for item i in time t

S^*_{it} = optimal sales quantity of one-period problem (P) for item i in period t

X^*_{it} = optimal disassembly quantity of one-period problem (P) for item i in period t

I^*_{it} = optimal inventory level of one-period problem (P) for item i in period t

z^*_t = optimal objective function value for one-period integer programming model (P) for period t

z^* = optimal objective function value for T-period integer programming model (P)

z_H = objective function value for the heuristic solution

The solution algorithm is given below.

Step 0: Set t =1. Initialize the decision variables X_{it}, S_{it}, and I_{it} for all *i* and *t*.

Step 1: (Relaxation of problem (P))

Note that for period $t=1$, $I^*_{i,0}=0$ which is the beginning inventory for initial period provided as an assumption (A9) in Section 3.1.

1.1: Construct the relaxed problem (RP) by adding the optimal quantity of items kept in stock at the end of the previous period $t-1$, $I^*_{i,t-1}$, as a beginning inventory for period t. Solve the problem (RP) for the planning horizon from t to T and obtain its solution, X_{it}, I_{it}, and S_{it}. If $t=1$, go to Step (1.2), otherwise go to Step (1.3).

1.2: If X_{it}, I_{it}, and S_{it} for all *i* and *t* are integers, stop. The solution is optimal to the original problem (P). Set the solution $X_{it}=X^*_{it}$, $I_{it}=I^*_{it}$, and $S_{it}=S^*_{it}$. If the solution to the problem (RP) includes decision variables which have fractional values, go to Step (1.3).

1.3: Round down the fractional values of the solution to (RP) for period t, i.e. set $X'_{i,t}=\lfloor X_{i,t} \rfloor$ for $i=1, 2, ..., i_l-1$ and $S'_{i,t}=\lfloor S_{i,t} \rfloor$ for $i=i_r+1, i_r+2, ..., N$,

where $\lfloor \bullet \rfloor$ is the largest integer less than or equal to $\bullet$.

Step 2: (Single period (P) problem with additional restrictions)

Use rounded-down values for period t, $X'_{i,t}$ and $S'_{i,t}$, as parameters to restrict the decision variables $X_{i,t}$ and $S_{i,t}$ in the one-period model (P) for period t. Additional restrictions are as follows.

When $X'_{i,t}$ has a positive value, add the following constraint as a lower bound on variable $X_{i,t}$:

$$X_{i,t} \geq X'_{i,t} \qquad \text{for } i=1,\ 2,\ ...,\ i_l - 1$$

When $X'_{i,t}$ is equal to zero, add the constraint $X_{i,t} = 0$ for item i. An upper bound on the sales variable $S_{i,t}$ is as follows:

$$S_{i,t} \leq S'_{i,t} \quad \text{for} \quad i = i_r + 1,\ i_r + 2,\ ...,\ N$$

Solve integer programming model (P) for that period t (one-period problem) after adding the constraints above and the previous period's optimal inventory quantity, $I^*_{i,t-1}$, as a beginning inventory for period t. Determine optimal values of the variables $X^*_{i,t}$, $I^*_{i,t}$ and $S^*_{i,t}$ for the period t and the objective function value, z^*_t. Keep the objective function value, z^*_t, and optimal values of the variables $X^*_{i,t}$, $I^*_{i,t}$ and $S^*_{i,t}$ to construct the heuristic solution. Set $t = t + 1$. If $t \leq T$, go to Step 1.1; else, stop. Calculate the objective function value for the heuristic algorithm by summing up the objective function values, $z_H = z^*_1 + z^*_2 + ... + z^*_T$.

The algorithm starts with the solution to the problem (RP) in Step 1. Since we relax integer variables and keep the binary decision variables, the problem (RP) provide the solution which is close to the optimal solution to the problem (P). Therefore, we

do not want to be far away from the solution to (RP). The relaxed variables are rounded down in order to obtain possible large integer values on the variables. The rounded down values are used as parameters to construct additional bounds on the decision variables of the problem (P). Specifically, we use rounded-down disassembly quantity in the solution to the problem (RP), $X'_{i,t}$, as a lower bound on the quantity of items to be disassembled for the problem (P) in Step 2. It means that the disassembly quantity can be more than the value of $X'_{i,t}$. Larger quantity of items to be disassembled creates more items. Therefore, quantity of items to be sold can be increased to the possible maximum sales quantity. We cannot allow to change the case where the problem (RP) gives the quantity of disassembled item is zero. These disassembly quantities are equal to zero in the solution to the problem (P). We use rounded-down sales quantity in the solution to the problem (RP), $S'_{i,t}$, as an upper bound on the quantity of items to be sold for the problem (P) in Step 2. Rounded-down sales quantity is the possible maximum quantity of an item for that period to be sold. These additional restrictions force the problem (P) for the solution which is close to rounded-down values. Thus, the heuristic solution can be close to the optimal solution.

3.5. A Numerical Example

In this section we present an example problem to illustrate the solution algorithm. Figure 3.1 shows the disassembly structure for a four-period example problem. Root items to be disassembled are numbered 1 and 2. Items 3-6 are the leaf items to be sold or stocked. Item 4 is a common item that is the child of items 1 and 2.

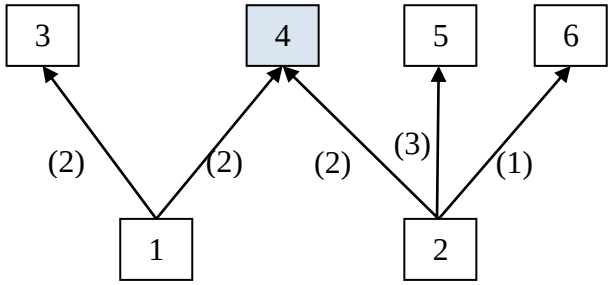

Figure 3.1. The disassembly structure for the example problem

The demand quantities for the leaf items; and cost and revenue parameters are given in Tables 3.1 and 3.2, respectively.

Table 3.1. The demand quantities for the leaf items

	Periods			
Items	1	2	3	4
3	102	0	70	59
4	54	200	0	126
5	69	148	186	0
6	72	58	65	0

Table 3.2. Cost and revenue parameters for the items

	Items					
	1	2	3	4	5	6
Unit Purchasing Cost	131	120	-	-	-	-
Unit Inventory Holding Cost	-	-	10	9	8	10
Unit Disassembly Cost	81	70	-	-	-	-
Fixed Set-up Cost	5000	6000	-	-	-	-
Unit Sales Price	-	-	72	75	50	54

We first calculate cumulative demand for the leaf items by using Equation 3.1. We calculate the required parameter, M_{it}, by using Equation 3.2 and Equation 3.3. M_{it} values for the disassembled items are shown in Table 3.3.

Table 3.3. Mit values for the disassembled items

	Periods			
Items	1	2	3	4
1	190	163	65	63
2	195	163	65	63

First, we solve the problem (P) optimally using Cplex 10.1. The optimal solution to the problem is given in Table 3.4. Total profit of the optimal solution, z^*, is 9876. 81.2% of total demand is satisfied in the optimal solution.

Table 3.4. Optimal solution to the example problem

	Periods				
	Items	1	2	3	4
Number of items purchased and disassembled	1	79	-	-	-
	2	-	111	-	-
Number of items kept in stock	3	56	56	-	-
	4	104	126	126	-
	5	-	185	-	-
	6	-	53	-	-
Number of items sold	3	102	-	56	-
	4	54	200	-	126
	5	-	148	185	-
	6	-	58	53	-

Then, we apply our heuristic solution algorithm for the example problem as follows.

Step 0. Set t=1.

Step 1.1. Solve the problem RP from period 1 to the end period T=4. (Solution for 4-periods).

Step 1.3. The relaxed solution of disassembly quantity is $X_{1,1}$=78.67, the quantities of items kept in stock are $I_{3,1}$=55.33, $I_{4,1}$=103.33 and sales quantities are $S_{3,1}$=102.0, $S_{4,1}$=54.0. Rounded-down disassembly quantity is $X'_{1,1}$=78; quantities of items kept in stock are $I'_{3,1}$=55, $I'_{4,1}$=103; and sales quantities are $S'_{3,1}$=102, $S'_{4,1}$=54. The remaining variables for period t=1 are equal to zero.

Step 2. Additional constraints are $X_{1,1} \geq 78$, $X_{2,1}=0$, $S_{3,1} \leq 102$, $S_{4,1} \leq 54$, $S_{5,1} \leq 0$, and $S_{6,1} \leq 0$. We add these constraints and then solve the model (P) for period $t=1$ and determine the optimal solution to one-period (P). Optimal quantities are $X^*_{1,1}=78$, $X^*_{2,1}=0$, $I^*_{3,1}=54$, $I^*_{4,1}=102$, $S^*_{3,1}=102$, $S^*_{4,1}=54$. The remaining variables are equal to zero. The total profit for period $t=1$ is $z^*_1 = -9968$.

