

DISASSEMBLY SCHEDULING ON PARALLEL RESOURCE ENVIRONMENT

BURAK GÖKGÜR

JULY 2013

DISASSEMBLY SCHEDULING ON PARALLEL RESOURCE ENVIRONMENT

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
IZMIR UNIVERSITY OF ECONOMICS


BY

BURAK GÖKGÜR

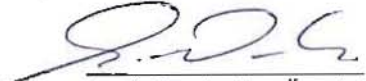
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
MASTER OF SCIENCE
IN
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

JULY 2013

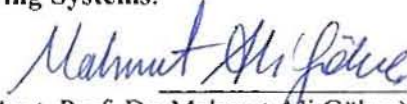
Approval of the Graduate School of Natural and Applied Sciences

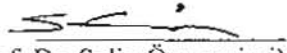

(Prof. Dr. Murat Aşkar)
Director

I certify that this thesis satisfies all the requirements for the degree of **Master of Science in Intelligent Production Systems** option of **Intelligent Engineering Systems**.


(Assoc. Prof. Dr. Arslan Örnek)
Head of Department

We have read the thesis entitled **Disassembly Scheduling on Parallel Resource Environment** prepared by **Burak GÖKGÜR** under supervision of **Asst. Prof. Dr. Mahmut Ali GÖKÇE** and **Asst. Prof. Dr. Selin ÖZPEYNİRCİ** and we hereby agree that it is fully adequate, in scope and in quality, as a thesis for the degree of **Master of Science in Intelligent Production Systems** option of **Intelligent Engineering Systems**.


(Asst. Prof. Dr. Mahmut Ali Gökçe)
Advisor


(Asst. Prof. Dr. Selin Özpeynirci)
Co-Advisor

Examining Committee Members:
(Chairman, Supervisors, Members)

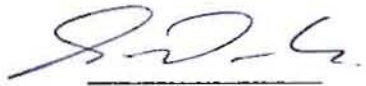


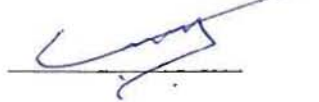
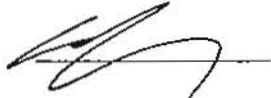
Assoc. Prof. Dr. Arslan Örnek
(Industrial Engineering Dept., IUE)

Asst. Prof. Dr. Mahmut Ali Gökçe
(Industrial Engineering Dept., IUE)

Asst. Prof. Dr. Selin Özpeynirci
(Industrial Engineering Dept., IUE)

Prof. Dr. Cemali Dinçer
(Industrial Engineering Dept., IUE)

Assoc. Prof. Dr. Bilge Karaçalı
(Electrical and Electronics Engineering Dept., IYTE)

ABSTRACT

DISASSEMBLY SCHEDULING ON PARALLEL RESOURCE ENVIRONMENT

Gökgür, Burak

M.Sc. in Intelligent Engineering Systems
Graduate School of Natural and Applied Sciences

Advisor: Asst. Prof. Dr. Mahmut Ali Gökçe
Co-Advisor: Asst. Prof. Dr. Selin Özpeynirci

July 2013, 62 pages

Disassembly systems obtain valuable parts from end-of-life products to remanufacture, reuse or recycle them. Also, pure material fractions, isolated hazardous substances and separated reusable parts and subassemblies are recovered by disassembly systems. Disassembly systems are investigated in two ways; disassembly planning and disassembly scheduling. Disassembly planning is to find optimal sequence of disassembly operations. Disassembly scheduling is to determine the quantity of items that will be disassembled with timing decisions over a planning horizon. In this thesis, we deal with the scheduling for disassembly systems. There are two main contributions of this thesis. The first is to develop a mathematical model that takes into account different perspectives than models that already exist in the literature. These perspectives are; disposal option of items, fill rate level of demand that must be satisfied to not to pay any penalty cost and demand for all items except root items. Another subject given attention in this thesis is the difference of the multi-resource consideration in disassembly systems, which has not been studied yet in the literature to the best of our knowledge. The second is to develop benchmark problem generation mechanism and evaluate problem characteristics based on problem size. Computational studies are carried out to evaluate performances of the mathematical model proposed.

Keywords: Disassembly Systems, Disassembly Scheduling, Mathematical Modeling,

ÖZ

PARALLEL KAYNAKLI ORTAMLARDA DEMONTAJ SİSTEMLERİNİN ÇİZELGELENMESİ

Gökgür, Burak

Akıllı Mühendislik Sistemleri Yüksek Lisans Programı
Fen Bilimleri Enstitüsü

Tez Danışmanı: Yrd. Doç. Dr. Mahmut Ali Gökçe
Ortak Tez Danışmanı: Yrd. Doç. Dr. Selin Özpeynirci

Temmuz 2013, 62 sayfa

Demontaj sistemleri, hurda veya atık ürünlerden üretimde tekrar kullanılabilir, geri dönüştürülebilir ürünleri elde etmek için tasarlanmış sistemlerdir. Zararlı parçaların ve içeriklerin ayrılması işlemleri de demontaj sistemlerinde yapılmaktadır. Demontaj sistemleri dahilinde iki farklı ana konu bulunmaktadır. Bu konular demontaj planlama ve demontaj çizelgelemedir. Demontaj planlamanın amacı, demontajın en iyi yapılma sırasını bulmaktır. Demontaj çizelgeleme ise, belirlenen amaç fonksiyonuna göre planlama ufku içerisinde hangi ürünün, hangi zamanda, kaç tane parçalanacağını bulmayı hedefler. Tezin iki ana amacı bulunmaktadır. Birinci amacı, demontaj çizelgeleme literatüründe bulunan modeller dışında, farklı etmenleri de göz önünde bulunduran matematiksel model geliştirmektir. Tezde, değerlendirilen ve şu ana kadar literatürde çalışılmamış belirgin etmenler ise; ürünlerin imha edilmesi, minimum sipariş karşılama oranı, demontaj sistemlerinin çoklu kaynak ortamında kullanılması ve her bir parça için talep olmasıdır. İkinci amacı ise test problemi yaratma mekanizması geliştirmek ve problemin karakteristik özelliklerini gösterecek geniş test problemleri yaratmaktır. Geliştirilen problem yaratma mekanizması ile matematiksel modelin performansı ölçülmüştür.

Anahtar Kelimeler: Demontaj Sistemleri, Demontaj Sistemlerinin Çizelgelenmesi, Matematiksel Modelleme

ACKNOWLEDGEMENTS

I cannot find words to express my sincere gratitude to my advisor Asst. Prof. Dr. Mahmut Ali Gökçe for his guidance, contribution, academic and personal support, and patience throughout all years that I have been in the Izmir University of Economics as student and assistant.

I also would like to express my unfeigned gratitude to my co-advisor Asst. Prof. Dr. Selin Özpeynirci for her support and contribution in my thesis and especially her contribution to my MSc years in the Izmir University of Economics as student and assistant.

My family, Erol Gökğür and Didem Yurtsev Gökğür, have been a constant source of eternal support and love. I am in endless debt to them for believing in and trusting me all my life. I would not be here if it weren't for you.

I am indeed grateful to my love Elif Ayvalı for her morale support, patience and endless love.

I want to thank Burcu Çelik, Serkan Osman Dibek and Oktay Karabağ for their endless support and priceless friendship. Three years that we have been will be always memorable. I also thank to Kaya Oğuz for his strong helps in programming language.

I lastly thank to Hakan Eğri for his help in providing me infrastructure services of laboratory in the Izmir University of Economics.

TABLE OF CONTENTS

ABSTRACT	iii
ÖZ	v
ACKNOWLEDGEMENTS.....	vi
TABLE OF CONTENTS.....	vii
1.INTRODUCTION.....	1
2.LITERATURE REVIEW.....	5
2.1. Uncapacitated Disassembly Scheduling.....	5
2.2. Capacitated Disassembly Scheduling.....	9
3.PROBLEM ENVIRONMENT	13
3.1. Capacitated Disassembly Scheduling Problem (CDSP).....	13
3.2. Problem Structure and Definition of CDSP on Parallel Resource Environment (CDSP-PR).....	15
3.3. Mathematical Model.....	19
4.COMPUTATIONAL EXPERIMENTATION	25
4.1. Design Parameters	26
4.1.1. Number of root items	26
4.1.2. Maximum number of levels.....	26
4.1.3. Maximum number of children per item.....	27
4.1.4. Probability of part commonality.....	27
4.1.5. Gozinto factor	28

4.1.6. Fill rate	28
4.1.7. Probability Distribution of number of children	28
4.2. Test-bed Problem Generating Mechanism and Settings.....	29
4.3. Computational Results	35
4.3.1. Analysis of Effects of Design Parameters.....	36
4.3.1.1. Effect of the Number of Root Items	37
4.3.1.2. Effect of the Maximum Number of Levels	38
4.3.1.3. Effect of the Maximum Number of Children per Item	40
4.3.1.4. Effect of Probability of Commonality	40
4.3.1.5. Effect of Gozinto Factor	43
4.3.1.6. Effect of Fill Rate	44
4.3.1.7. Effect of Probability Distribution of number of children	45
4.3.1.8. Interaction between Number of Root Items and Maximum Number of Levels	48
4.3.1.9. Interaction between Number of Root Items and Maximum Number of Children per Item	50
4.3.1.10. Interaction between Number of Root Items and Probability Distribution of number of children	50
4.3.1.11. Interaction between Maximum Number of Levels and Maximum Number of Children per Item	53
4.3.1.12. Interaction between Maximum Number of Levels and Probability Distribution of number of children	53
4.3.1.13. Interaction between Maximum Number of Children per Item and Probability Distribution of number of children.....	54
5.CONCLUSION AND FUTURE RESEARCH DIRECTIONS.....	58

REFERENCES..... 60

LIST OF TABLES

Table 2-1 Disassembly Scheduling Literature Review Summary	12
Table 4-1 Target root probability values.....	30
Table 4-2 Target level probability values	30
Table 4-3 Value of design parameter values.....	31
Table 4-4 Probability values of Gozinto factor.....	32
Table 4-5 Probability values of probability distribution of number of children for level 1	32
Table 4-6 Probability values of probability distribution of number of children for level 2.....	32
Table 4-7 Probability values of probability distribution of number of children for level 3.....	32
Table 4-8 Summary Results of Experimentation Set.....	35
Table 4-9 – 95% Confidence Level and Mean Values of Design Parameters in Execution time (sec.) for Main Effect Analysis.....	47
Table 4-10 - 95% Confidence Level and Mean Values of Design Parameter Pairs in Execution time (sec.) for Interaction Effect Analysis.....	56

LIST OF FIGURES

Figure 1.1 Phases of remanufacturing systems	2
Figure 3.1 An illustrative example of disassembly product structure for the single root item without parts commonality	16
Figure 3.2 An illustrative example of disassembly product structure for the multiple root items without parts commonality.....	17
Figure 3.3 An illustrative example of disassembly product structure for the multiple root items with parts commonality.....	17
Figure 4.1 Normality plot of execution time.....	36
Figure 4.2 Histogram of execution time	37
Figure 4.3 Effect of the number of root items levels to execution time.....	38
Figure 4.4 Effect of the number of root items levels to success percentage.....	39
Figure 4.5 Effect of the maximum number of levels to execution time	39
Figure 4.6 Effect of the maximum number of levels to success percentage.....	40
Figure 4.7 Effect of the maximum number of children per item to execution time ..	41
Figure 4.8 Effect of the maximum number of children per item to execution time ..	41
Figure 4.9 Effect of the probability of commonality to execution time.....	42
Figure 4.10 Effect of the probability of commonality to success percentage.....	42
Figure 4.11 Effect of the gozinto factor to execution time	43
Figure 4.12 Effect of the gozinto factor to success percentage.....	43
Figure 4.13 Effect of the fill rate to execution time	44
Figure 4.14 Effect of the fill rate to success percentage	44

Figure 4.15 Effect of the probability distribution of number of children to execution time.....	45
Figure 4.16 Effect of the probability distribution of number of children to success percentage	45
Figure 4.17 Interaction between number of root items and maximum number of levels in execution time.....	48
Figure 4.18 Interaction between number of root items and maximum number of levels in success percentage	49
Figure 4.19 Interaction between number of root items and maximum number of children per item in execution time.....	49
Figure 4.20 Interaction between number of root items and maximum number children per item in success percentage	50
Figure 4.21 Interaction between number of root items and probability distribution of number of children in execution time	51
Figure 4.22 Interaction between number of root items and probability distribution of number of children in success percentage.....	51
Figure 4.23 Interaction between maximum number of levels and maximum number of children per item in execution time	52
Figure 4.24 Interaction between maximum number of levels and maximum number of children per item in success percentage.....	52
Figure 4.25 Interaction between maximum number of levels and probability distribution of number of children in execution time.....	53
Figure 4.26 Interaction between maximum number of levels and probability distribution of number of children in success percentage	54
Figure 4.27 Interaction between maximum number of children per item and probability distribution of number of children.....	54
Figure 4.28 Interaction between maximum number of children per item and probability distribution of number of children in success percentage	55

CHAPTER - 1

INTRODUCTION

Importance of environmental challenges has become crucial for manufacturing and service firms and countries. Promoting focus on environmental awareness causes countries to enact environmental regulations. New legislations enforce firms to consider manufacturing operations in a more environmentally conscious way. Hence, remanufacturing firms are now responsible for collecting, re-using, recycling and disposing of products. Disposing of products is also another critical issue because of shortage of dumping sites. Many organizations are established to increase environmental awareness such as; EuP (Eco-design requirements for Energy using Products), WEEE (Waste Electronic and Electronic Equipment), RoHS (Restriction of the use of certain Hazardous Substance in electric and electronic equipment), ELV (Directives for End-of-Life Vehicle) and EMAS (Eco-Management-and-Audit Scheme). Also, Germany and United States enacted new regulations on automotive and electronics manufacturers to be responsible for their products at the end of life cycle (Pnueli and Zussman [4]). Remanufacturing, recycling and re-using processes are not only environmentally mandatory but also have economical gain. Firms can turn these processes into an economical advantage (De Brito and Dekker [10]). For instance, they can collect and re-use separated parts from end-of-life (EOL) products. In this way, they can use EOL items as raw or sub-assembly materials in remanufacturing processes.

Ilgın and Gupta [20] summarize main areas of environmentally conscious manufac-

turing and product recovery as product design, reverse and closed-loop supply chains, remanufacturing and disassembly. They defined remanufacturing as an industrial process that converts EOL product into a product with new conditions. Disassembly system is one of the most important phases of remanufacturing systems. Because the aim of remanufacturing is to obtain as many recovered parts as possible. Likewise, main purpose of disassembly is to obtain items as many as possible in an economic scale. Lee et al. [7] depicts the phases of remanufacturing systems shown in Figure 1.1.

In disassembly systems, end-of-life (EOL) products are used to obtain valuable parts, recover broken parts or dispose of environmental threatening parts. Disassembly operation is defined as a systematic method for separating a product into its constituent parts, components, subassemblies or other groupings on the work of Moore et al. [5].

Obtained parts can be used either for recycling or remanufacturing processes where they can be used as parts of new products as shown in Figure 1.1.

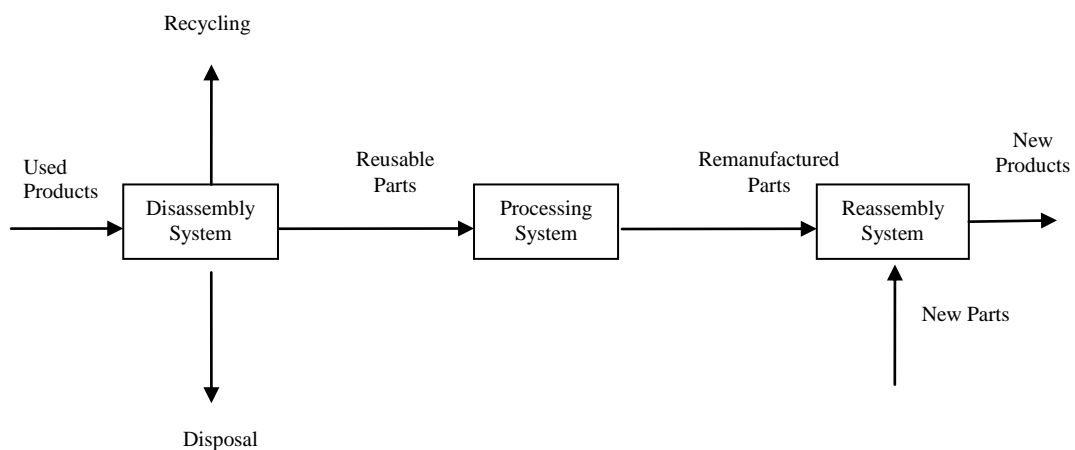


Figure 1.1 Phases of remanufacturing systems

Besides, scarcity of resources, shortened lifetime of products due to rapid development in technology and shortage of dumping and waste-incineration facilities can be seen as other causes of why disassembly operations become important. (Güngör and Gupta [6])

Kang and Xirouchakis [13] classified disassembly operations as; complete, incomplete and selective disassembly. In the complete disassembly operation, all parts of an EOL product are separated. In the incomplete operation, one or more parts are separated from an EOL product to remove harmful or damaged components. The aim in this case is to determine the disassembly sequence of a product. In the selective operation, final state of the EOL product is known in advance and its target is to determine a sequence to obtain the selected final state of product while minimizing number of removal of other parts.

In disassembly systems, there are two main decisions that need to be made. The first one is determining the sequence of disassembly operations, which is called *disassembly planning*. Disassembly planning is a one-time decision and deals with the problem of determining the best sequence of disassembling a product into its parts. The second decision is to determine the quantity of items that will be disassembled with timing decisions, which is referred as *disassembly scheduling*. Disassembly scheduling deals with the problem of scheduling of the ordering and disassembly of EOL products to satisfy demand for parts over a planning horizon. Unlike disassembly planning, disassembly scheduling plan has to be made for each planning horizon, based on demand, availability, capacity and other constraints. Disassembly sequence plan is one of the inputs of the disassembly scheduling.

Kim et al. [20] consider the main differences between material requirement planning (MRP) and disassembly scheduling. Although disassembly can be seen as the reverse form of the MRP, there are significant differences between the two from an operational point of view. While in MRP, multiple part/subassemblies are assembled to form a single end item, in disassembly, an end item is disassembled to obtain multiple parts with demand. This issue, having multiple end items, is called as *divergence property* of disassembly scheduling problem. Another important difference is *surplus inventory*. Inevitable surplus inventory results from the situation where two or more different parts in disassembly structure are obtained by disassembling same parent item. Hence, this situation may lead to significant amount of inventory. Surplus inventory can be held to satisfy demand of future periods. These differences make the disassembly scheduling problem more complicated than MRP.

This thesis is about the capacitated disassembly scheduling problem on parallel resources. As to the best of our knowledge, this study is first to consider parallel resources, disposal of item, selling parts, penalty of unsatisfied demand and demand for all items except root items in addition to disassembling and holding items all at the same time. The objective is to maximize net revenue from disassembling operations. Net revenue is composed of sales and costs that are accrued by holding inventory, disassembling, disposing and also the penalty for unsatisfied demand. Moreover, this study is the first to consider a wide range of bill-of-material (BOM) structures; single/multiple product types and without/with part commonalities.

In the next chapter, the literature review is given about uncapacitated disassembly scheduling and capacitated disassembly scheduling. Also, our motivation of study is given in Chapter 2. In Chapter 3, the problem definition and basic mathematical model of capacitated disassembly scheduling are presented. Moreover, some basic disassembly product structure examples are given with disassembly notations. Then, our proposed mixed-integer programming model is given with novelties. Chapter 4 explains selected design parameter for experimentation. Also, testbed generating mechanism is presented and computational experimentation and results are given with our data. Based on results of experimentation, effect of each design parameter and relationship between them is discussed in Chapter 4. Conclusion and future research directions are given in Chapter 5.

CHAPTER - 2

LITERATURE REVIEW

Studies about disassembly systems include disassembly scheduling and sequencing (planning) problems. Kim et al. [20] defined disassembly scheduling problem as determining the quantity and timing of the EOL products while satisfying the demand of their parts over a planning horizon. Kang and Xirouchakis [13] described that disassembly sequencing problem deals with the feasible disassembly sequence generation and sequence optimization. In this thesis, we deal with the problem of disassembly scheduling and therefore focus on disassembly scheduling literature.

The majority of studies in disassembly scheduling literature deal with the problem of how to disassemble parent items with/without capacity restrictions so that demand of leaf items are satisfied while minimizing the number of items to be disassembled or total costs incurred by set-up, disassembly operations, holding items and purchasing costs over a planning horizon. To evaluate performance of mathematical models and algorithms, computational experiments are made on specific EOL items or generated instances that cover only a subset of all alternatives for product tree structures. Disassembly scheduling literature can be investigated as uncapacitated and capacitated. Detailed review of uncapacitated and capacitated disassembly scheduling literature is given in Section 2.1 and Section 2.2, respectively.

2.1. Uncapacitated Disassembly Scheduling

Uncapacitated disassembly scheduling deals with the problem of quantity and timing of EOL products while satisfying demands of leaf items. It does not consider time capacity restriction to disassemble items. This subsection presents studies on uncapacitated disassembly scheduling in chronological order.

Study of Gupta and Taleb [1] is the first to consider disassembly scheduling problem. They discussed similarities and differences between disassembly and assembly systems for the case of single product type without parts commonality. In the considered problem, there is demand for only leaf items. They proposed MRP-like algorithm to solve disassembly scheduling problem. In the algorithm, they calculate the net requirement of parent of each item starting from the highest numbered item (leaf items). This calculation is made for each period in planning horizon. By calculating net requirement of each parent, disassembly amount of each parent is also obtained at the same time. Excess amount of disassembled items are carried to satisfy demand of future periods. In computational experimentation, an illustrative example of the algorithm on a specific product structure is presented. In following studies of uncapacitated disassembly scheduling, almost all objectives are to minimize total cost; sum of disassembly, inventory and set-up.

Taleb and Gupta [2] considered a disassembly scheduling problem to find ordering and disassembly schedule so as to fulfill demand of all the leaf items while minimizing the total disassembly cost. They presented two companion algorithms; Core Algorithm and Allocation Algorithm. Core algorithm determines the number of root items to be disassembled to minimize total disassembly cost. Allocation algorithm determines a disassembled schedule for the roots and subassemblies by allocating the requirements for each period in planning horizon. In computational experimentation, out of generated 25 cases, the heuristic finds 19 optimal solutions. Deviation of 6 cases is between 1.4% and 3.8%. Information on the experimentation data such as the number of items, the number of levels in product structures is not provided.

Taleb et al. [3] modified their previously proposed heuristic algorithm (Taleb and Gupta [2]) in case of parts commonality to determine the quantity and operations schedule of disassembly. To handle commonality case, they decomposed every

common part of a product structure into different item numbers. After adjusting item numbers, they calculate net requirement of each parent item starting from highest numbered item (leaf items) for each period in planning horizon. By calculating net requirement of each parent, disassembly amount of each parent is also obtained at the same time. Excess amount of disassembled items are carried to satisfy demand of future periods. In computational experimentation, they gave an illustrative example of the algorithm on a specific product structure.

Studies until mentioned so far did not consider the case of multiple product types with parts commonality. The study of Kim et al. [9] is the first to deal with multiple product types with parts commonality and develop mixed-integer programming model. They developed a mixed-integer programming and a heuristic algorithm for disassembly scheduling problem. The case of multiple product types with parts commonality is considered for the objective of minimizing the sum of setup, disassembly operation and inventory holding costs subject to demand constraints for only leaf items. In proposed heuristic algorithm, linear programming relaxation approach is used. Proposed heuristic consists of two steps. In the first step, mathematical model is solved after setup, disassembly and inventory variables are relaxed. Relaxed solutions are rounded down. In the second step, rounded-down solution obtained in Step 1 is modified so that constraints in the original mathematical model are satisfied. To show the effectiveness of LP relaxation, a case study (inkjet printer) was performed and deviation of proposed approach is 1.29% for 10 periods, 1.03% for 15 periods and 1.33% for 20 periods.

Lee et al. [12] proposed several mathematical models on uncapacitated disassembly scheduling problem. Objective function of all considered models is same and is to minimize sum of setup, inventory, purchase and disassembly operation costs. There is demand only for leaf items. Investigated cases are for single product type with/without parts commonality, and multiple product type with parts commonality. The number of items in product structures is set to be 10, 20 and 30 in experimentation.

Lee and Xirouchakis [11] worked on EOL products with assembly structure for the

objective of minimizing total costs; composed of purchase, setup, inventory holding and disassembly operation costs. They developed a mathematical model and two-stage heuristic during the first stage of the heuristic an initial solution is obtained by the minimal latest ordering and disassembly schedule. Improvement (second) step is done by iteratively considering trade-offs among different cost factors. To evaluate the performance of heuristic, 750 problems are randomly generated. Two-stage heuristic gives very near optimal solutions between 0.5% and 1%. The number of items in the experimentation is set to be 10, 20, 30, 40 and 50.

Another work on disassembly scheduling belongs to Barba-Gutierrez et al. [17]. They extended the work of Gupta and Taleb [1] by incorporating lot sizing decisions. They presented an algorithm for reverse MRP to facilitate the use of lot sizing. They also conducted statistical analysis for cost factors to evaluate the behavior of the proposed algorithm.

Kim et al. [18] proposed a branch and bound algorithm based on the problem analysis that incorporates the Lagrangean relaxation-based upper and lower bounds for the case of single product type without parts commonality with the objective of minimizing total cost; setup and inventory holding costs subject to demand constraints for leaf items. They proved that the problem is NP-Hard. Computational experiments were done on randomly generated problems. 625 problems were generated, that is, 25 problems for each combination of five levels of the number of items (10, 15, 20, 25 and 30) and five levels of the number of periods (10, 15, 20, 25 and 30). For each level of the number of items, five disassembly structures were randomly generated. For each disassembly structure, five problems with different data were generated for each level of the number of periods. They concluded that the branch and bound algorithm suggested is not viable alternative for large-sized problems.

Prakash et al. [22] developed a mathematical model and constraint-based simulated annealing approach for disassembly scheduling problem by considering part commonalities with the aim of minimizing inventory level subject to demand constraints for leaf items. This study is first to propose a metaheuristic approach in disassembly scheduling literature. Evaluation of metaheuristic approach is made on an illustrative

example.

2.2. Capacitated Disassembly Scheduling

This subsection presents works on capacitated disassembly scheduling. Capacity term on disassembly scheduling can be defined as total available time of resources to disassemble items. There are a few studies in the literature on capacitated disassembly scheduling and all existing studies found so far consider single resource.

The first study on capacitated disassembly scheduling is by Lee et al.[8]. They developed an integer programming model for the case of single product type without parts commonality to determine optimal schedule for ordering and disassembling products over the planning horizon with the objective of minimizing the sum of purchase, inventory and disassembly costs subject to capacity restrictions and demand constraints only for leaf items. Model was tested on a case study.

Another study with capacity restriction belongs to Kim et al. [15]. They proposed integer programming model with the objective of minimizing the number of products disassembled subject to capacity restrictions and demand constraints. Demand constraints are considered only for leaf items. Also, they developed an alternative solution algorithm based on optimal solution properties. In the proposed solution algorithm, an initial (infeasible) solution is obtained by relaxing capacity constraints in the mathematical model. Then, they try to obtain a feasible solution by iteratively changing the initial solution while considering the capacity constraints for each period in the planning horizon. Computational experimentation is made on a case study (inkjet printer) and specific generated test problem set. Number of items in the experimentation is set to be 20, 40, 60, 80 and 100. Over 750 test problems and case study, proposed solution algorithm gave the optimal solution in all problems and requires much smaller computational time than MIP solver CPLEX 9.0.

Kim et al. [14] proposed an integer programming model and Lagrangean heuristic for disassembly scheduling with capacity constraints for single product type without part

commonality. The aim is to minimize total cost; setup, disassembly operation and inventory holding costs. There is demand only for leaf items. In proposed solution algorithm, first, original problem is reformulated so that Lagrangean relaxation can be applied more effectively. By reformulating the problem, relaxed problem can be decomposed into single item lot-sizing problems which can be solved with a polynomial time algorithm. Second, Lagrangean relaxation heuristic algorithm is proposed to find good feasible solutions while considering the trade-offs among relevant costs. Percentage deviation of proposed algorithm from lower bound is between 0.18% and 0.46% and from optimal solution is between 0.08% and 0.19%. Execution time of the algorithm is less than 17 seconds. The considered numbers of item levels in the experimentation are 10, 20, 30, 40 and 50.

There are also a number of review studies on disassembly scheduling. Interested readers are referred to Lee et al. [7], Kim et al. [16], Ilgin and Gupta [21] and Morgan and Roger [23] for a more detailed review on disassembly scheduling. Table 2.1 shows the comparative summary information about disassembly scheduling literature.