Set $t=2$.

Step 1.1. We add the following constraints to the problem (RP).

$I^*_{3,1}=54$, $I^*_{4,1}=102$, $I^*_{5,1}=0$, and $I^*_{6,1}=0$.

Step 1.3. We solve the (RP) from period 2 to the end period *T*=4. Rounded-down disassembly quantities are $X'_{1,2}=0$, $X'_{2,2}=111$; the quantities of items kept in stock $I'_{3,2}=54$, $I'_{4,2}=124$, $I'_{5,2}=186$, $I'_{6,2}=53$; and the sales quantities are $S'_{3,2}=0$, $S'_{4,2}=200$, $S'_{5,2}=148$, $S'_{6,2}=58$.

Step 2. Additional constraints are $X_{1,2}=0$, $X_{2,2} \geq 111$, $S_{3,2} \leq 0$, $S_{4,2} \leq 200$, $S_{5,2} \leq 148$, and $S_{6,2} \leq 58$. We also add the constraint $I^*_{i,1}$ and solve (P) for period $t=2$. Optimal quantities are $X^*_{1,2}=0$, $X^*_{2,2}=111$, $I^*_{3,2}=54$, $I^*_{4,2}=124$, $I^*_{5,2}=185$, $I^*_{6,2}=53$, $S^*_{3,2}=0$, $S^*_{4,2}=200$, $S^*_{5,2}=148$, and $S^*_{6,2}=58$. The total profit for the second period is $z^*_2=-5224$.

Set $t=3$.

Step 1.1. Additional constraints are: $I^*_{3,2}=54$, $I^*_{4,2}=124$, $I^*_{5,2}=185$ and $I^*_{6,2}=53$.

Step 1.3. Rounded-down quantities of the solution to (RP) are $X'_{1,3}=0$, $X'_{2,3}=0$, $I'_{3,3}=0$, $I'_{4,3}=124$, $I'_{5,3}=0$, $I'_{6,3}=0$, $S'_{3,3}=54$, $S'_{4,3}=0$, $S'_{5,3}=185$, and $S'_{6,3}=53$.

Step 2. Additional constraints are $X_{1,3}=0,\ X_{2,3}=0,\ S_{3,3}\le 54,\ S_{4,3}\le 0,\ S_{5,3}\le 185$, and $S_{6,3}\le 53$. We also add the constraints $I^*_{i,2}$ and and solve (P) for period $t=3$. Optimal quantities are $X^*_{1,3}=0,\ X^*_{2,3}=0,\ I^*_{3,3}=0,\ I^*_{4,3}=124,\ I^*_{5,3}=0,\ I^*_{6,3}=0,\ S^*_{3,3}=54,\ S^*_{4,3}=0,\ S^*_{5,3}=185$, and $S^*_{6,3}=53$. The total profit for the third period is $z^*_3=15748$.

Set $t=4$.

Step 1.1. $I^*_{3,3}=0,\ I^*_{4,3}=124,\ I^*_{5,3}=0$, and $I^*_{6,3}=0$ are the additional constraints for the problem (RP).

Step 1.3. We solve the RP for the last period $t=T=4$. Rounded-down quantities are $X'_{1,4}=0,\ X'_{2,4}=0,\ I'_{3,4}=0,\ I'_{4,4}=0,\ I'_{5,4}=0,\ I'_{6,4}=0,\ S'_{3,4}=0,\ S'_{4,4}=124,\ S'_{5,4}=0$, and $S'_{6,4}=0$.

Step 2. Additional constraints are $X_{1,4}=0,\ X_{2,4}=0,\ S_{3,4}\le 0,\ S_{4,4}\le 124,\ S_{5,4}\le 0,\ S_{6,4}\le 0$. We also add the constraints $I^*_{i,3}$ and and solve (P) for period $t=4$. Optimal quantities are $X^*_{1,4}=0,\ X^*_{2,4}=0, I^*_{3,4}=0,\ I^*_{4,4}=0,\ I^*_{5,4}=0,\ I^*_{6,4}=0,\ S^*_{3,4}=0,\ S^*_{4,4}=124,\ S^*_{5,4}=0,\ S^*_{6,4}=0$. The total profit for the fourth period is $z^*_4=9300$.

The heuristic solution to the example problem is given in Table 3.5.

We sum up the objective function values of four integer programming models solved in Step 2 in order to obtain total profit for our heuristic solution.

z_H = -9968-5224+15748+9300

$z_H=9856$ is the total profit for the heuristic solution. The heuristic solution deviates 0.2 % from the optimal solution and satisfies 80.9% of the total demand.

Table 3.5. Heuristic solution to the example problem

	Periods				
	Items	1	2	3	4
Number of items purchased and disassembled	1	78	-	-	-
	2	-	111	-	-
Number of items kept in stock	3	54	54	-	-
	4	102	124	124	-
	5	-	185	-	-
	6	-	53	-	-
Number of items sold	3	102	-	54	-
	4	54	200	-	124
	5	-	148	185	-
	6	-	58	53	-

CHAPTER 4

COMPUTATIONAL STUDY

We conduct a computational study to measure the performance of the heuristic solution algorithm. The main performance measures are Central Processing Unit (CPU) in seconds, percent deviation of the heuristic solution from the optimal solution or the best integer solution found within the time limits set and service level. Computational tests are performed on randomly generated problems. Integer programming model (P) and the heuristic algorithm are solved by Cplex 10.1 and test is done in personal computer with Inter(R) Core(TM)2 Duo CPU E8400 @2.99 GHz, 3.49 GB of RAM.

We compute the percent deviation of the heuristic solution from the optimal solution, PD, as follows:

$$\text{PD}=\left(\frac{z^{*}-z_{H}}{z_{H}}\right)\times 100\,.$$

Service level, α, is defined as the percent of the demand fulfilled under a solution. It is calculated under for both the heuristic and the optimal solution. The total sales quantity of the items for the entire planning horizon is divided by the total demand quantity for the items for the planning horizon.

$$\alpha=\frac{\sum_{i=i_r+1}^{N}\sum_{t=1}^{T}S_{it}}{\sum_{i=i_r+1}^{N}\sum_{t=1}^{T}d_{it}}$$

We adopt the computational setting used by Kim et. al. (2006a) to generate the problem instances studied. Three levels for number of items are considered: 10, 30, and 50. For each of these three cases, we generate five different product structures. For each product structure 10 different problem instances that differ in cost and revenue parameters are generated. The number of child items for each parent and the corresponding yield quantities are generated from DU (2, 5) and DU (1, 3), respectively. (DU(a, b) is the discrete uniform distribution in intervel (a, b)). The number of root items is generated from DU (1, 2), DU (1, 4) and DU (1, 6) for the problems with 10, 30 and 50 items, respectively. The number of common items is generated from DU (1, 3), DU (1, 6) and DU (1, 9) for the problems with 10, 30 and 50 items, respectively.

Unit disassembly operation costs are generated from DU (50, 100) and inventory holding costs are generated from DU (5, 10). Set-up cost is generated by multiplying the three parameters as follows:

s= the factor used to adjust the magnitude of the set-up cost

$\bar{p}$ = the average disassembly operation cost= $\bar{p} = \dfrac{\sum_{i=1}^{i_l - 1} p_i}{i_l - 1}$.

v= is a parameter used to incorporate the randomness in the set-up cost.

We set s to 1, 5 and 10 for the cases of low, medium and high set-up costs, respectively, and v is generated from U (5, 15), i.e. the uniform distribution between 5 and 15.

The fixed disassembly cost, f_i for item $i = 1, 2, ..., i_l - 1$ is generated as follows:

$$f_i = s.\ \bar{p}\ .v \qquad \text{for } i = 1, 2, ..., i_l\text{-}1.$$

Three different length of planning horizon values are considered: 10, 20 and 30 periods. We first generate the problems for T=30. We take the demand of the first 10

and the first 20 periods of the 30-period problems and to construct the problem instances for *T*=10, *T*=20, respectively. Demands for the intermediate and leaf items are set to 0 or DU (50, 200) with probabilities 0.1 and 0.9, respectively. Initial inventory is set to 0 in the test problems.

Our model includes cost and revenue parameter which are not included in the study by Kim et. al. (2006a). Purchasing cost of root items are generated from DU (100, 150). Unit sales price of non-root items are generated using the variable cost of disassembly and the yield information. First, we calculate the unit cost added to the items using Equation (4.1) and (4.2). Low sales price is generated by multiplying unit cost with a U(1.2, 1,5) random variable whereas high sales price is generated by multiplying unit cost with a U(1.7, 2.0) random variable.