There is limited number of studies about capacitated disassembly scheduling problem. Among existing studies, none of the models take into account different types of resources. All consider single resources. Also, in all considered problems, there is demand for only leaf items. All of the models in literature are limited in terms of bill-of-material (BOM) structures considered. Also in the literature, there is no information about how level of BOM structures depth is determined. Evaluations of proposed methods and mathematical models have to be tested under a broad range of BOM structures. Objective function of all mathematical models is to minimize total cost or total number of EOL items disassembled. None of the models consider that subassemblies/parts can be sold to be used in remanufacturing or reusing process.

Because of the nature of the disassembling, there are indispensable inventories. When a parent item is disassembled, all of its children parts are inevitably obtained even if there is no demand for some children parts. These excess parts either can be kept in inventory or can be disposed of. There is a trade-off between holding parts in

inventory to be used in future periods and disposing them. For instance, if there is no demand for future periods for an excess item, it can be disposed of. However, environmental legislations and rules affect the amount of disposed items at each period. Based on demand of future periods, an excess item can be kept in inventory but holding items in inventory also costs for each period. Because of environmental legislations, limited available capacity of dumping sites and costs, disposing and holding items become important considerations. These decisions and trade-offs between them also are not considered so far in the disassembly scheduling literature.

This thesis focuses on capacitated disassembly scheduling problem on parallel resources with the objective of maximizing net revenue from disassembly operations by considering sales as revenue and inventory, disassembly operations, penalty and disposal as cost factors. The main objective of the study is to develop a mathematical model that also considers the leading different perspectives than models that already exist in the literature.

Authors	Problem Type		Problem Characteristics			Solution Approaches			Experimentation Set
	Uncapacitated Disassembly Scheduling	Capacitated Disassembly Scheduling	Demand Type	Parts Commonality	Multiple EOL items	Mathematical Programming	Heuristics	Metaheuristics	
Gupta, Taleb (1994)	✓	x	Leaf items	x	x	x	✓	x	Specific product
Taleb, Gupta (1997)	✓	x	Leaf items	x	✓	x	✓	x	Randomly generated 25 problems
Taleb et al. (1997)	✓	x	Leaf items	✓	✓	x	✓	x	Specific product
Kim, et al. (2003)	✓	x	Leaf items	✓	✓	✓	✓	x	Specific product (inkjet printer)
Lee, et al. (2004)	✓	x	Leaf items	✓	✓	✓	x	x	Randomly generated problems
Lee, Xirouchakis (2004)	✓	x	Leaf items	x	x	✓	✓	x	Randomly generated 750 problems
Barba-Gutierrez et al. (2007)	✓	x	Leaf items	x	x	x	✓	x	Randomly generated 6480 problems
Kim, et al. (2009)	✓	x	Leaf items	x	x	✓	✓	x	Randomly generated 625 problems
Prakash et al. (2012)	✓	x	Leaf items	✓	x	✓	x	✓	Specific product
Lee et al. (2002)	x	✓	Leaf items	x	x	✓	x	x	Specific product (inkjet printer)
Kim et al. (2006)	x	✓	Leaf items	x	x	x	✓	x	Randomly generated 750 problems
Kim et al. (2006)	x	✓	Leaf items	x	x	✓	✓	x	Randomly generated 750 problems

Table 2-1 Disassembly Scheduling Literature Review Summary

CHAPTER - 3

PROBLEM ENVIRONMENT

3.1. Capacitated Disassembly Scheduling Problem (CDSP)

In the Capacitated Disassembly Scheduling Problem, there are I number of items in a disassembly product structure and T number of periods in a planning horizon. There is demand only for each leaf item for each period that has to be satisfied. Non-leaf items are used for satisfying demand of leaf items, i.e. non-leaf items have dependent demand. Excess items are kept in inventory to satisfy demand of future periods or to be disassembled in future periods. Also, there is available time capacity that is used in disassembling operations for each period. Total disassembly time of disassembled items cannot be greater than available time capacity for each period. In most CDSP studies, the objective function is to minimize the sum of purchasing, disassembly operation, set-up and inventory holding costs. Cost parameters can be time-varying, i.e. they can vary in different periods in a planning horizon. Set-up times occur because many disassembly operations require manual labor work to change resource (machine) settings. If set-up time and cost is not considered explicitly, it is usually added to the corresponding disassembly processing time and cost. Purchase cost incurs when EOL items are bought. If items are kept in inventory to be used for future periods, then inventory holding cost occurs.

Lee et al. [8] introduced the Capacitated Disassembly Scheduling Problem (CDSP) for the case of single product type without parts commonality. They defined the

problem as “For a given disassembly product structure, the problem is to determine the ordering schedule of the root item and the disassembly schedule of all parent items while satisfying the demands of leaf items over the planning horizon, subject to capacity restriction in each period. The objective is to minimize the sum of purchase, inventory holding and disassembly operation costs.” Capacity is defined as the sum of individual resource capacities. Disassembly processing time of each parent item is assumed to be constant for each period. Other assumptions are given below:

- Disassembly product structure is given
- There is no shortage of each item in product tree
- Demands for leaf items are given and deterministic
- Backlogging is not allowed and demand should be satisfied on time
- Disassembly process is perfect and defective parts are not considered
- Disassembly lead times with discrete time scales are given and deterministic

They proposed an integer programming model given below:

Sets

I : the number of items in the disassembly product structure

T : the number of time periods in the planning horizon

Parameters

p_t : purchase cost of the product in period t

h_i : inventory cost for one unit of item i per period

d_i : disassembly cost for one unit of parent item i

g_i : disassembly processing time of parent item i

C_t : available aggregated capacity in period t , i.e., sum of individual worker (or machine) capacities

R_{it} : demand requirement of leaf item i in period t

a_{ij} : number of units of item j obtained by disassembly of one unit of item i ($i < j$)

s_{it} : external scheduled receipt of item i in period t

$\varphi(i)$: parent of item i

l_i : disassembly lead time (DLT) of item i

I_{i0} : initial inventory of item i

Decision Variables

Z_t : purchase quantity of product (number of ordered products) in period t

X_{it} : amount of item i disassembled in period t

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

$$\text{Minimize} \quad \sum_{t=1}^T p_t Z_t + \sum_{i=1}^{i_l} \sum_{t=1}^T d_i X_{it} + \sum_{i=1}^I \sum_{t=1}^T h_i I_{it} \quad (3.1)$$

subject to

$$I_{1t} = I_{1,t-1} + s_{1t} + Z_t - X_{1t} \quad \forall t = 1, \dots, T \quad (3.2)$$

$$I_{it} = I_{i,t-1} + s_{it} + a_{\varphi(i),i} X_{\varphi(i),t-l_{\varphi(i)}} - X_{it} \quad \forall i = 2, \dots, i_l ; \forall t = 1, \dots, T \quad (3.3)$$

$$I_{it} = I_{i,t-1} + s_{it} + a_{\varphi(i),i} X_{\varphi(i),i-l_{\varphi(i)}} - R_{it} \quad \forall i = i_l + 1, \dots, l ; \forall t = 1, \dots, T \quad (3.4)$$

$$\sum_{i=1}^{i_l} g_i X_{it} \leq C_t \quad \forall t = 1, \dots, T \quad (3.5)$$

$$Z_t \geq 0, \text{ integer} \quad \forall t = 1, \dots, T \quad (3.6)$$

$$X_{it} \geq 0, \text{ integer} \quad \forall i = 1, \dots, i_l ; \forall t = 1, \dots, T \quad (3.7)$$

$$I_{it} \geq 0, \text{ integer} \quad \forall i = 1, \dots, l ; \forall t = 1, \dots, T \quad (3.8)$$

The objective function (3.1) minimizes the sum of purchase, disassembly operation and inventory holding costs. Constraint set (3.2) is inventory balance constraint for the root item. Constraints (3.3) and (3.4) are inventory flow equations for parent items except the root item and leaf items, respectively. Constraint set (3.5) state that the total disassembly processing time of disassembled items cannot exceed available time capacity for each period. Constraints (3.6)-(3.8) are the set constraints.

The study of Lee et al. [8] is the first to consider capacity restriction in disassembly scheduling literature for single resource. Other studies, which deal with CDSP, such as [14] and [15] build new solution approaches on this problem. In the following subsection, problem structure and definition of CDSP on parallel resource environment is explained.

3.2. Problem Structure and Definition of CDSP on Parallel Resource Environment (CDSP-PR)

In a disassembly structure, EOL (or root) items are the source items to be disassembled to sell or to obtain re-usable items. Note that in this thesis we use the terms root

items and EOL items interchangeably. The root items do not have any parent items and they are at the top (level 0) of disassembly BOM structures. Intermediate parts have at least one parent and at least one child. Leaf items are at the bottom of disassembly BOM structures and they cannot be disassembled further and they do not have any children. Disassembly BOM structures can vary based on the number of root items and whether there is parts commonality in the structure or not. Different types of disassembly BOM structures are given in Figures 3.1, 3.2 and 3.3. An illustrative example of disassembly product structure for the case of single root item without parts commonality can be seen in Figure 3.1.

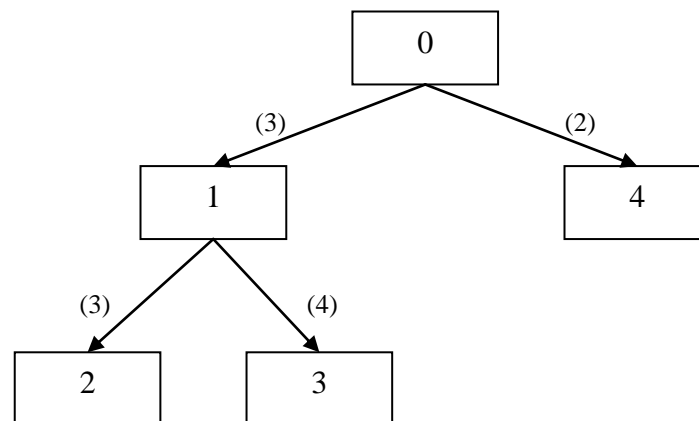


Figure 3.1 An illustrative example of disassembly product structure for the single root item without parts commonality

In Figure 3.1, item 0 is the root item. Items 1 and 4 are intermediate items. Items 2 and 3 are leaf items. The numbers in parentheses represent the yield of the item when its parent is disassembled, e.g. one unit of item 1 is disassembled into three units of item 2 and four units of item 3. In this figure, intermediate item 1 is the parent of items 2 and 3, while items 2 and 3 are the children of item 1. An example of disassembly product structure for the multiple root items without parts commonality is given in Figure 3.2.

In Figure 3.2, items numbered as 0 and 5 are root items of each disassembly tree structure respectively. By disassembling one unit of item 5, one unit of item 6 and five units of item 7 are obtained. In the same manner, three units of item 2 and four units of item 3 are obtained when one unit of item 1 is disassembled. Item 1 is only intermediate item in product structures. Items 2, 3, 4, 6 and 7 are leaf items.

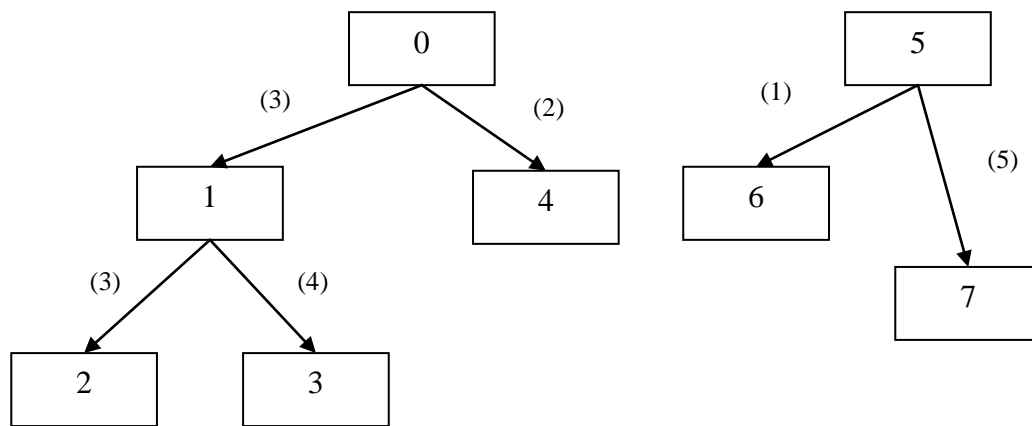


Figure 3.2 An illustrative example of disassembly product structure for the multiple root items without parts commonality

Another illustrative example of product structure for the multiple root items with parts commonality is given in Figure 3.3.

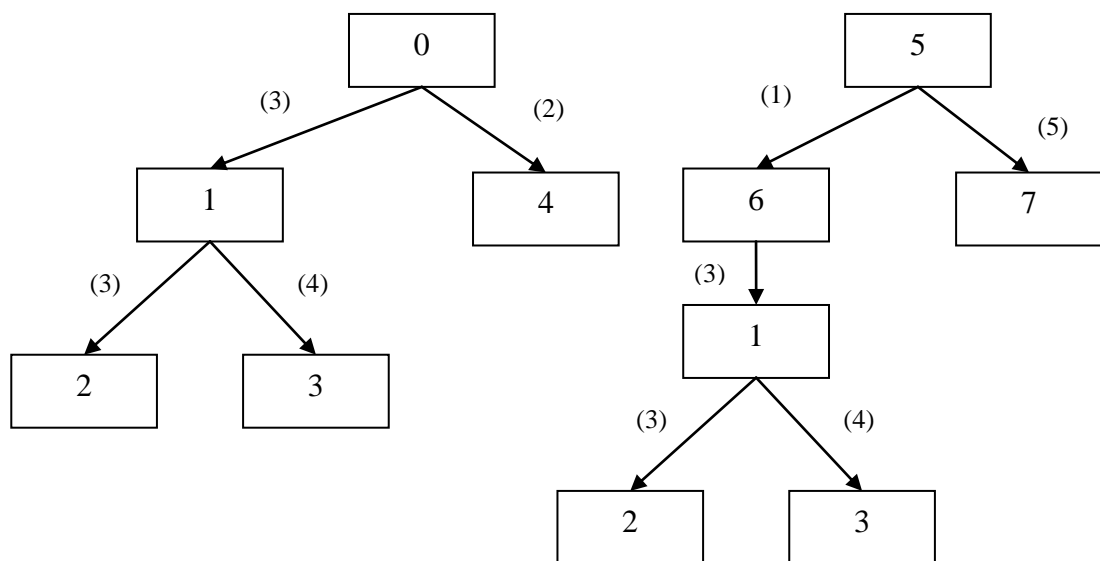


Figure 3.3 An illustrative example of disassembly product structure for the multiple root items with parts commonality

In the case of disassembly product structure for the multiple root items with part commonality, at least an item, except root item, has to show up in more than one disassembly structures. For example, item 1 is common item in Figure 3.3. It has more than one parent, items 0 and 5. Children of item 1 can also be obtained in two structures.

In the Capacitated Disassembly Scheduling Problem on parallel resource environ-

ment (CDSP-PR), there are I number of items, T number of periods and J number of resources. Parallel resources are available to be used in disassembling root items and intermediate items. These items have to be assigned to a resource to be disassembled. Each item can be disassembled only on one resource for each period. *Disassembling time* of root and intermediate items depends on the resource used. Each resource has limited *capacity* for each period to disassemble non-leaf parts. *Disassembling cost* of each non-leaf part also depends on resource. Leaf items cannot be disassembled further so there is no disassembling time and cost for them. *Setup time* and *cost* of each item is included to disassembly time and cost. The number of EOL items available is not known in advance. Therefore, to fulfill demand of items, number of EOL items to be purchased must be decided. For an EOL item, there is a unit *purchasing cost*. EOL products are obtained by recycling processes. Purchasing cost of EOL items can be considered as negligible.

There is a demand for each item, not limited to leaf items, except root item. Demand of each item does not have to be satisfied fully. There is a ratio for each item that indicates the minimum level of demand that should be satisfied, called *fill rate*. It is enough to satisfy fill rate demand for an item to avoid any penalty. However, if level of satisfied demand for each item falls short of the fill rate, *penalty cost* occurs for each unit of unsatisfied demand below the specified fill rate.

Another issue to be considered is sales of items. Each item, except root items, can be sold to be used in remanufacturing process. There is a *selling price* for each item for each period. All parts except the root items can be sold. Excess parts can either be kept in inventory to satisfy future demand or disposed of. There is a *disposal cost* for each part. This may reflect the actual cost of disposing of item and/or costs associated with environment constraints. Except root items, all parts can be disposed of. If disposed part is not root item, its parent item(s) have to be disassembled to obtain same part again.

If a part is held to be used in future periods, *holding cost* incurs. All these considerations cause many trade-offs. For instance, an intermediate part can either be disassembled to obtain its children or held in inventory to be used in future periods or sold

to gain profit otherwise disposed of. All these alternatives make the problem complex.

Kim et al. [18] prove that disassembly scheduling for the case of single product type without parts commonality is NP-Hard. The problem considered also includes parallel resource environment and demand for each part except root items. Therefore, it can be said that considered problem is also NP-Hard because NP-Hard problem is a subproblem of the problem considered in this thesis.

The problem considered in this study, *capacitated disassembly scheduling problem on parallel resource environment*, can be defined as follows:

“For a given disassembly product structure with/without parts commonality and disassembly sequence, the problem is to determine the quantity and assignment of purchased root items, disassembled, sold, held and disposed items in each period while satisfying demand and capacity constraints over a planning horizon with discrete time periods with the objective of maximization of the total net revenue.”

Assumptions made in the problem are as follows:

- Defective parts are not considered. Perfect disassembly is assumed.
- When an item is disassembled, its children are obtained within same period.
- Ordering lead time is assumed to be negligible
- Backlogging is not allowed. Unsatisfied demand is penalized but not carried over.

3.3. Mathematical Model

We propose the following Mixed-Integer Programming (MIP) model for the problem CDSP on parallel resource environment. Definition of sets, parameters, decision variables used in mathematical model is given as follows:

Sets

I : number of items, $i, q = 1, \dots, I$

T : number of periods, $t = 1, \dots, T$

J : number of resources, $j = 1, \dots, J$

PS_i : parent set of item i , $i = 1, \dots, I$

$root$: set of root items

$leaf$: set of leaf items

Parameters

P_{it} : selling price of item i in period t

PN_i : penalty cost of not satisfying one unit of item i

DM_{qi} : number of item i obtained from disassembling one unit of item q

D_{it} : demand of item i in period t

β_{it} : fill rate of item i in period t

H_{it} : holding cost of item i in period t

IC_{it} : disposal cost of item i in period t

DT_{ij} : disassembly time of item i on resource j

CDT_{ij} : cost of disassembling one unit of item i on resource j

CAP_{jt} : available time capacity of resource j in period t

O_{it} : purchasing cost of EOL item i in period t

Decision Variables

A_{it} : number of purchased EOL item i in period t

$R_{ijt} : \begin{cases} 1, & \text{if item } i \text{ is assigned to resource } j \text{ in period } t \\ 0, & \text{otherwise} \end{cases}$

X_{ijt} : number of item i disassembled on resource j in period t

I_{it} : number of item i in inventory at the end of period t

Y_{it} : number of disposed item i in period t

S_{it} : number of sold item i in period t

U_{it} : amount of unsatisfied demand of item i in period t

Mixed-Integer Programming (MIP) model is given below:

$$\text{Maximize} \quad \sum_{t=1}^T \sum_{i=1}^I (P_{it} S_{it} - I_{it} H_{it} - PN_i U_{it} - IC_{it} Y_{it} - O_{it} A_{it}) - \left(\sum_{j=1}^J X_{ijt} CDT_{ij} \right) \quad (3.9)$$

subject to

$$U_{it} \geq \beta_u D_{it} - S_{it} \quad i=1, \dots, I ; t=1, \dots, T \quad (3.10)$$

$$S_{it} \leq D_{it} \quad i=1, \dots, I ; t=1, \dots, T \quad (3.11)$$

$$I_{it} = I_{i,t-1} + A_{it} - \left(\sum_{j=1}^J X_{ijt} + S_{it} + Y_{it} \right) \quad i=1, \dots, root ; t=1, \dots, T \quad (3.12)$$

$$I_{it} = I_{i,t-1} + \sum_{q \in PS_i} DM_{qi} \sum_{j=1}^J X_{qjt} - \left(\sum_{j=1}^J X_{ijt} + S_{it} + Y_{it} \right) \quad \begin{array}{l} i = root + 1, \dots, I \\ t = 1, \dots, T \end{array} \quad (3.13)$$

$$\sum_{j=1}^J R_{ijt} \leq 1 \quad \begin{array}{l} i = 1, \dots, I \mid i \notin leaf \\ t = 1, \dots, T \end{array} \quad (3.14)$$

$$\sum_{i=1 \mid i \notin leaf}^I DT_{ij} X_{ijt} \leq CAP_{jt} \quad j=1, \dots, J ; t=1, \dots, T \quad (3.15)$$

$$X_{ijt} \leq \left(\sum_{t \in T} \sum_{i \in I} D_{it} \right) R_{ijt} \quad \begin{array}{l} i = 1, \dots, I \mid i \notin leaf \\ j = 1, \dots, J ; t = 1, \dots, T \end{array} \quad (3.16)$$

$$I_{i,0} = 0 \quad i = 1, \dots, I \quad (3.17)$$

$$I_{i,T} = 0 \quad i = 1, \dots, I \quad (3.18)$$

$$X_{ijt} \geq 0 \quad \begin{array}{l} i = 1, \dots, I ; j = 1, \dots, J \\ t = 1, \dots, T \end{array} \quad (3.19)$$

$$R_{ijt} \in \{0, 1\} \quad \begin{array}{l} i = 1, \dots, I ; j = 1, \dots, J \\ t = 1, \dots, T \end{array} \quad (3.20)$$

$$I_{it} \geq 0 \quad i = 1, \dots, I ; t = 1, \dots, T \quad (3.21)$$

$$Y_{it} \geq 0 \quad i = 1, \dots, I ; t = 1, \dots, T \quad (3.22)$$

$$S_{it} \geq 0 \quad i = 1, \dots, I ; t = 1, \dots, T \quad (3.23)$$

$$A_{it} \geq 0 \quad i = 1, \dots, I ; t = 1, \dots, T \quad (3.24)$$

$$U_{it} \geq 0 \quad i = 1, \dots, I ; t = 1, \dots, T \quad (3.25)$$

The objective function (3.9) maximizes the net total revenue from disassembly operations. In the first summation term of the objective function, the first term is total profit realized by selling items. Second term is total holding cost incurred by items kept in inventory. Total penalty cost of unsatisfied demand for all items and all periods is calculated in the third term. Fourth term is total disposal cost of items incurred by disposing items. Total purchasing cost of EOL items is calculated in the fifth term. Second summation term of the objective function calculates the total dis-

assembly cost of non-leaf items for all resources and periods.

Constraint set (3.10) enables to calculate the amount of unsatisfied demand for each item and each period. If number of sold item i for period t is greater than fill rate demand ($\beta_{it}D_{it}$), there will be no unsatisfied amount for item i . Otherwise, unsatisfied amount of item i is equal to the difference between amount of selling and fill rate demand.

Constraint set (3.11) states that number of sold item cannot be greater than its demand for each period.

In the constraint set (3.12), inventory balance equation for root items is stated. Inventory level of a root item for each period is equal to the sum of inventory level of previous period, purchased number of item for that period and number of items disassembled, sold and disposed of for that period.

Constraint set given by (3.13) is inventory flow constraint for all items except root items. Inventory level of a non-root item for period t depends on three terms as given in equation (3.13). First term is the inventory level of the non-root item that comes from previous period. Available number of the non-root item obtained by disassembling its parents for that period is stated in the second term. Third term is the sum of how many of the non-root item is disassembled, sold and disposed of for that period.

Constraint set (3.14) indicates that each item, except leaf items, can be assigned to at most one resource for each period. If a non-leaf item is going to be disassembled, then it has to be assigned to one resource. However, a non-leaf item does not have to be disassembled for each period. There can be such a situation that selling all available number of a non-leaf item or keeping all available number of the non-leaf item in the inventory for that period is more profitable. Also, if there is no available time capacity in all resources, the non-leaf item cannot be disassembled. Then, corresponding R_{ijt} binary variable takes value zero.

Time capacity constraints for each resource and for each period are presented in

(3.15). The constraint states that total disassembly time of non-leaf items assigned to resource j cannot be greater than available time capacity of resource j for each period. Constraint set (3.16) assures that if disassembled amount of item i on resource j in period t is greater than zero, then corresponding binary variable R_{ijt} takes value 1. The term, $\sum_{t \in T} \sum_{i \in I} D_{it}$, stands for big- M value. It provides that if a non-leaf item is disassembled, the right-hand-side of the constraint becomes non-binding for the number of disassembled amount.

Constraints (3.17) and (3.18) are used to determine that initial and ending inventory level of all items in the planning horizon are set to zero.

Constraint sets (3.19) through (3.25) are the set constraints for corresponding decision variables.

There are differences between the model developed by Lee et al. [8] in Section 3.1 and the model proposed in this section. We call these models as CDSP and CDSP-PR, respectively. While objective function of CDSP is to minimize total cost, CDSP-PR is maximizing net revenue from disassembly operations. Constraint set (3.2) corresponds to (3.12) in CDSP-PR. Inventory constraints for intermediate and leaf items, (3.3) and (3.4), correspond to (3.13) in CDSP-PR.

In the CDSP model, there is demand for only leaf items that have to be satisfied. However, this is not the case in CDSP-PR model. In CDSP-PR model, there is minimum level of demand that should be satisfied. Also, there is demand for not only leaf items but also intermediate items, relation of the number of sold items and its demand is reflected in constraint sets (3.10) and (3.11) in CDSP-PR. In CDSP, there is single resource and disassembly time of items does not depend on period. Available time capacity constraint is stated in (3.5) in CDSP. There are parallel resources and disassembly time of items depends on resource assigned and period in CDSP-PR. Constraint set (3.5) corresponds to (3.15) in CDSP-PR. Moreover, there are resource assignment constraints, (3.14), and initial and ending inventory level constraints, (3.17) and (3.18).

In this chapter, the capacitated disassembly scheduling problem on parallel resource with number of novelties is introduced and explained with proposed MIP model. Developed mathematical model has a number of novelties including parallel resource consideration, assignment of disassembly operations to these parallel resources, different disassembly time of non-leaf items for each resource, demand for not only leaf items but also non-leaf items except root item(s), selling and disposal decision options of an item and aim of the model is to maximize net revenue obtained from disassembly operations.

In the next Chapter, design parameters used in experimentation and testbed generating mechanism are presented.

CHAPTER - 4

COMPUTATIONAL EXPERIMENTATION

The introduced problem and mixed-integer programming model defined in Sections 3.2 and 3.3 has a number of novelties. To the best of our knowledge, proposed mathematical model in Section 3.3 is the first to take into account many of the considerations in capacitated disassembly scheduling such as parallel resource consideration, assignment of disassembly operations to parallel resources, different disassembly time of non-leaf items for each resource, demand for not only leaf items but also non-leaf items except root item(s), selling and disposal decision options of an item.

Comprehensive computational experimentation is essential for the proposed new mixed-integer programming models to determine limits of the model and to learn more about the problem, such as what parameters make the problem harder or to what extent they affect performance of the proposed model. Computational experiments made in the literature so far are based on case studies (disassembly of specific EOL items) or limited number of scenarios in product tree structures as given in Table 2.1. There is a lack and a need for comprehensive benchmark problems set in disassembly scheduling literature that covers all possible product tree structure alternatives.

Scenarios generated in literature so far particularly almost exclusively depend on the different number of items in a disassembly product tree. However, there are other tree-based parameters that affect structure of a product tree such as; the number of root items, the number of levels and the number of children per parent item. For instance, having more number of levels in a product tree complicates the problem be-

cause the number of level makes the product tree divaricated and there will be more decision options such as which item should be disassembled or demand of which item should be satisfied in contrast to decisions in the product tree with less number of levels.

We present a newly developed benchmark problem generation mechanism for the disassembly scheduling problem that includes the following design parameters:

- *Number of root items (EOL product type)*
- *Maximum number of levels in product tree structure*
- *Maximum number of children per item*
- *Probability of part commonality*
- *Gozinto factor*
- *Fill rate*
- *Probability distribution of number of children*

4.1. Design Parameters

In this section, design parameters of the benchmark problem generation mechanism are explained.

4.1.1. Number of root items

The *number of root items* in a product structure represents the number of different product types (different EOL items) to be disassembled. The *number of root items* is a product tree-based parameter. It affects solution performance of the problem because when the number of root items increases, the number of items in a product structure increases. Both single and multiple product types should be considered in experimentation of a solution approach for the disassembly scheduling problem. In this experimentation, both cases are taken into consideration.