Let u_i is a unit variable cost added to obtain child item i by its parent item (root item), and u_e is a unit variable cost added to obtain child item *e* by its parent item (non-root parent item).

We calculate u_i by considering purchasing cost of the associated root items and disassembly cost of the root item. It is given in Equation 4.1.

$$u_i = \frac{a_j + p_j}{\sum_{k \in H(j)} \lambda_{jk}} \quad \text{for } i \in H(j) \tag{4.1}$$

Unit cost added to child items of the non-root parent items is calculated by using Equation 4.2.

$$u_e = \frac{u_i + p_i}{\sum_{g \in H(i)} \lambda_{ig}} \quad \text{for } e \in H(i) \tag{4.2}$$

where u_i is the unit variable cost added to parent item of child item *e*. Here, we calculate unit costs added to the items level-by-level. We start from the calculation of unit cost added to the child of the root items and at the end, we reach last leaf item.

Unit sales price for the child items of root and non-root items for low price case are generated as follows:

$$\text{Low sales price}: r_i = u_i \times U(1.2,\ 1.5)$$
$$\text{Low sales price}: r_e = u_e \times U(1.2,\ 1.5)$$

where r_i is the sales price for the child items of the root items

r_e is the sales price for the child items of the non-root items

Unit sales price for the child of the non-root items for high price case are calculated as follows:

$$\text{High sales price}: r_i = u_i \times U(1.7,\ 2.0)$$
$$\text{High sales price}: r_e = u_e \times U(1.7,\ 2.0)$$

Note that the unit sales price generation above is valid for the non-common items. Since a non-common item has one parent item, corresponding cost parameters of its parent item are used to generate the sales price of the non-common item. For common items, a parent item is randomly selected and the unit sales price of the common item is generated by using the cost parameters of the selected parent item.

We generate three levels of number of items (10, 30, and 50), five different product structures for each level of number of items and each five cases include ten problem instances. We also consider the three different length of planning horizon values (10, 20, and 30), two levels of sales price (low, high), and three levels of set-up cost (low, mid, high). In total, 2700 problem instances are generated and solved.

We use CPLEX 10.1 to solve the integer programming model (P) optimally. We limit the CPU time with 3-hours. For the problems that cannot be solved within 3 hours optimally, the objective function value of the best integer solution found within an hour and 3 hours are reported. The heuristic solutions are compared with both of these to assess the benefits of increasing the CPU time limit.

For some problem instances, the heuristic algorithm requires high computational times. The most demanding part of the developed heuristic in terms of computational effort is the solution of (RP) which is a mixed integer programming model. In the case that the heuristic solution cannot be obtained within an hour, we divide the planning horizon into small pieces, and create subproblems. Then the heuristic algorithm is applied to the subproblems sequentially. The RP problem with shorter planning horizon requires less computational time than the RP problem with longer planning horizon.

Especially, problem instances with complex product structures and/or long planning horizons require high computational time even if the proposed heuristic solution is employed for the solution. Different schemes for dividing the entire planning horizon to smaller ones are considered. First, we divide 30-period problems into two 15-period problems. This scheme reduces the computational time. However, for some problem instances, this division does not yield a total run time which is less than an hour. Then, we divide 30-period problems into six 5-period problems. This scheme requires very low computational time. However, the heuristic solution deviates from the optimal or best integer solution by 15-20%. Finally, we decide to divide *T*-period RP problem into 10-period problems. For instance, we obtain three 10-period problems for a problem instance with a planning horizon of 30. This scheme yields a total run time which is less than an hour and the heuristic solution deviates from the optimal or best integer solution between 0.1-5%.

We test the behavior of the objective function with the changing time limit. Specifically, we test the change in the objective function value of the best integer solution within an hour. We record the objective function value of the best integer solution in every 5 minutes for the problem instances cannot be solved optimally within 3 hours. We classify the observations on the behavior of the objective function values recorded to as follows.

- There is no change in the objective function value with the increasing time limit. The objective function value which is obtained in the first 5 minutes does not change through 60 minutes.
- The first 15 minute of run time are enough to reach the best solution obtained in 60 minutes. In other words, solution obtained at 15 minutes is very close (less than 1 %) to the solution at 60 minutes.
- The objective function value increases as the run time from 5 minutes to 60 minutes.

When we analyze the results tested, we conclude that the first two cases dominate the last one. 37%, 45%, and, 18% of the problem instances behave as case one, case two and case three, respectively. These cases are affects of the number of items and the planning time horizon. Especially, the problem instances with the less number of items and the short planning horizon time behave as the first case. The problem instances with longer planning behave as the second and third case. We show these cases on three problem instances with 10, 30 and 50 items for 30 periods of planning horizon. Figure 4.1, 4.2 and 4.4 shows the behavior of total profit with changing time limit for each three cases. Horizontal axis shows the time limit from 5 minutes to 60 minutes and vertical axis shows the objective function value (total profit). We also report the percent deviations of the solutions for a certain time limit from the best solution obtained at 1 hour in Figure 4.3 and 4.5.

The first case of the behavior of the objective function can be seen in Figure 4.1. It shows that increasing time limit yields no improvement on the total profit. In this case, increasing time limit to an hour is unnecessary. For some problem instances, objective function value can be improved with increasing time limit by ignorable amount. We include these problem instances in first case of the behavior of the objective function.

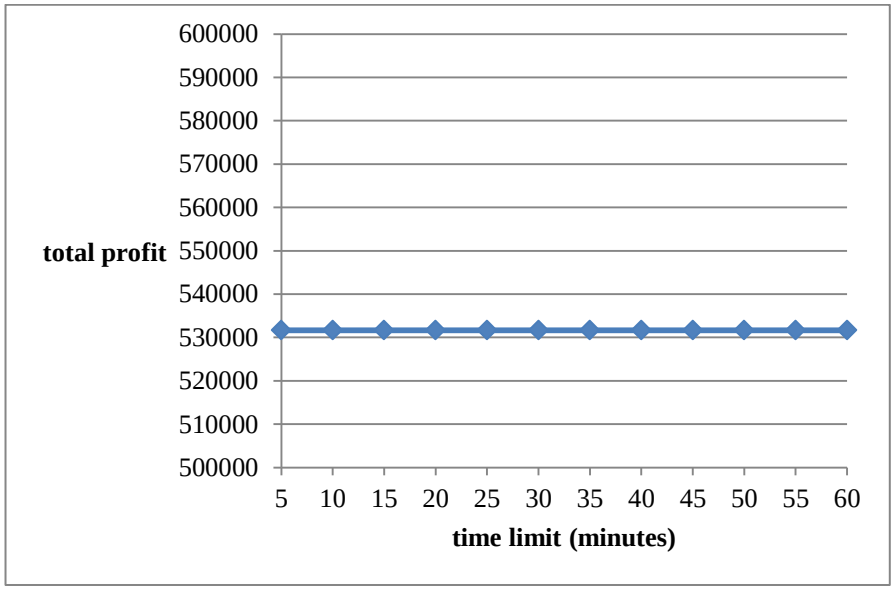

Figure 4.1. Total profit vs. time limit for N=10, T=30 with high sales price level and low set-up cost

The second case for the behavior of the objective function is given in Figure 4.2. The objective function value increases as the time limit increases within the first 15 minutes and it is stable after 30 minutes. As it can be seen in Figure 4.3, total profit at 15 minutes deviates nearly 0.1% from the total profit at 1 hour. At the 30 minutes, it reaches to the solution obtained at 1 hour. Since the objective function yields significant improvement within first 15 minutes and ignorable improvement after 15 minutes, 15 minutes time limit is sufficient for this situation.

The third case for the behavior of the objective function is given in Figure 4.4. As the time limit increases, the total profit increases. As it can be seen in Figure 4.5, the solution obtained within first 10 minutes deviates nearly 2.5-3% from the solution obtained at 1 hour. The deviation is stable on the 1.5% between 20-35 minutes. The solution at 40 minutes deviates more than 1% from the solution at 1 hour. At 45 minutes, the solution is close to the solution in 60 minutes.

If the objective function still yields significant improvement from 15 minutes to 60 minutes, we include this situation in case three for the behavior of the objective function.

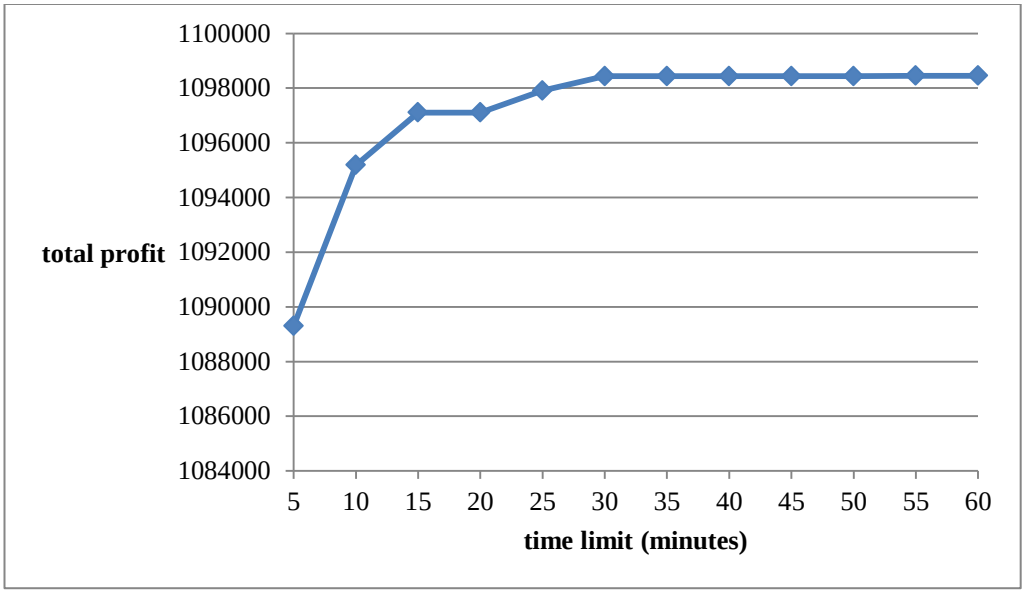

Figure 4.2. Total profit vs. time limit for N=30, T=30 with high sales price level and high set-up cost

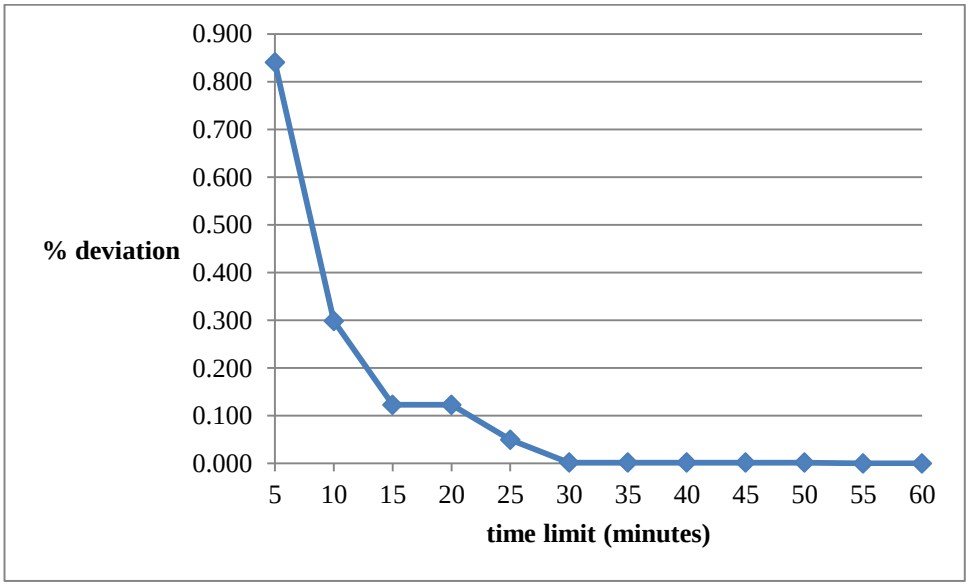

Figure 4.3. Percent deviation of the solutions from the best feasible solution at 1 hour for N=30, T=30 with high sales price level and high set-up cost

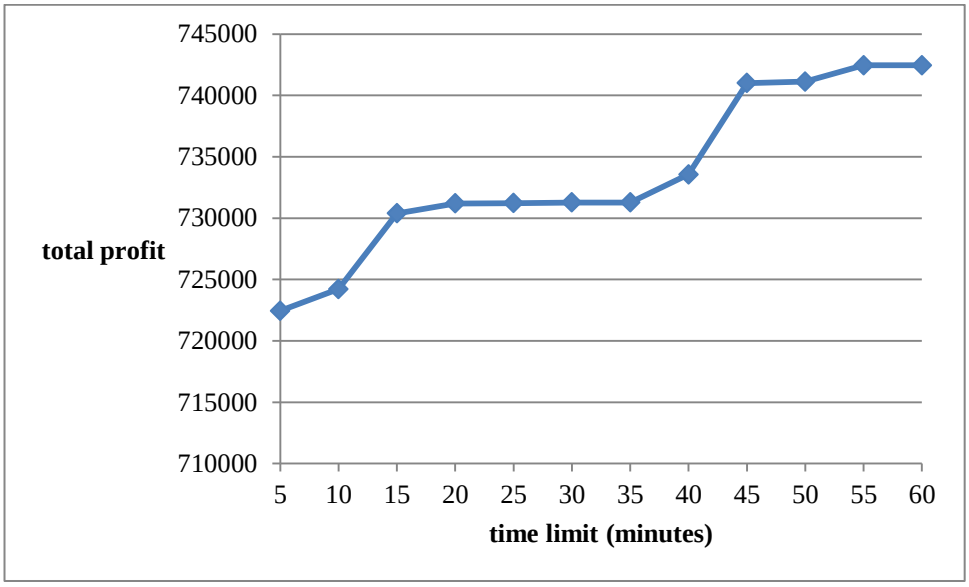

Figure 4.4. Total profit vs. time limit for N=50, T=30 with low sales price level and medium set-up cost

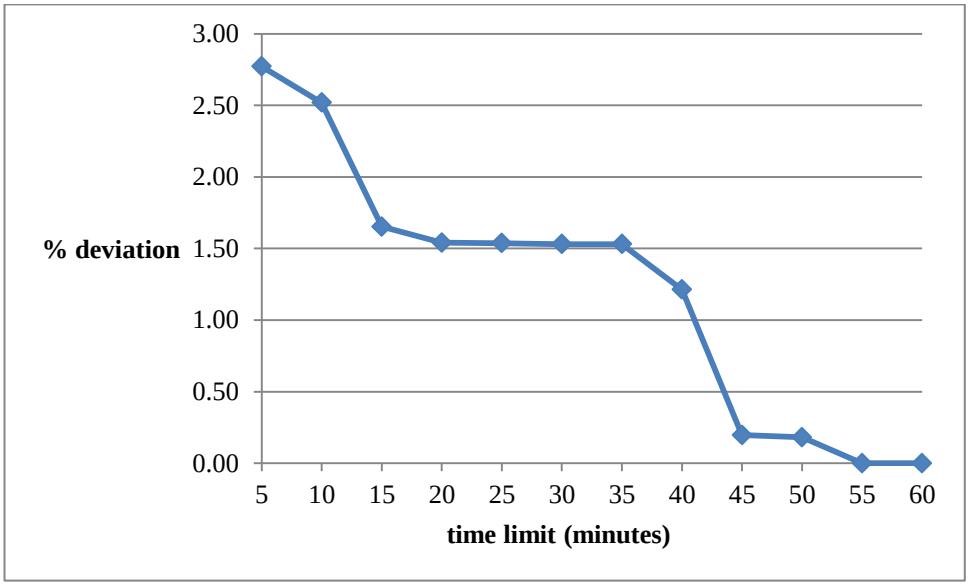

Figure 4.5. Percent deviation of the solutions from the best feasible solution at 1 hour for N=50, T=30 with low sales price level and medium set-up cost

4.1. Discussion of the Results

CPU times in seconds, percent deviation from the optimal solution and service level are the performance measures considered. We report the maximum, the minimum and the average value of these performance measures. In addition to these measures, we also report the number of problem instances solved optimally within 1-hour and 3-hour time limit. Number of problem instances solved optimally for three levels of number of items 10, 20 and 30; three different set-up cost magnitude; and two levels of sales price are given in Table 4.1. The total number of problem instances solved optimally is 1155. 1100 of them are solved within an hour; and 55 problem instances are solved optimally within the additional 2 hours. Additional two hours do not provide a significant increase on the number of problems solved optimally. Empty cells in Table 4.1 shows no problem instances solved optimally.

Table 4.1. Number of problem instances solved optimally

Sales price		**low**						**high**					
Set-up cost		**low**		**mid**		**high**		**low**		**mid**		**high**	
CPU time limit (hr)		1	3	1	3	1	3	1	3	1	3	1	3
N=10	**T=10**	50	-	50	-	50	-	50	-	50	-	50	-
	T=20	45	2	50	-	45	4	41	1	47	2	35	7
	T=30	28	2	27	2	24	1	27	2	22	2	21	-
N=30	**T=10**	43	3	48	2	47	2	43	1	49	1	40	4
	T=20	-	-	6	2	13	4	-	-	-	-	-	-
	T=30	-	-	-	-	-	-	-	-	-	-	-	-
N=50	**T=10**	19	2	22	1	20	1	9	2	16	2	13	3
	T=20	-	-	-	-	-	-	-	-	-	-	-	-
	T=30	-	-	-	-	-	-	-	-	-	-	-	-
Total		185	9	203	7	199	12	170	6	184	7	159	14

The effects of the parameters on the number of problem instances solved optimally are as follows.