4.1.2. Maximum number of levels

This parameter determines *maximum number of levels* to which a product structure

tree can expand. It is a product tree-based parameter. Number of levels in a product structure affects solution performance of the problem. When the maximum number of levels is high, the expected number of items in the product structure increases. It is assumed that the level number of root item(s) is 0; the level number of roots' children is 1 and level number increases when a product structure gets deeper. There is a non-zero probability that a parent item in a structure can have no children. Therefore, number of levels in the product structure does not always have to reach its maximum number of level.

4.1.3. Maximum number of children per item

This parameter determines for each non-leaf item the *maximum number of children* that it can have. It depends on the discrete probability distribution. There is also probability of having no children. Each non-leaf item can have different number of children but this parameter limits the maximum number of children that each non-leaf item can have. This parameter is a product tree-based parameter. Based on this parameter, along with the parameters defined in Sections 4.1.1 and 4.1.2 the number of items in a product structure is determined.

4.1.4. Probability of part commonality

The *probability of commonality* parameter determines whether an item is common or not. The commonality attribute of an item is decided by the given probability level. While each item is being created, it is created as one of the already existing nodes with this probability. With high probability level, there is a higher chance to be common item(s) in a product structure. The probability of commonality is another product tree-based parameter. It affects and changes product tree structure. Also, solution approaches differ greatly based on product structures without/with part commonality. Taleb et al. [3] had to modify their proposed algorithm for part commonality case. Also, Kim et al. [9] proposed solution algorithm for part commonality case for uncapacitated disassembly scheduling. In this experimentation, part commonality is considered for all product structures; single and multiple root items.

4.1.5. Gozinto factor

The parameter of *gozinto factor* determines the number of children items obtained when one unit of a parent is disassembled. It is a problem-based parameter. It does not affect and change product tree structure. However, problem may get more complicated according to level of gozinto factor. For instance, the number of required disassembled items to satisfy a specific demand may decrease with high level of gozinto factor in contrast to low level of gozinto factor. This is because when one unit of parent item is disassembled, more number of children is obtained. It may also affect the inevitable inventory level of bi-items.

4.1.6. Fill rate

Fill rate parameter determines the minimum percentage level (service level) of demand that must be satisfied so that penalty cost does not incur. It is another problem-based parameter. With high level of fill rate, demand of items that must be satisfied increases. With low level of fill rate, demand of items that must be satisfied decreases. Based on level of fill rate, the number of disassembled non-leaf items, the number of sold items and other decisions may be affected.

4.1.7. Probability Distribution of number of children

Probability distribution of number of children parameter is a discrete probability distribution that gives the probability of having different number (up to the level of maximum number of children) of children or zero children. This parameter is affined with the parameter of the maximum number of children per item. The latter determines the maximum number of children an item can have. The first gives the probability levels of number of children up to the latter parameter. Children probability distribution parameter is another product tree-based parameter. Number of children of each non-leaf item is affected according to the probability distribution of children.

4.2. Test-bed Problem Generating Mechanism and Settings

In this section, detailed steps of generation mechanism of test-bed problem instances and required data set for generation are explained. To generate test-bed problem instances, level of product structure tree-based parameters; number of root items, maximum number of levels, maximum number of children per item, probability of part commonality, gozinto factor, fill rate and children probability distribution must be given as an input.

Also, level or distribution of cost-based and other parameters given in the mathematical model; number of resources, planning horizon, selling price of items for each period, penalty cost of unsatisfied one unit of items for each period, demand distribution of items for each period, cost of holding items at the inventory for each period, disposal cost of items for each period, disassembly time and cost of items for each resource and period, available resource capacity of each resource for each period, purchasing cost of EOL items for each period (if purchasing is not considered as negligible) must be given as an input.

Generation scheme of product structure tree and numbering items in the structure is depth-first approach. Steps of problem instances generation is given below:

1. Root (EOL) items are generated up to the *number of root items*.
2. Generate a random variate to determine a number of children for each parent item based on given *probability distribution of number of children* and level number of parent item.
3. Children are marked to be common or not, based on given *probability of commonality*. Items marked as common are put into common item list.
4. Generate a random variate to determine *gozinto factor* of each item, based on the given probability distribution for levels of *gozinto factor* in Table 4.4.
5. If level number is equal to *maximum number of levels* parameter and num-

ber of root items is equal to *number of root items* parameter, then go to Step 6. Otherwise, go to Step 2.

6. For items marked as common, to find another item that will be common with an item in the common list, steps defined below are carried out for each common item:

6.1 Determine a target root. Target root is an item at level zero and one of its children is copied to marked item in the common list. When there are two or more root items, the selection of target root is determined by generating random number and comparing it with the given discrete probability distribution of target root. Probability distributions of target root used in this experimentation are given in Table 4.1.

Design Level of Number of Root Items	Target root number		
	1	2	3
Low	1		
Medium	0.5	0.5	
High	0.3	0.3	0.4

Table 4-1 Target root probability values

6.2 Determine a target level. Target level shows the level number of an item where marked item is copied to one of the item at that level. There is no chance for an item to be copied with root item(s). In the target level phase, each level does not have equal probabilities to be selected. Based on given inputs, one can determine in which levels there are high probability for items to be common. Probability distributions of target level used in this experimentation are given in Table 4.2.

Design Level of Max. Number of Level	Target level number			
	1	2	3	4
Low	0.2	0.8		
Medium	0.1	0.4	0.5	
High	0.1	0.2	0.3	0.4

Table 4-2 Target level probability values

6.3 Determine an item that is copied to item in the common list. First item that fits target root and target level is selected to be common. Search strategy for this phase is breadth-first approach.

7. Demand, time-based (such as disassembly time) and cost-based (disassembly cost) parameters are generated for each part by considering whether it is leaf item or not. Since leaf items cannot be disassembled further, their disassembly time is zero.

After generation steps explained above, problem instance sets are generated automatically up to given number of instance for each problem set. This generation mechanism is essential for disassembly scheduling literature to generate benchmark problems. It contains all possible product tree-based design, cost and other parameters. One can easily generate problem sets with given input data.

In experimentation we set, three different design levels are determined for all design parameter values as low, medium and high. Level 1 indicates the low level and Level 3 shows high level. Level 2 represents the medium level. Values of all design parameters are given in Table 4.3.

		Design Levels			
		No	Low(1)	Medium(2)	High(3)
Design Parameters	# of root items	1	1	2	3
	max. # of level	2	2	3	4
	max. # of children/item	3	2	3	4
	P(Commonality)	4	0.01	0.03	0.05
	Gozinto Factor	5	2.8	2.9	3.04
	Fill Rate	6	0.9	0.95	0.99
	Probability Distribution of number of children	7	1.83	2.05	2.45

Table 4-3 Value of design parameter values

In Table 4.3, levels of gozinto factor and children probability distribution are given as expected values. Gozinto factor and children probability discrete distribution values for each level are given in Tables 4.4, 4.5, 4.6 and 4.7.

Design Level of Gozinto Factor	Gozinto factor values			
	1	2	3	4
Low	0.1	0.3	0.3	0.3
Medium	0.1	0.2	0.4	0.3
High	0.2	0.1	0.2	0.5

Table 4-4 Probability values of Gozinto factor

In Table 4.4, rows of the table shows levels of the gozinto factor parameter and columns indicate the gozinto factor values. Probability values are given for each corresponding level and gozinto factor value. For instance, probability value of gozinto factor 2 at low level is 0.3. It means that when one unit of a parent of an item is disassembled, two units of child are obtained with probability 0.3.

Design Level of Max. # of levels	Max. # of children per item				
	0	1	2	3	4
Low	0.1	0.3	0.6		
Medium	0.1	0.1	0.6	0.2	
High	0.1	0.3	0.3	0.2	0.1

Table 4-5 Probability values of probability distribution of number of children for level 1

Design Level of Max. # of levels	Max. # of children per item				
	0	1	2	3	4
Low	0.05	0.15	0.8		
Medium	0.1	0.1	0.4	0.4	
High	0.1	0.2	0.2	0.3	0.2

Table 4-6 Probability values of probability distribution of number of children for level 2

Design Level of Max. # of levels	Max. # of children per item				
	0	1	2	3	4
Low	0	0.05	0.95		
Medium	0.05	0.05	0.3	0.6	
High	0.05	0.05	0.2	0.3	0.4

Table 4-7 Probability values of probability distribution of number of children for level 3

Probability values of probability distribution of number of children are shown in Tables 4.5, 4.6 and 4.7. Probability distribution of number of children depends on the parameters of maximum number of children per item and maximum number of levels in the product structure. For each design level of maximum number of levels parameter, there are corresponding probability values for each number of children. For example, probability of having one child is 0.3 for low level of maximum number of children and maximum number of levels in Table 4.5. Low level for maximum num-

ber of levels is 2. Tables 4.6 and 4.7 represent medium and high levels of probability values of probability distribution of number of children, respectively.

There are other parameters that have to be determined; demand distribution, disassembly time of non-leaf items for each resource, disassembly cost of non-leaf items for each resource, holding cost of items, selling price of items, disposal cost of items and penalty cost of items.

There is demand for each item except root item at each period. With probability 0.1, there is no demand for an item in a period. With probability 0.9, demand is generated from discrete uniform distribution between 50 and 200 for each period; $DU\sim[50,200]$ where $DU\sim[a,b]$ is the discrete uniform distribution with range $[a,b]$. We set demand distribution as explained above like in the studies of Kim et al. [9] and [14].

For each non-leaf items, disassembly time and disassembly cost are generated for each resource at each period as in [12]. Disassembly time is generated from discrete uniform distribution between 1 and 4; $DU\sim[1,4]$. Disassembly cost is generated from discrete uniform distribution between 50 and 100; $DU\sim[50,100]$. Holding cost is generated between 5 and 10 for each item at each period; $DU\sim[5,10]$, where lower and upper levels are 10% that of disassembly time discrete uniform distribution.

Selling price and disposal cost are generated from the relationships between disassembly cost and holding cost. There are different disassembly costs for each non-leaf item depending on resource used. However, selling price of an item does not depend on the resources. Average disassembly cost (AVGC) of an item is determined by taking the average of its disassembly cost over all resources. Selling price of an item is set to one and a half times its average disassembly cost as given in equation (4.1) and selling price of each item is constant for each period.

$$SP_i = 1.5 * AVGC_i \quad \forall i \in I \quad (4.1)$$

There is a trade-off between disposing of an item and holding that item at the inventory. Disposal cost of an item is determined based on its holding cost. It is set to be cheaper than holding cost. The reason is that when an item is disposed, it cannot be used in future periods and to obtain the item for next periods, its parent(s) have to be

disassembled. This disassembly operation causes extra costs. Disposal cost of each item is constant for each period. It is set to half of its holding cost per period as given in equation (4.2).

$$IC_i = 0.5 * H_i \quad \forall i \in I \quad (4.2)$$

If level of satisfied demand for each item falls short of the fill rate, penalty cost incurs for each unsatisfied unit of demand. Penalty cost of an item is constant for each period. It is set to one and a half times its selling price as stated in equation (4.3).

$$PN_i = 1.5 * SP_i \quad \forall i \in I \quad (4.3)$$

Available time capacity of each resource is determined based on fill rate, demand and average disassembly time of all items. Required time to obtain one unit of children by disassembling its parent is called as “expected disassembly time”. Available capacity calculation for each resource and each period is given in equation (4.4).

$$CAP_{jt} = \frac{\sum_{i \in I} \sum_{t \in T} \beta_{it} D_{it} (E[DT_i])}{T * J} \quad j \in J, t \in T \quad (4.4)$$

where

CAP_{jt} : Expected time required to disassemble enough items so that fill rate constraints are satisfied for resource j in period t

β_{it} : fillrate of item i in period t

$E[DT_i]$: average disassembly time of item i

T : number of periods

J : number of resources

In equation (4.4), we set available time capacity of each resource for each period to time that is enough for disassembly operations for each period and resource to satisfy fill rate demand on the average.

Planning horizon is set to 10 periods. There are total 2187 ($=3^7$) different problem sets as the combination of design parameters and their levels, given in Table 4.3. For each problem setting, five instances are generated. Testbed generation is coded in Microsoft Visual Studio 2008 C++ language version. Execution time limit is set to

3600 seconds for each instance. Instances are solved with CPLEX 10 solver in GAMS 22.5 using Intel® Xeon® CPU, X5482 3.20 GHz (2 processors) computer with 10 GB RAM.

4.3. Computational Results

To evaluate the performance of the proposed mathematical model, *average execution time in seconds*, **O1**, *average percentage of optimal solution found in each problem set (% Success)*, **O2** are reported.

Among all instances, average node size of problem for which MIP can solve within the time limit is 24.3. The average execution time limit of these problems is 1029.67 sec. Average percentage success and resource utilization is 73.65% and 85.10%, respectively. Table 4.8 summarizes results of experimentation set grouped by success percentage.

According to Table 4.8, on the average 35% of the problem set can be solved with 100% success within less than one and a half minute. The average number of nodes in product tree of these sets is below 15. It can be seen from Table 4.8 that success rate decreases, as the number of nodes increases. When the average number of nodes is 48, only very few of the instances can be solved optimally within time limit. In the experiment set, maximum node size that MIP can handle is 116 with 4 levels. When the average number of items is between 23 and 35, instances can be solved between 40% and 80% success. Clearly, there are other factors that strongly affect solvability (or difficulty) of the problem set. Starting from next subsection, we examine effects of other factor to the problem.

% Success	% of the Total Sets	Average Execution Time (sec.)*	Average Node Size	Min. Node Size	Max. Node Size	Average Utilization(%)
100	34.8	72.21	14.29	4	31	83
80	9.9	784.07	22.93	10	61	91
60	7.2	1523.99	27.57	14	46	90
40	7.4	2252.63	35	18	80	89
20	12.4	2987.13	48.24	21	118	92
0	28.3	-	95.46	27	605	-

Table 4-8 Summary Results of Experimentation Set (*If MIP cannot solve within time limit, solution time is taken as 3600 sec. for average calculation purposes)

4.3.1. Analysis of Effects of Design Parameters

One would ideally like to find out the relationships between design parameters and problem difficulty. With the results from an experimental design like this, ANOVA (Analysis of Variance) is generally used to measure effect of design parameters to the solution performance. To be able to conduct ANOVA, three basic assumptions have to be simultaneously satisfied [19]. We use average execution time (O2) to test validity of ANOVA assumptions because O1 is continuous whereas O2 is discrete data. Assumptions of ANOVA are given below:

- Independence of observations
- Normality
 - The distribution of residuals are normal
- Homoscedasticity
 - The variance of data in groups should be the same

To test the validity of normality assumption, we applied Kolmogorov-Smirnov test, using MINITAB 16. Unfortunately, our results do not satisfy normality assumption. Normality plot and Histogram of average execution time are provided in Figures 4.1 and 4.2.