- As the number of item increases, the number of problem instances that are solved optimally within the given time limit decreases. Specifically, 737, 308 and 110 problem instances for 10, 30 and 50 item problems, respectively.

- We cannot say that as the set-up cost increases, the number of problem instances solved optimally decreases. The number of the problem instances is solved optimally with three set-up cost levels are as follows.

- In the case of medium set-up cost, more problem instances are solved optimally when we compare low and high set-up cost levels, in the overall. Specifically, 370, 401, and 384 problem instances for low, medium and high set-up cost levels, respectively. For the case of low set-up cost, the solver (Cplex) may not prove optimality of the solutions obtained. Therefore, less number of problem instances is solved optimally for low set-up cost level compared to the medium and high set-up costs levels.

- As the length of the planning horizon increases, the number of problem instances solved optimally decreases. It can be seen in Table 4.1, none of the problem instances with 50 items when T=30 and with 50 items when T=20 are solved optimally. 693, 304 and 158 problem instances for a palnning horizon of length 10, 20 and 30, respectively.

- As the sale price level increases, the number of problem instances solved optimally decreases. In the overall, 615 and 540 of the problem instances solved optimally in the cases of low and high sales prices, respectively.

- As it can be observed in Table 4.1, the number of problem instances that could not be solved optimally within the time limit increases as

    - The number of items increases
    - The number of periods increases
    - The sales price increases.

We report the percent improvement on the total profit when the time limit is extended to 3 hours from 1 hour in Table 4.2. All problem instances with 10 items and 10 periods are solved optimally. Therefore, we do not report these cases in Table 4.2. The percent improvement on the total profit decreases as

- The number of items increases
- The number of periods increases

The maximum improvement is 2.02 %. The best feasible solution can be improved less than 1% on the average. Especially, for some problem instances with 50 items, the best feasible solution cannot be improved within an additional 2 hours time limit. Therefore, we conclude that the effect of extending the time limit from 1 hour to 3 hours is not significant.

Table 4.2. Percent improvement on the total profit of the best feasible solution obtained extending the time limit from 1 hour to 3 hours

Number of items	10		30			50		
	T=20	T=30	T=10	T=20	T=30	T=10	T=20	T=30
min	0.03	0.00	0.01	0.02	0.00	0.00	0.00	0.00
avg	0.65	0.53	0.68	0.58	0.50	0.35	0.29	0.26
max	0.94	0.94	2.02	0.91	0.51	0.96	0.82	0.49

We report CPU time in seconds for the optimal solutions in Table 4.3. There are some *missing rows* and *empty cells* in Table 4.3 since no problem can be solved optimally in these cases.

The effects of the parameters on the CPU time of the optimal solution are as follows.

- As the number of items increases, the CPU time required to get the optimal solution increases.

- As the length of the planning horizon increases, the CPU time to get the optimal solution increases. We expect that the CPU times increase exponentially as the number of periods increase. It can be seen in Table 4.3, as the length of the planning horizon increases from 10 to 20 periods for 10-item problems, the average CPU time increases exponentially. For the other cases, the increase is not as significant as this case.

- As the set-up cost increases, the CPU time to get the optimal solution increases. There is no so much difference in the CPU time of low, medium and high set-up costs, on the average. Since we cannot solve same number of problem instances for each case, average CPU time can be close to each other. As the sales price increases, the CPU time to get the optimal solution increases.

Table 4.3. CPU times in seconds of optimally solved problem instances

	Sales price		low			high		
	Set-up cost		low	mid	high	low	mid	high
N=10	T=10	min	0.01	0.04	0.04	0.01	0.05	0.06
		avg	3.62	4.28	4.35	3.65	4.54	4.91
		max	3.84	4.54	4.59	6.23	7.73	8.11
	T=20	min	0.05	0.06	0.11	0.06	0.14	0.30
		avg	1441.44	1611.54	1898.02	1580.97	1803.22	1951.90
		max	3888.91	3149.60	3871.35	4377.40	4739.97	4967.94
	T=30	min	0.08	0.11	0.17	0.11	0.23	0.36
		avg	2266.94	2280.06	2370.74	2356.53	2464.76	2991.23
		max	4308.82	5609.70	4020.14	4461.49	4269.87	3241.93
N=30	T=10	min	0.69	0.72	0.41	1.34	1.28	6.67
		avg	2032.52	2232.40	2685.29	2147.90	2250.08	2778.28
		max	3799.91	3899.05	4631.45	3781.29	3947.37	4431.48
	T=20	min	-	209.04	238.75	-	-	-
		avg	-	3633.91	3759.36	-	-	-
		max	-	4541.06	4742.26	-	-	-
N=50	T=10	min	21.20	30.84	32.56	36.34	37.92	39.01
		avg	2834.48	2961.71	3053.37	3168.19	3337.29	4161.88
		max	4314.72	4674.39	4865.91	6283.69	6302.46	8807.33

The generated problem instances solved optimally are also solved using the proposed heuristic approach. We report the CPU time in seconds for the heuristic solutions in Table 4.4. Since some problem instances cannot be solved within an hour by the heuristic approach and they are not reported in Table 4.4, increase in CPU time can be less than we expect.

Table 4.4. CPU times in seconds of heuristic solutions for optimally solved problem instances

	Sales price		low			high		
	Set-up cost		low	mid	high	low	mid	high
N=10	T=10	min	0.10	0.17	0.15	0.12	0.14	0.14
		avg	0.25	0.43	0.53	0.26	0.44	0.71
		max	0.67	1.15	1.45	0.60	1.90	3.65
	T=20	min	0.31	0.13	0.70	0.42	0.69	0.19
		avg	0.94	16.69	26.95	2.82	28.87	38.08
		max	4.12	138.01	131.98	6.36	338.13	215.26
	T=30	min	0.81	1.26	1.51	0.72	2.12	2.66
		avg	54.74	66.92	86.80	57.80	75.32	93.12
		max	210.00	226.70	168.54	69.20	385.19	348.02
N=30	T=10	min	0.20	0.67	0.53	0.30	1.20	1.53
		avg	0.73	3.13	4.43	0.81	9.23	11.66
		max	1.26	7.25	8.50	6.89	81.14	51.08
	T=20	min	-	4.78	3.74	-	-	-
		avg	-	22.12	40.31	-	-	-
		max	-	226.30	243.42	-	-	-
N=50	T=10	min	0.36	0.72	1.22	0.52	1.91	2.69
		avg	1.80	9.41	16.26	2.04	102.89	107.69
		max	3.64	20.45	18.00	12.64	138.94	137.23

The effects of the parameters on the CPU time of the heuristic solution are as follows.

In general, as the number of item increases, the CPU time of the heuristic solution increases. However, the maximum CPU time for the problem instances with 50 items for the ones with 10 and 30 items. Since some problem instances with 50 items cannot be solved within an hour. For those problems, we decompose the planning horizon to solve them within an hour. The heuristic with decomposition is explained in this chapter.

As the lenght of the planning horizon increases, the CPU time of the heuristic solution increases. As in the optimal solution, the average CPU times of heuristic solution increases exponentially.

As the set-up cost increases, the CPU time of the heuristic solution increases.

As the sales price increases, the CPU time of the heuristic solution increases.

Based on Table 4.3 and Table 4.4, we compare the heuristic solution with the optimal ones.

The heuristic approach solves the problems within less computational time than the optimal ones.

- For 10-period problems with 10 items, CPU times for the optimal and the heuristic solution are close on the average. Since they are short planning period problem instances, the optimal solution can be easily obtained.

- When the length of the planning horizon increases to 20 and 30, the heuristic solutions are obtained within less computational time than the optimal solutions for all levels of set-up cost and sales price.

- For the problem instances with 30 and 50 items, the heuristic approach also solves the problems within low computational time. While the optimal solutions are obtained within high computational time. As a result, we benefit of using the heuristic approach in terms of CPU time.

When we solve (RP) problem, we obtain upper bound on the problem (P). We report the percent deviation of the upper bound from the optimal solution. As it can be observed in Table 4.5, the percent deviation of the upper bound from the optimal solution decreases as

- Sales price increases
- Set-up cost increases
- Number of item decreases
- The length of the planning horizon increases

- Maximum deviation of the upper bound from the optimal solution is 0.697%.
- Average deviations of upper bound from the optimal solution are less than 0.5%.
- Overall average deviation of upper bound from the optimal solution is 0.09%.
- As a result, (RP) problem gives sharp upper bounds on the optimal solutions. Thus it affects the quality of the heuristic solution to obtain near optimal solution.