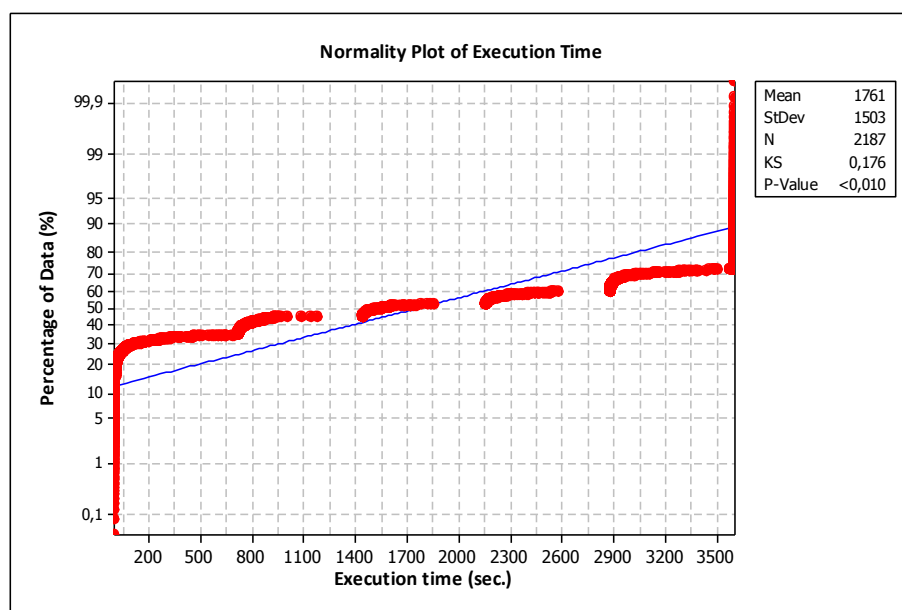


Figure 4.1 Normality plot of execution time

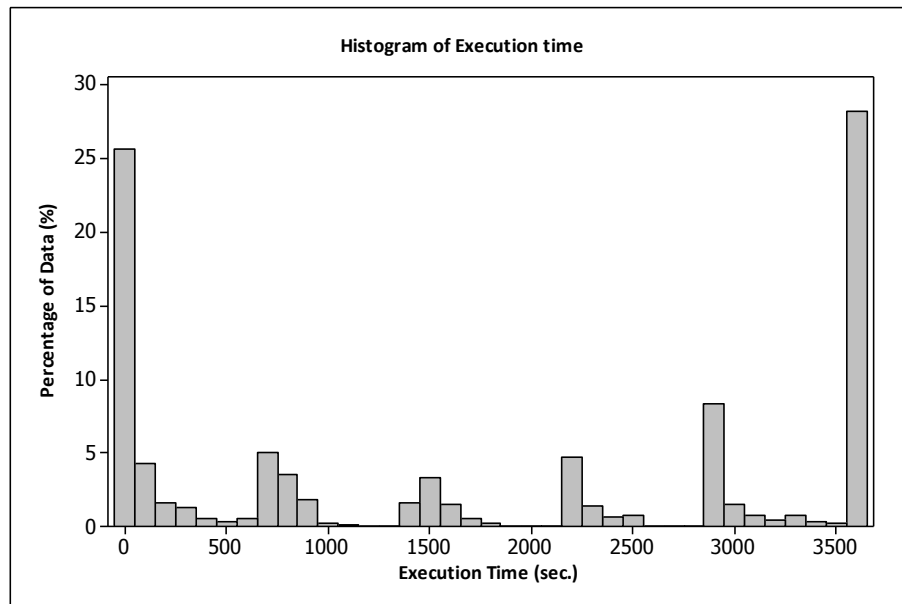


Figure 4.2 Histogram of execution time

It is also worth to note that normality of data is tested by converting data to logarithmic base. In the form of logarithmic base, data does not satisfy normality assumptions. Because we cannot use ANOVA, to find out the effects of different levels of design parameters to performance in terms of average execution time in seconds and average percentage success, main effect analyses are conducted. In main effect analysis figures, black point, red point and green point indicate mean values of first design level, second design level and third design level of parameters, respectively. Bars on point of each design level show lower and upper bounds of 95% confidence interval for the mean values. Detailed information on confidence interval values is given on Table 4.9. Based on Central Limit Theorem (CLT) [19], regardless of the original distribution of the data, the sample mean of n independent observations will be approximately normally distributed as n goes to infinity (especially for $n \geq 50$). In our experimentation, there is a total of 10935 ($=3^7 \times 5$) instances. Also, there are more than 100 problem sets for each design level of each parameter. Subsections 4.3.1.1 – 4.3.1.7 present results of this analysis.

4.3.1.1. Effect of the Number of Root Items

The number of root items in a product structure affects solution quality, in terms of O1 and O2, negatively. An increase in number of root items directly affects number of nodes and increases problem size. Figures 4.3 and 4.4 show the effects of the

number of root items to execution time and success percentage rate, respectively. As it can be seen from Figures 4.3 and 4.4, number of root items directly affects the solution quality. When there is an increase in level of number of root item, execution almost doubles and success percentage rate almost decreases by 50%.

4.3.1.2. Effect of the Maximum Number of Levels

Number of levels in a product structure tree determines depth of the Bill-of-Material (BOM) tree. Therefore, it is expected that increasing number of levels makes the problem more complicated. Table 4.5 and 4.6 indicate the effect of maximum number of levels to the execution time and success percentage, respectively.

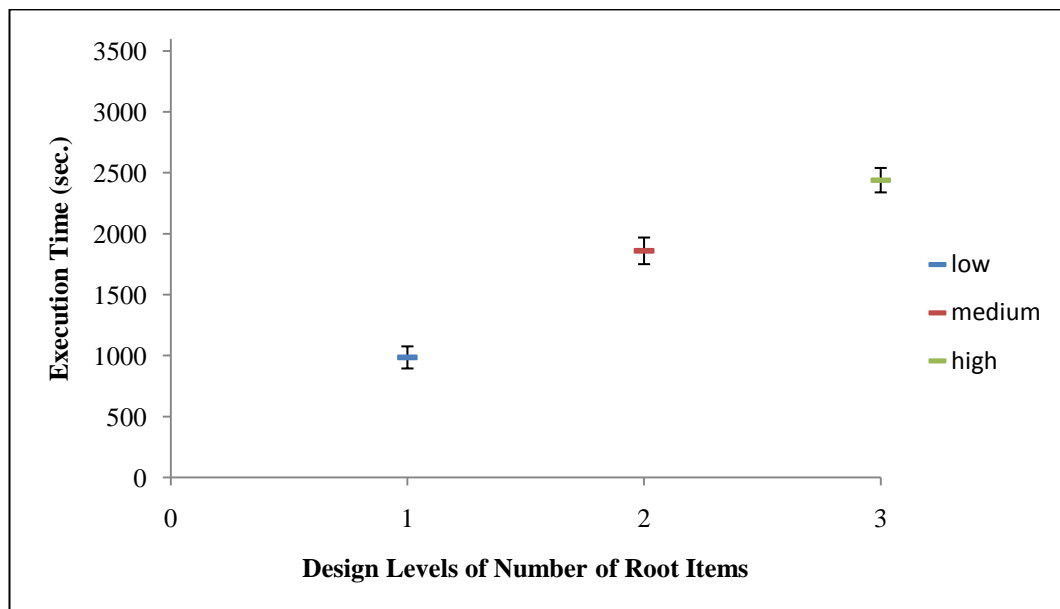


Figure 4.3 Effect of the number of root items levels to execution time

Based on Figures 4.5 and 4.6, increasing number of levels from level one to level two directly increases execution time by more than 1500 seconds and decreases success percentage by almost 40%. However, effect of increasing number of levels from level two to level three is less than effect of increasing number of levels from level one to level two.

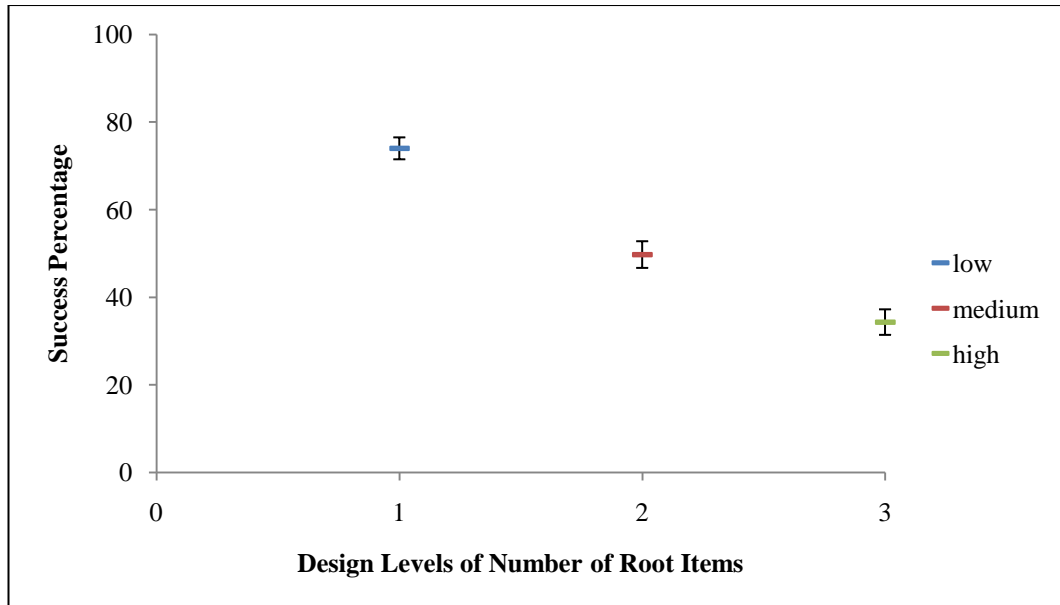


Figure 4.4 Effect of the number of root items levels to success percentage

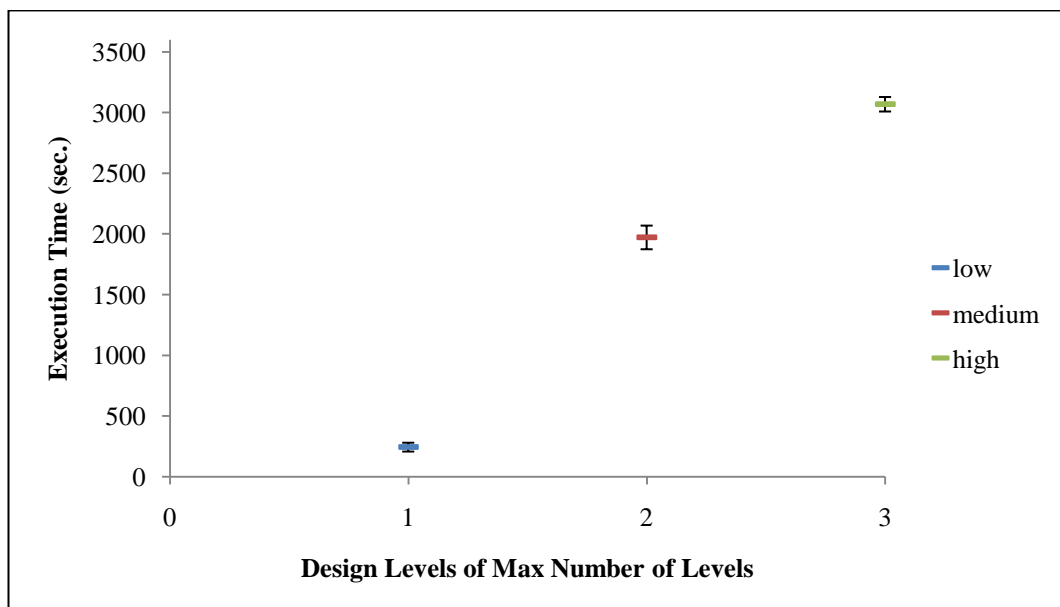


Figure 4.5 Effect of the maximum number of levels to execution time

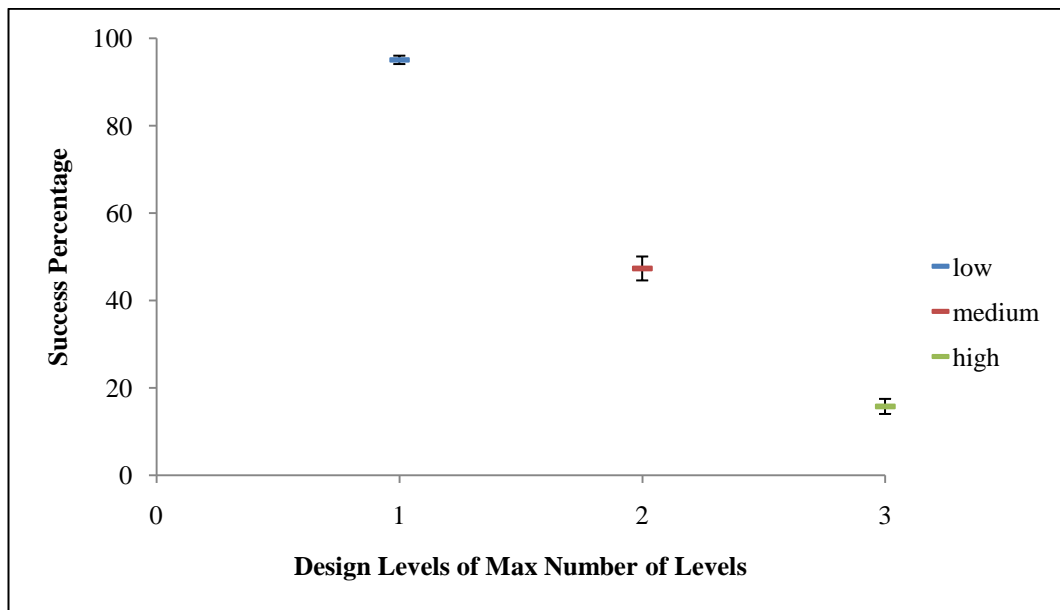


Figure 4.6 Effect of the maximum number of levels to success percentage

4.3.1.3. Effect of the Maximum Number of Children per Item

Execution time and success percentage change very little with varying levels of maximum number of children per item. Effect of the maximum number of children per item is shown in Figures 4.7 and 4.8, respectively.