Table 4.5. Percent deviation of upper bound from the optimal solution

Sales price			low			high		
Set-up cost			low	mid	high	low	mid	high
N=10	T=10	min	0.002	0.000	0.000	0.000	0.000	0.000
		avg	0.110	0.093	0.089	0.078	0.065	0.052
		max	0.697	0.649	0.564	0.395	0.373	0.336
	T=20	min	0.003	0.001	0.000	0.001	0.000	0.000
		avg	0.092	0.090	0.082	0.068	0.064	0.050
		max	0.479	0.359	0.279	0.241	0.238	0.182
	T=30	min	0.004	0.000	0.000	0.001	0.000	0.000
		avg	0.091	0.088	0.080	0.060	0.045	0.044
		max	0.219	0.208	0.191	0.195	0.178	0.144
N=30	T=10	min	0.037	0.024	0.017	0.028	0.022	0.010
		avg	0.113	0.094	0.090	0.081	0.068	0.053
		max	0.199	0.187	0.178	0.185	0.144	0.156
	T=20	min	-	0.068	0.062	-	-	-
		avg	-	0.091	0.085	-	-	-
		max	-	0.250	0.192	-	-	-
	T=30	min	-	-	-	-	-	-
		avg	-	-	-	-	-	-
		max	-	-	-	-	-	-
N=50	T=10	min	0.119	0.090	0.017	0.101	0.053	0.012
		avg	0.204	0.153	0.132	0.154	0.126	0.105
		max	0.380	0.347	0.324	0.189	0.177	0.155
	T=20	min	-	-	-	-	-	-
		avg	-	-	-	-	-	-
		max	-	-	-	-	-	-
	T=30	min	-	-	-	-	-	-
		avg	-	-	-	-	-	-
		max	-	-	-	-	-	-

Percent deviations of the heuristic solution from the optimal solution are summarized in Table 4.6. It is reported for all level of number of items, number of periods, sales price and set-up costs.

Table 4.6. Percent deviations of heuristic solutions from the optimal solutions

	Sales price		low			high		
	Set-up cost		low	mid	high	low	mid	high
N=10	T=10	min	0.00	0.00	0.00	0.01	0.00	0.00
		avg	0.33	0.28	0.20	0.24	0.20	0.18
		max	0.72	0.61	0.43	0.45	0.47	0.43
	T=20	min	0.01	0.02	0.00	0.00	0.01	0.02
		avg	0.29	0.24	0.19	0.22	0.18	0.17
		max	0.60	0.50	0.42	0.48	0.47	0.40
	T=30	min	0.01	0.00	0.00	0.01	0.00	0.01
		avg	0.28	0.20	0.17	0.20	0.17	0.16
		max	0.52	0.44	0.40	0.47	0.41	0.40
N=30	T=10	min	0.11	0.06	0.00	0.13	0.02	0.02
		avg	0.35	0.31	0.30	0.30	0.24	0.20
		max	0.74	0.61	0.48	0.50	0.51	0.50
	T=20	min	-	0.10	0.08	-	-	-
		avg	-	0.29	0.25	-	-	-
		max	-	0.50	0.50	-	-	-
N=50	T=10	min	0.24	0.13	0.08	0.22	0.12	0.09
		avg	0.41	0.39	0.32	0.35	0.30	0.28
		max	0.87	0.84	0.97	0.86	0.52	0.60

As it can be observed that in Table 4.6, the average percent deviation of heuristic solution from the optimal solution decreases as

- Sales price increases
- Set-up cost increases
- Number of items decreases
- The length of the planning horizon increases

The heuristic approach gives near optimal results. The heuristic solution deviates less than 0.5% on the average. The maximum deviations is less than 1% for all cases.

Percent deviations of the heuristic solution from the best feasible solution within 3-hour time limit are given in Table 4.7 for the instances that cannot be solved in 3 hours. The heuristic solution is close to the feasible solution obtained within 3 hours. The percent deviation is less than 1% on the average. The heuristic solution can

reach to the best feasible solution which is obtained within 3 hours time limit. As it can be observed that in Table 4.7, the average percent deviation of the heuristic solution from the best feasible solution decreases as

- Sales price increases
- Set-up cost increases
- Number of item decreases
- The length of the planning horizon increases

Table 4.7. Percent deviations of the heuristic solutions from the best feasible solution within 3 hours

Sales price			**low**			**high**		
Set-up cost			**low**	**mid**	**high**	**low**	**mid**	**high**
N=10	**T=20**	min	0.18	-	0.25	0.21	0.26	0.15
		avg	0.29	-	0.26	0.28	0.26	0.20
		max	0.55	-	0.97	0.73	0.26	0.50
	T=30	min	0.18	0.17	0.07	0.11	0.13	0.15
		avg	0.30	0.28	0.26	0.28	0.25	0.24
		max	0.63	0.94	0.88	0.59	0.83	0.84
N=30	**T=10**	min	0.30	-	0.32	0.17	-	0.31
		avg	0.37	-	0.32	0.33	-	0.26
		max	0.64	-	0.32	0.41	-	0.90
	T=20	min	0.09	0.02	0.04	0.16	0.11	0.21
		avg	0.34	0.32	0.27	0.32	0.30	0.24
		max	0.58	1.29	1.30	0.42	1.09	0.97
	T=30	min	0.00	0.02	-	0.03	0.22	0.22
		avg	0.31	0.30	-	0.29	0.27	0.26
		max	0.64	0.92	-	0.51	0.80	0.72
N=50	**T=10**	min	0.20	0.08	0.10	0.11	0.07	0.00
		avg	0.39	0.38	0.33	0.34	0.32	0.28
		max	0.58	0.53	0.96	0.48	0.48	0.73
	T=20	min	0.10	0.09	0.00	0.07	-	-
		avg	0.38	0.36	0.29	0.33	-	-
		max	0.60	0.62	0.90	0.45	-	-
	T=30	min	0.04	0.03	0.05	0.03	-	-
		avg	0.32	0.31	0.26	0.30	-	-
		max	0.45	0.82	0.80	0.43	-	-

As the number of items and periods increase, the heuristic approach requires a computational time more than an hour. The number of these problem instances is given in Table 4.8. In total, 726 of the problem instances out of 2700 cannot be solved within an hour using the heuristic solution approach. For these problem instances, we provide the heuristic with decomposition to solve the problems within less than an hour. The planning horizons of 20 and 30 periods are divided into 2 and 3 equal planning horizons, respectively.

Table 4.8. Number of problem instances cannot be solved optimally within 3-hour

	Sales price	**low**			**high**		
	Set-up cost	**low**	**mid**	**high**	**low**	**mid**	**high**
N=10	**T=30**	-	10	9	-	5	12
N=30	**T=20**	-	40	16	-	10	16
	T=30	-	46	50	-	40	39
N=50	**T=20**	7	40	40	22	50	50
	T=30	10	43	46	25	50	50
Total		**17**	**179**	**161**	**47**	**155**	**167**

The CPU time in seconds for the heuristic with decomposition scheme is given in Table 4.9.

- Most of the problem instances with 50 items are solved by the heuristic approach with decomposition.
- The problems with 10 items are solved within a few seconds for the case of low sales price.
- As the number of items, number of periods, setup cost and sales price increase, the average CPU time of the heuristic approach with decomposition increases.
- The maximum CPU time of the heuristic with decomposition is 2023.89 seconds which is still less than an hour.

We also report the percent deviations of the heuristic solution with decomposition from the best feasible solution obtained within 3 hours. As it can be observed that in Table 4.10., the average percent deviation of the heuristic solution with decomposition from the best feasible solution increases as

- Sales price increases
- Set-up cost increases
- Number of item increases
- The length of the planning horizon increases

Table 4.9. CPU times in seconds for the heuristic solutions with decomposition

	Sales price		**low**			**high**		
	Set-up cost		**low**	**mid**	**high**	**low**	**mid**	**high**
N=10	**T=30**	min	-	2.33	1.44	-	1.52	1.86
		avg	-	2.78	3.14	-	8.66	11.18
		max	-	4.25	6.53	-	87.87	205.91
N=30	**T=20**	min	-	0.93	4.22	-	6.75	4.38
		avg	-	7.88	11.42	-	32.05	36.45
		max	-	20.59	24.56	-	283.14	258.80
	T=30	min	-	4.29	4.56	-	8.17	9.57
		avg	-	12.83	13.23	-	37.69	67.10
		max	-	66.99	99.26	-	193.75	536.59
N=50	**T=20**	min	4.95	4.59	5.23	0.13	3.73	7.06
		avg	14.40	134.93	57.12	18.26	451.72	598.18
		max	67.28	773.70	248.86	59.69	645.43	1107.26
	T=30	min	2.95	2.29	4.78	2.50	5.84	11.64
		avg	19.20	145.19	505.17	18.55	629.54	986.75
		max	77.37	937.64	886.66	163.51	1293.28	2023.89

- Although the maximum deviation is 5%, the heuristic approach with decomposition deviates between 0.57%-1.53% from the best feasible solution on the average.

- Average deviations are reasonable for hard problems. For example, the maximum of the average deviation is 1.53% for problem instances with 50 items for 30 periods, high set-up cost and high sales price case.
- The minimum percent deviations are very low; in some cases of 50 items with 30 periods, the heuristic solution is close to the optimal solutions.

Table 4.10. Percent deviations of the heuristic solution with decomposition from the best feasible solution obtained within 3 hours

	Sales price		**low**			**high**		
	Set-up cost		**low**	**mid**	**high**	**low**	**mid**	**high**
N=10	**T=30**	min	-	0.20	0.21	-	0.22	0.24
		avg	-	1.05	1.13	-	1.08	1.17
		max	-	1.20	1.30	-	1.40	1.80
N=30	**T=20**	min	-	0.03	0.05	-	0.04	0.27
		avg	-	0.97	1.15	-	1.24	1.40
		max	-	1.77	3.08	-	4.00	4.20
	T=30	min	-	0.22	0.23	-	0.27	0.29
		avg	-	1.12	1.17	-	1.26	1.51
		max	-	3.05	3.70	-	4.30	4.51
N=50	**T=20**	min	0.09	0.09	0.10	0.10	0.13	0.02
		avg	0.57	1.12	1.20	0.60	1.26	1.48
		max	1.20	3.68	3.45	1.25	4.04	4.32
	T=30	min	0.11	0.09	0.13	0.18	0.03	0.02
		avg	0.61	1.21	1.25	0.71	1.32	1.53
		max	1.21	4.63	3.45	1.26	5.00	4.71

We report the service levels for the problem instances in Tables 4.11, 4.12 and 4.13. In Table 4.11, service levels under the optimal and the heuristic solutions are given. Service levels of the heuristic solutions are close to the optimal ones. It other words, the percentages of demand fulfilled in the optimal and heuristic solutions are close to each other. Service level of the heuristic solution and the best feasible solution obtained within 3 hours is given in Table 4.12. Table 4.13 shows the service level under the solution of the heuristic approach with decomposition.

The following observations are made based on Table 4.11.

- Service levels are between 64%-91% and 64%-88% for optimal and heuristic solutions, respectively.
- As the number of items increases, the service level increases.
- As the set-up cost increases, the service level decreases.
- As the sales price increases, the service level under the solution increases. The amount of increase is between 0.01 and 0.07 unit in percent.

The following observations are made based on Table 4.12.

- The service level of the heuristic solution is close to the service level of the best feasible solution, and the effects of the parameters on the service level is the same as in the cases that the optimal solutions can be found within 3 hours.
- Service levels are between 71%-91% and 70%-90% for the best feasible solution and heuristic solutions, respectively.

The following observations are made based on Table 4.13.

Since the maximum deviations of the heuristic solutions with decomposition to some problem instances are near to 5% which is reported in Table 4.10, the maximum and minimum service levels of the solutions of the heuristic with decomposition are not close to each other.

We cannot satisfy the demand for the items fully, i.e., service level which all is less than 100%. As a result, service level shows the importance of profit maximization rather than cost minimization.

Table 4.11. Service level of the optimal and heuristic solutions

	Sales price		low						high					
	Set-up cost		low		mid		high		low		mid		high	
	Optimal vs.Heuristic		OPT	HEUR	OPT	HEUR	OPT	HEUR	OPT	HEUR	OPT	HEUR	OPT	HEUR
N=10	T=10	min	0.69	0.69	0.67	0.67	0.65	0.65	0.76	0.75	0.74	0.72	0.73	0.70
		avg	0.72	0.71	0.71	0.70	0.69	0.67	0.78	0.76	0.75	0.74	0.75	0.72
		max	0.74	0.72	0.73	0.71	0.70	0.68	0.79	0.78	0.78	0.76	0.76	0.75
	T=20	min	0.70	0.70	0.68	0.68	0.65	0.65	0.72	0.72	0.72	0.70	0.71	0.70
		avg	0.73	0.72	0.72	0.70	0.66	0.64	0.74	0.73	0.73	0.71	0.72	0.71
		max	0.74	0.73	0.73	0.71	0.68	0.67	0.75	0.75	0.74	0.73	0.74	0.72
	T=30	min	0.72	0.72	0.72	0.72	0.70	0.70	0.76	0.75	0.76	0.75	0.75	0.73
		avg	0.74	0.73	0.73	0.73	0.74	0.72	0.78	0.76	0.77	0.77	0.76	0.75
		max	0.75	0.75	0.74	0.74	0.75	0.74	0.79	0.78	0.79	0.78	0.78	0.78
N=30	T=10	min	0.73	0.73	0.72	0.72	0.70	0.70	0.77	0.76	0.72	0.72	0.73	0.73
		avg	0.74	0.74	0.74	0.74	0.73	0.71	0.78	0.77	0.77	0.74	0.76	0.75
		max	0.76	0.75	0.75	0.75	0.74	0.73	0.80	0.78	0.78	0.75	0.77	0.76
	T=20	min	-	-	0.74	0.72	0.72	0.72	-	-	-	-	-	-
		avg	-	-	0.75	0.74	0.74	0.73	-	-	-	-	-	-
		max	-	-	0.77	0.76	0.75	0.74	-	-	-	-	-	-
N=50	T=10	min	0.86	0.84	0.84	0.82	0.80	0.80	0.88	0.85	0.85	0.85	0.82	0.82
		avg	0.87	0.85	0.85	0.83	0.82	0.80	0.90	0.86	0.87	0.86	0.85	0.84
		max	0.89	0.88	0.87	0.86	0.83	0.81	0.91	0.88	0.88	0.88	0.86	0.85

Table 4.12. Service level of the heuristic solution and the best feasible solution obtained within 3 hours

	Sales price		low						high					
	Set-up cost		low		mid		high		low		mid		high	
	Feas vs. Heur		FEAS	HEUR	FEAS	HEUR	FEAS	HEUR	FEAS	HEUR	FEAS	HEUR	FEAS	HEUR
N=10	T=20	min	0.75	0.75	-	-	0.74	0.73	0.78	0.77	0.78	0.76	0.77	0.76
		avg	0.76	0.76	-	-	0.75	0.74	0.79	0.79	0.79	0.77	0.79	0.78
		max	0.78	0.77	-	-	0.77	0.76	0.81	0.80	0.80	0.79	0.80	0.80
	T=30	min	0.78	0.76	0.77	0.75	0.75	0.75	0.80	0.79	0.79	0.77	0.77	0.76
		avg	0.79	0.78	0.79	0.78	0.78	0.76	0.82	0.82	0.80	0.79	0.79	0.77
		max	0.80	0.79	0.80	0.79	0.80	0.79	0.83	0.83	0.81	0.80	0.80	0.79
N=30	T=10	min	0.75	0.74	-	-	0.71	0.68	0.78	0.76	-	-	0.74	0.71
		avg	0.77	0.76	-	-	0.73	0.72	0.79	0.78	-	-	0.75	0.73
		max	0.79	0.78	-	-	0.74	0.73	0.80	0.79	-	-	0.76	0.74
	T=20	min	0.76	0.76	0.74	0.74	0.71	0.70	0.78	0.75	0.75	0.74	0.75	0.73
		avg	0.79	0.77	0.75	0.75	0.75	0.73	0.79	0.77	0.77	0.77	0.76	0.74
		max	0.80	0.79	0.77	0.76	0.76	0.75	0.81	0.80	0.78	0.78	0.77	0.76
	T=30	min	0.83	0.83	0.82	0.82	0.80	0.79	0.83	0.83	0.86	0.85	-	-
		avg	0.85	0.84	0.83	0.83	0.82	0.82	0.88	0.86	0.87	0.87	-	-
		max	0.86	0.86	0.85	0.84	0.84	0.84	0.90	0.88	0.89	0.88	-	-
N=50	T=10	min	0.86	0.83	0.85	0.85	0.83	0.82	0.87	0.85	0.84	0.84	0.80	0.78
		avg	0.88	0.85	0.86	0.86	0.84	0.83	0.99	0.87	0.85	0.85	0.81	0.80
		max	0.90	0.87	0.88	0.87	0.86	0.86	0.90	0.88	0.88	0.87	0.83	0.82
	T=20	min	0.85	0.85	0.81	0.81	0.82	0.82	0.84	0.84	-	-	-	-
		avg	0.87	0.86	0.85	0.84	0.85	0.83	0.89	0.85	-	-	-	-
		max	0.88	0.88	0.86	0.85	0.87	0.86	0.90	0.87	-	-	-	-
	T=30	min	0.86	0.84	0.81	0.81	0.82	0.82	0.88	0.84	-	-	-	-
		avg	0.88	0.87	0.86	0.84	0.85	0.83	0.90	0.89	-	-	-	-
		max	0.90	0.89	0.86	0.85	0.87	0.86	0.91	0.90	-	-	-	-