It can be said that the maximum number of children per item parameter affects execution time and success percentage according to Figures 4.7 and 4.8. However, effect of the parameter does not seem to be as strong as the number of root items and the maximum number of levels.

4.3.1.4. Effect of Probability of Commonality

When there are common items in a product structure, problem may get difficult because there will be alternative ways to obtain same item with varying amounts. However, it does not seem to be as strong as the number of root items and the maximum number of levels. Its effects to performance measures are shown in Figures 4.9 and 4.10. Based on Figures 4.9 and 4.10, increasing or decreasing the level of probability

of commonality has nearly no effect on the performance measures.

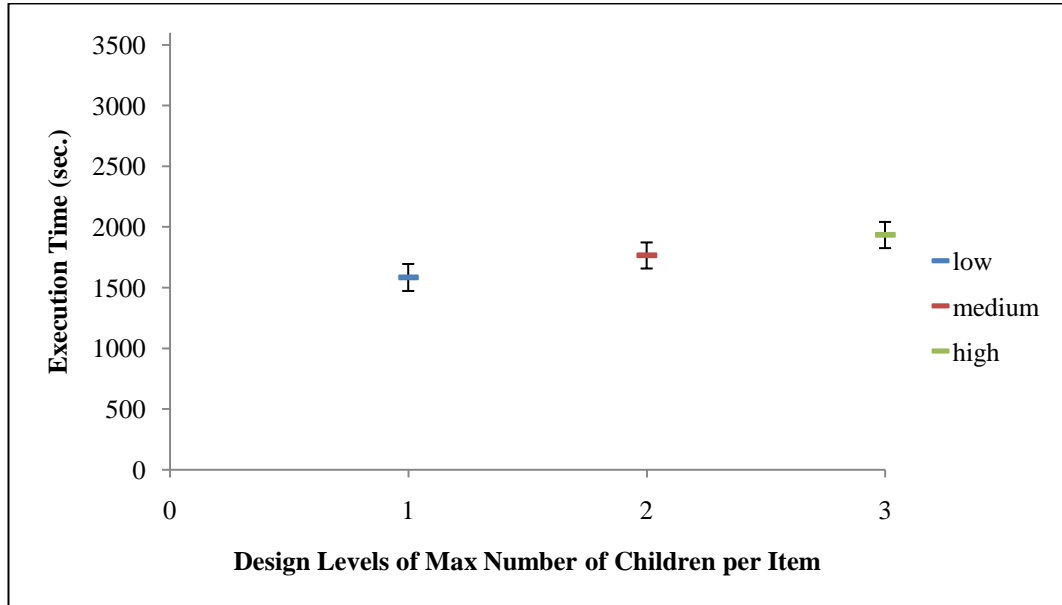


Figure 4.7 Effect of the maximum number of children per item to execution time

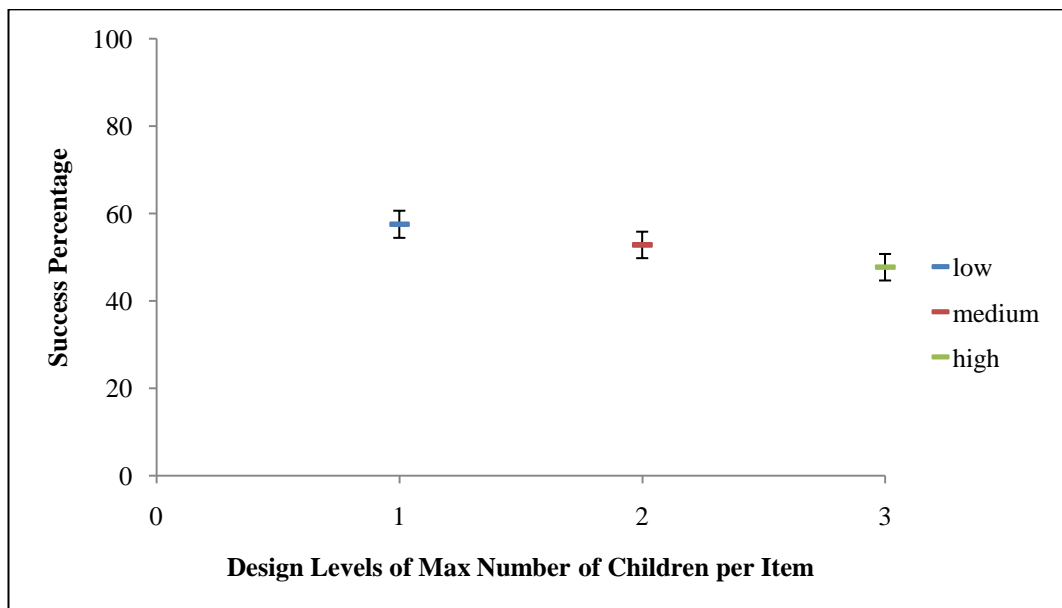


Figure 4.8 Effect of the maximum number of children per item to success percentage

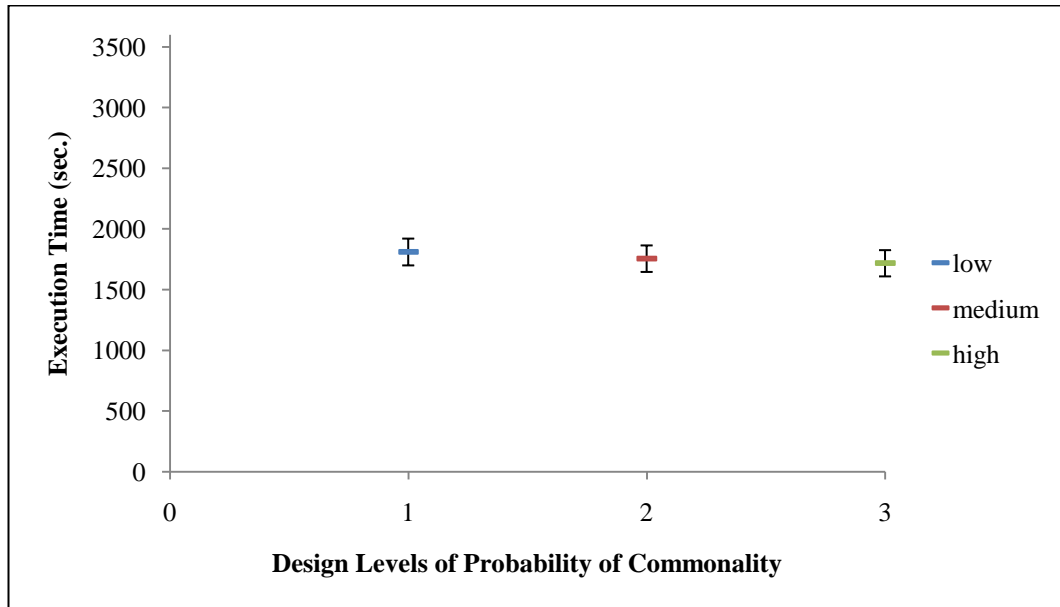


Figure 4.9 Effect of the probability of commonality to execution time

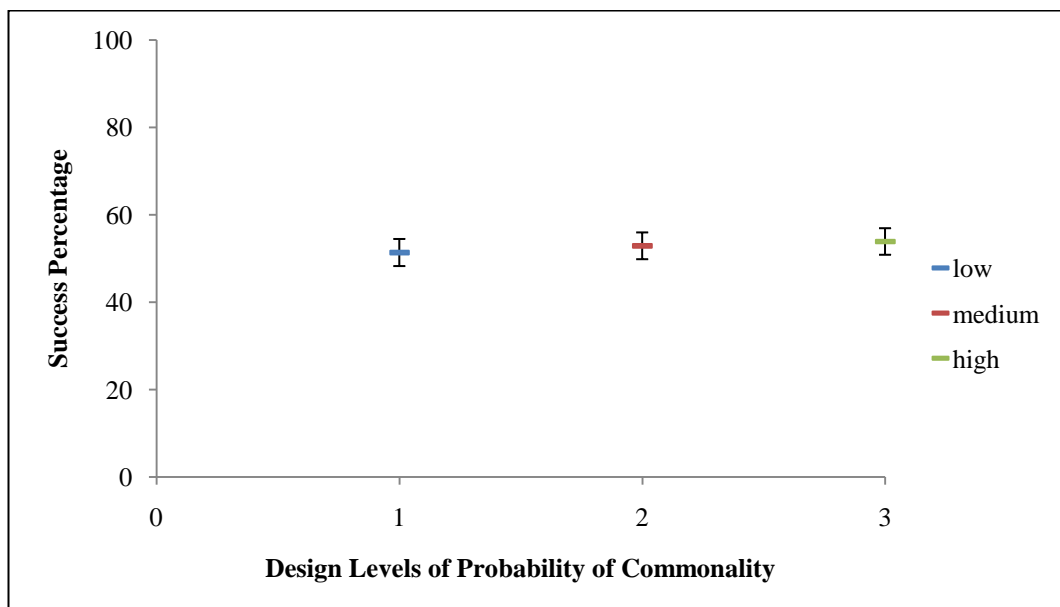


Figure 4.10 Effect of the probability of commonality to success percentage

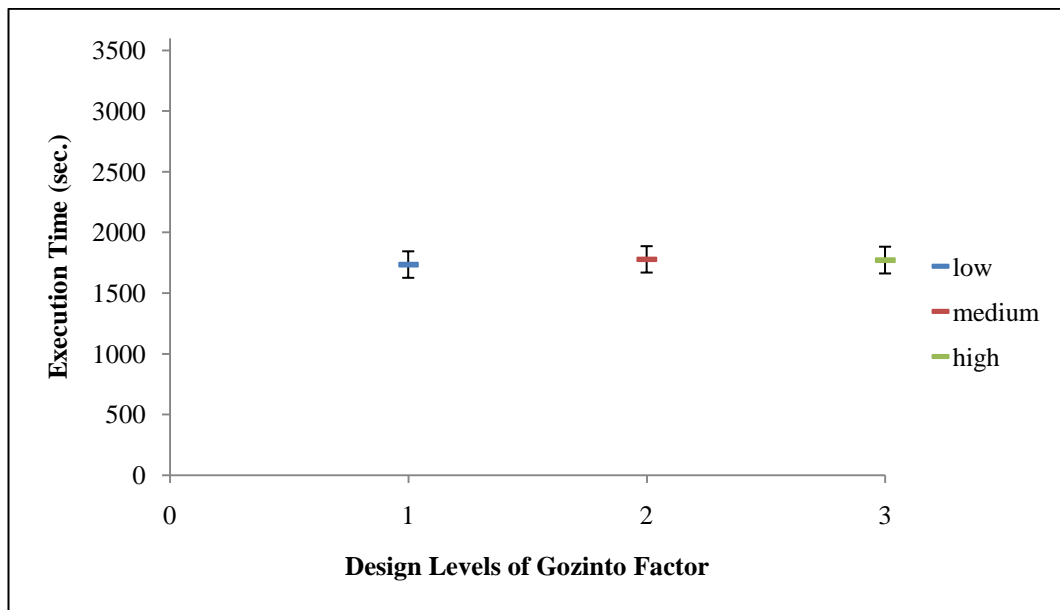


Figure 4.11 Effect of the gozinto factor to execution time

4.3.1.5. Effect of Gozinto Factor

Effects of different levels of gozinto factor may vary on the solution quality in terms of execution time and success percentage. Low level of gozinto factor may cause the more disassembly. On the other hand, high level of gozinto factor may cause more excess inventories and less disassembly. These situations can affect the problem positively or negatively depending on instances. Therefore, it seems to be stated that parameter of gozinto factor has no observable direct effects on the performance measures. Figures 4.11 and 4.12 show the effect of gozinto factor.

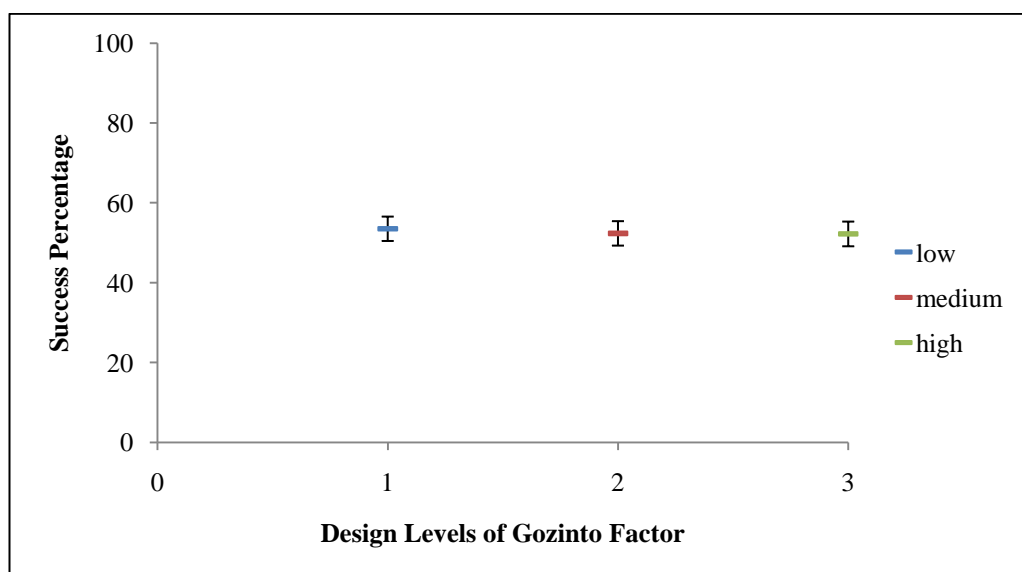


Figure 4.12 Effect of the gozinto factor to success percentage

4.3.1.6. Effect of Fill Rate

Figures 4.13 and 4.14 suggest that, much like the case of gozinto factor parameter, fill rate parameter also does not seem to have much effect on execution time and success percentage.