Table 4.13. Service level of the heuristic solution with decomposition

	Sales price		low			high		
	Set-up cost		low	mid	high	low	mid	high
N=10	T=30	min	-	0.70	0.68	-	0.73	0.72
		avg	-	0.75	0.71	-	0.77	0.74
		max	-	0.77	0.75	-	0.79	0.76
N=30	T=20	min	-	0.74	0.73	-	0.71	0.62
		avg	-	0.76	0.73	-	0.77	0.75
		max	-	0.78	0.76	-	0.80	0.79
	T=30	min	-	0.74	0.71	-	0.74	0.73
		avg	-	0.82	0.79	-	0.83	0.80
		max	-	0.83	0.81	-	0.87	0.86
N=50	T=20	min	0.69	0.63	0.61	0.74	0.67	0.62
		avg	0.72	0.71	0.69	0.75	0.74	0.71
		max	0.75	0.73	0.70	0.77	0.76	0.74
	T=30	min	0.68	0.67	0.65	0.71	0.70	0.68
		avg	0.72	0.71	0.69	0.75	0.82	0.80
		max	0.80	0.78	0.75	0.82	0.79	0.77

CHAPTER 5

CONCLUSIONS AND FURTHER RESEARCH ISSUES

Profitability is the main driver for disassembly systems as regular production systems. Value hidden in the discarded products encourages the producers to be engaged in the part and material recovery operations. A producer wants to maximize the profit by gaining all possible value from the discarded products. Therefore, effective disassembly plans are required to reach the possible profitability of the disassembly systems.

In this thesis, we study a disassembly lot-sizing problem that is a medium (or short) term production planning problem. We consider the problem of determining the time and quantity of discarded products and intermediate items to be disassembled while maximizing total profit by selling the intermediate and leaf items over a finite planning horizon. The case of multiple product types with parts commonality is considered. It is assumed that the supply of discarded products is infinite. The purchased discarded products are immediately disassembled. When an item is disassembled, all its immediate child items are obtained, i.e. complete disassembly case is considered. Obtained intermediate and leaf items can be sold or kept in stock. Intermediate items can be disassembled further. Sales of intermediate and leaf items are the revenue sources. The considered cost items are purchasing cost of discarded products, fixed and variable disassembly costs and inventory holding cost.

We formulate the problem as an integer programming model. We state that the problem is NP-hard by referring to the study of Kim et. al. (2009). As the number of items and the length of the planning horizon increase, the solution time required by

the integer programming model increases exponentially. Therefore, we propose a heuristic solution approach to solve the problem within reasonable computational times. The heuristic includes relaxed models (RP) and integer models with single planning horizon. (RP) is a relaxed version of problem (P) obtained by relaxing integer variables and keeping binary variables. The relaxed and single period integer models are sequentially solved. Results of the relaxed models are used in single period integer models as bounds for the sales and disassembly quantities. In order to measure the performance of the heuristic approach, we randomly generate 2700 problem instances. 1155 problem instances out of 2700 are solved optimally; and 819 problems out of 2700 cannot be solved optimally within 3-hour time limit. Percent deviations of the heuristic solution from the optimal solution or best feasible solution within 3-hour time limit are reported. Percent deviation is less than 1% on the average. Average CPU time of the optimal solution is 2064.98 seconds, whereas the average CPU times for heuristic solution is 27.63 seconds. However, the heuristic solution requires more than an hour for 726 problems out of 2700. Since the relaxed problems are still mixed integer programming model, it sometimes requires high computational time. In this case, we decompose the planning horizon into 10-period equally in order to reduce the solution time. The solution time is reduced and 1.229% overall deviation on the average from the best feasible solution is obtained. The other performance measure which is service level defined as the percentage of demand fulfilled. On the average, 76% and 74% of the demand is satisfied in the optimal and heuristic solutions, respectively.

As a future study, one can deal with stronger formulations of the problem (P). It is possible that the formulation can be made tighter by adding some valid inequalities to our integer programming model or reformulating the model.

In the lot-sizing literature, meta heuristics are extensively used to solve hard problems in reasonable computational time. One can work on meta heuristic approaches to the problem (P) in a reasonable computational time. When all integer

variables including binary variables of problem (P) are relaxed, a linear programming (LP) model is obtained. The LP model is easily solved. Then, some binary variables can be fixed to zero or one by using meta heuristic approaches. Restricted integer programming model (P) with fixed binary decisions is re-solved. Therefore, the problem (P) can be solved quickly.

As a future work, one can make the environment more realistic by considering the capacity limits. Most of the disassembly operations are manual, therefore the labor is the most important resource for the disassembly firms. Defining labor hours for each disassembly operation and considering an aggregate available labor hours a capacitated version of our problem can be defined. The capacitated version is harder than the uncapacitated one. Providing heuristic solutions are important to solve the capacitated problem.

The assumption A4 which is backlogging not allowed given in Section 3.1 is employed to simplify the problem. Backlogging can be allowed to make the environment more realistic. Since the demand can be allowed to meet in following time periods, the feasible region of the problem with backlogging is larger. The problem consists of more decision variables and larger big-M values for disassembled items. Thus, allowing backlogging makes the problem harder.

In the study, we do not consider the cost of lost demand other than lost profit. Unsatisfied demand can be penalized with unit variable cost. The additional cost parameter is used in the objective function of the (P) model. The effects of the cost of lost demand can be tested on the problem instances.

REFERENCES

Adenso-Diaza, B., Santiago, G. C., Gupta, S. M., (2008), "Lot sizing in reverse MRP for scheduling disassembly", *International Journal of Production Economics*, 111, 741–751.

Clegg, J., Williams, D.J., Uzsoy R., (1995), "Production planning for companies with remanufacturing capability", *Proceedings of the third IEEE international symposium of electronics and the environment*, Orlando,. p. 186–91.

Gupta, S.M., Taleb, K.N., (1994), "Scheduling disassembly", *International Journal of Production Research*, 32, 1857–1866.

Gupta, S.M., Taleb, K.N., (1997), "Disassembly of multiple product structures", *Computers and Industrial Engineering*, 32, 949–961.

Güngör, A., Gupta, S.M., (1999), "Issues in environmentally conscious manufacturing and product recovery", *Computers and Industrial Engineering*, 36, 811–853.

Inderfurth, K., Langella, I.M., (2006), "Heuristics for solving disassemble-to-order problems with stochastic yields", *OR Spectrum*, 28, 73–99.

Kim, H.-J., Lee, D.-H., Xirouchakis, P., (2006a), "Two-phase heuristic for disassembly scheduling with multiple product types and parts commonality", *International Journal of Production Research*, 44, 195–212.

Kim, H.-J., Lee, D.-H., Xirouchakis, P., (2006b), "A Lagrangean heuristic algorithm for disassembly scheduling with capacity constraints", *Journal of the Operational Research Society*, 57, 1231–1240.

Kim, H.-J., Lee, D.-H., Xirouchakis, P., (2007), "Disassembly scheduling: literature review and future research directions", *International Journal of Production Research*, 45, 4465–4484.

Kim, H.-J., Lee, D.-H., Xirouchakis, P., (2009), "A branch and bound algorithm for disassembly scheduling with assembly product structure", *Journal of the Operational Research Society*, 60, 419–430.

Kongar, E., Gupta S.M., (2002), "A multi-criteria decision making approach for disassembly-to-order systems", *Journal of Electronics Manufacturing*, 11, 171–83.

Langella, I.M., (2007), “Heuristics for demand driven disassembly planning”, *Computer and Operations Research*, 34, 552–577.

Lee, D.-H., Kim, H.-J., Xirouchakis, P., (2004a), “Disassembly scheduling: an integer programming approach”, *Journal of Engineering Manufacture*, 218 (Part B), 1357–1372.

Lee, D.-H., Xirouchakis, P., (2004b), “A two-stage heuristic for disassembly scheduling with assembly product structure”, *Journal of the Operational Research Society*, 55, 287–297.

Lee, D.-H., (2005), “Disassembly scheduling for products with assembly structure”, *International Journal of Management Science*, 11, 63–78.

Spengler, T., Ploog, M., Schröter, M., (2003), “Integrated planning of acquisition, disassembly and bulk recycling: a case study on electronic scrap recovery”, *OR Spectrum*, 25, 413–442.