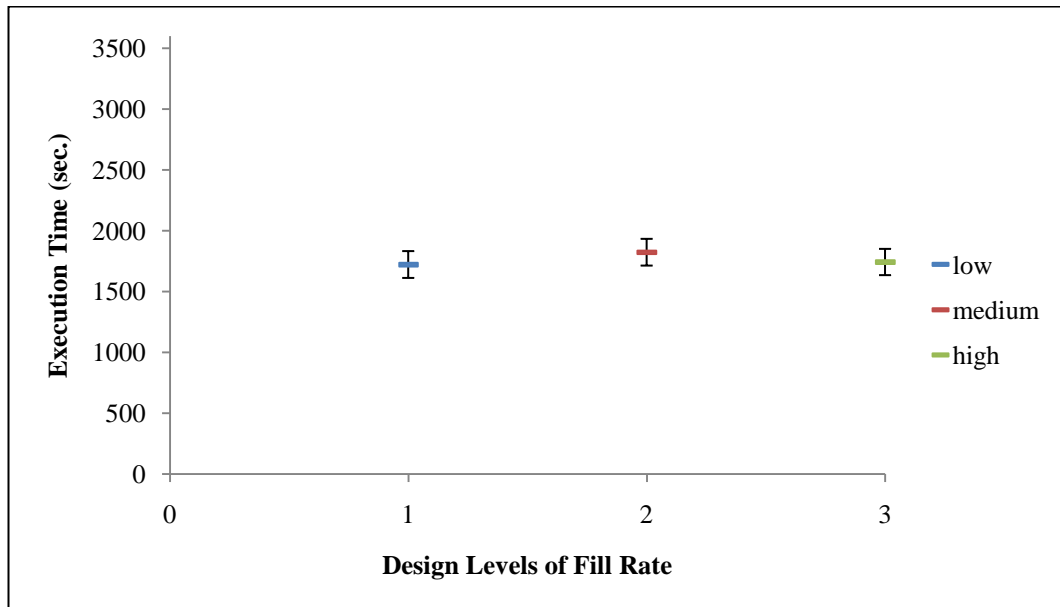


Figure 4.13 Effect of the fill rate to execution time

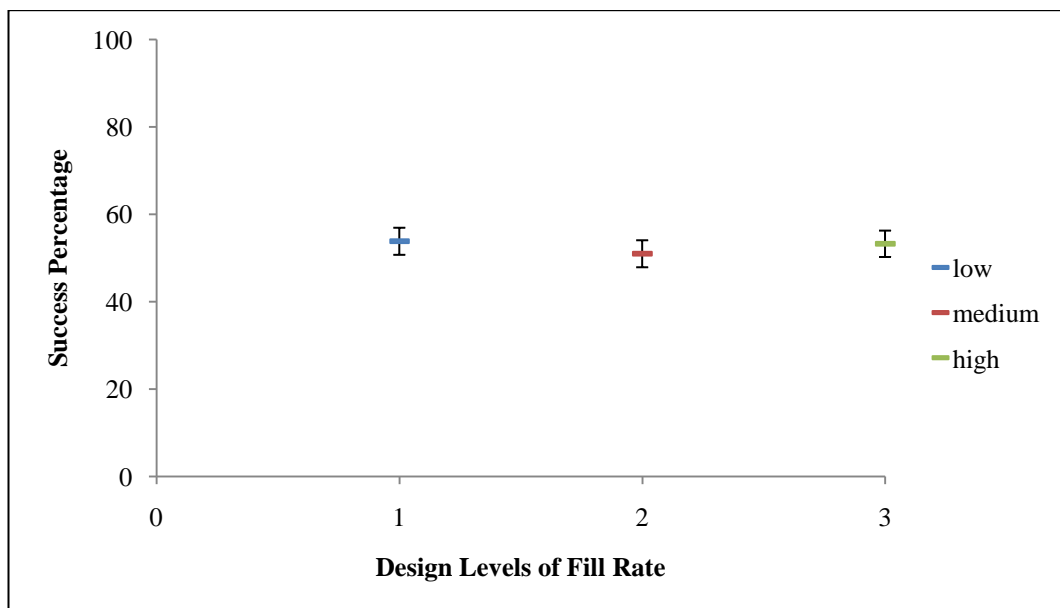


Figure 4.14 Effect of the fill rate to success percentage

4.3.1.7. Effect of Probability Distribution of number of children

The relevance of parameters of probability distribution of number of children and maximum number of children per item can be recognized from Figures 4.15 and 4.16.

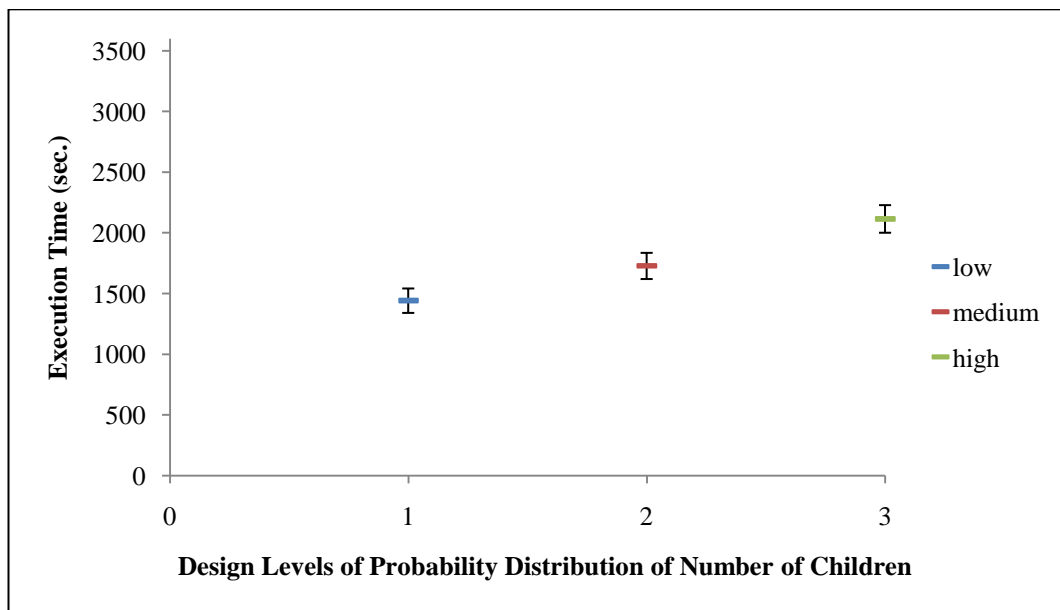


Figure 4.15 Effect of the probability distribution of number of children to execution time

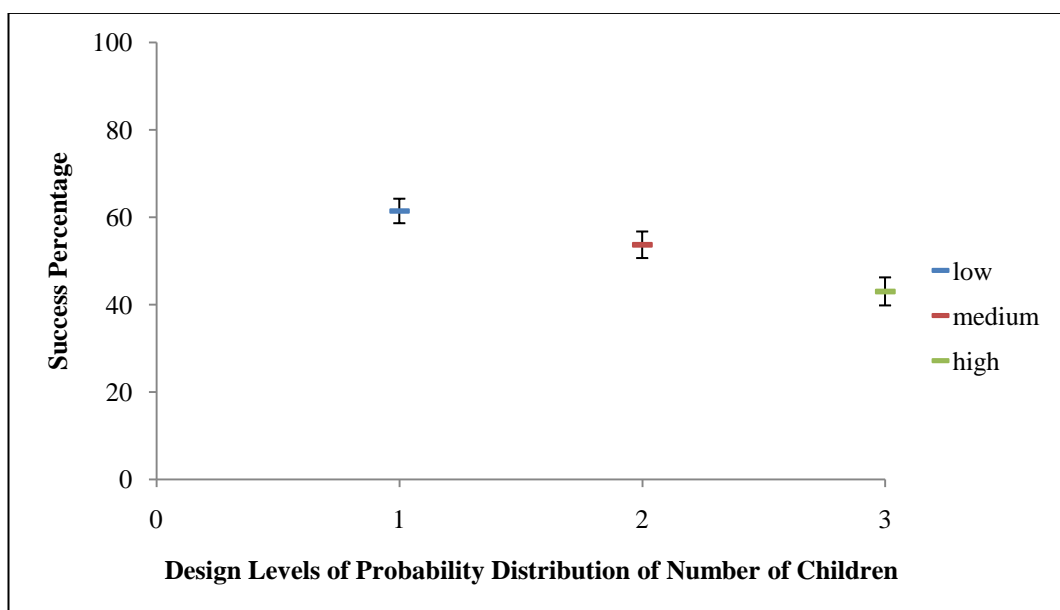


Figure 4.16 Effect of the probability distribution of number of children to success percentage

Probability distribution of number of children parameter determines the number of children of a parent item and has impact on the number of items in a product tree structure. Probability distribution of number of children affects the solution quality in terms of execution time and success percentage.

Parameter of probability distribution of number of children has more impact than maximum number of children per item parameter on the solution quality. This is also expected because maximum number of children per item determines only the maximum number of children that an item can have, whereas probability distribution of number of children determines the number of children of a parent item.

Figures (4.3) – (4.16) show that probability of commonality, gozinto factor and fill rate seem to have almost no effect on execution time and success percentage. Maximum number of children per item parameter also seems to have little effect on performance measures. It can also be noted that common trait of parameters which has direct impacts on the solution quality is that they actually determine the total number of items in the product structure tree.

In Table 4.9, mean execution time and (95%) confidence interval values of each level of design parameter calculated in main effect analysis are given. In Table 4.9, design parameter numbers are used as in Table 4.3. For example, at low design level of parameter 1, its mean is between $984.98 - 90,61 (=894,37)$ and $984.98 + 90,61 (=1075,59)$ with 95% confidence level. From Table 4.9, it is seen that none of design parameters pushes the execution time limit with 95% confidence level. However, there is significant difference between confidence interval values of medium and high design levels for number of root items and maximum number of levels parameters. At each of these two parameters, confidence interval gets narrower at high design values. This is because; number of root items and maximum number of levels parameters are most dominant over other parameters. Problem gets more complicated and execution time increases as design level of these parameters increases. For example, at high design values of number of root items and maximum number of level parameters, 337 problems out of 729 problem sets (=46% of total sets) and 439 problems out of 729 problem sets (=60% of total sets) push execution time limit. Howev-

er, these ratio falls to 27% and 26% for high design values of probability of commonality and fill rate parameters, respectively. As problem gets more complicated, execution time converges to the limit. Therefore, at high design level of number of root items and maximum number of level parameters, confidence level values are significantly less than values at medium design level.

Design Levels		Design Parameters No						
		1	2	3	4	5	6	7
Low Design Level	Mean	984.98	243.84	1584.23	1810.61	1734.74	1720.79	1441.47
	Confidence Level (95%)	±90,61	±36,36	±111,19	±110,44	±109,07	±110,26	±100,66
Medium Design Level	Mean	1859.54	1971.50	1765.68	1755.60	1777.61	1821.88	1727.69
	Confidence Level (95%)	±109,23	±97,49	±107,39	±109,31	±108,68	±109,80	±107,76
High Design Level	Mean	2439.51	3068.70	1934.13	1717.82	1771.68	1741.37	2114.87
	Confidence Level (95%)	±100,28	±59,77	±107,96	±108,19	±110,27	±107,84	±113,67

Table 4-9 – 95% Confidence Level and Mean Values of Design Parameters in Execution time (sec.) for Main Effect Analysis

Moreover, we investigate the interaction between parameters that have effects on execution time and success percentage. From the results of main effect analysis, it seems that parameters; number of root items, maximum number of levels, maximum number of children per item and probability distribution of number of children, affect performance measures; average execution time and average percentage success. Also, these parameters called as “dominant parameters”. Interaction analysis is made for these parameters. The purpose of interaction analysis between these parameters is to observe which effective parameter is dominant on which parameters to what extent. In all figures in interaction analysis; black point, red point and green point indicate mean values of first design level, second design level and third design level of parameters, respectively. In this analysis, bars on mean value of each design level show lower and upper mean values with 95% confidence interval. Values of confidence interval are presented on Table 4.10. CLT states that regardless of the original distribution of the data, the sample mean of n independent observations will be ap-

proximately normally distributed as n goes to infinity (especially for $n \geq 50$) [19]. There are more than 100 problem sets for each design level of parameter pair.

4.3.1.8. Interaction between Number of Root Items and Maximum Number of Levels

As it can be seen from Figures 4.17 and 4.18, parameters of maximum number of levels and number of root items seem to have strong effects on execution time and success percentage. At each level of number of root items, as level of maximum number of levels increases, execution time shows increase and success percentage shows decrease strongly. Effect of maximum number of levels decreases as level number of the parameter increases.

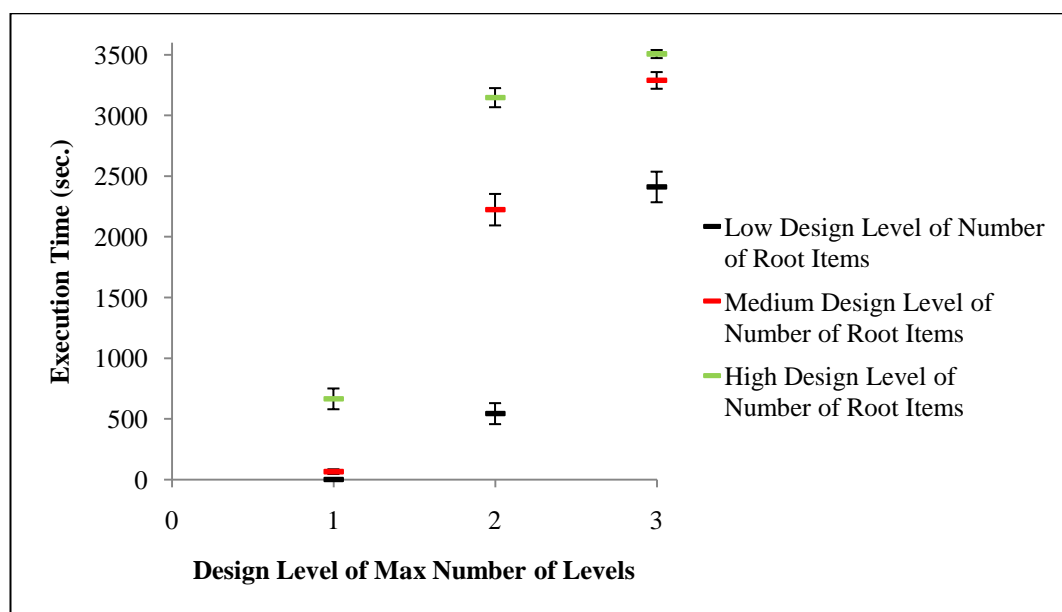


Figure 4.17 Interaction between number of root items and maximum number of levels in execution time

This is because many instance at high level number of root items and maximum number of levels push execution time limit. Also, as level numbers of parameters increases, it is seen that problem gets more difficult.

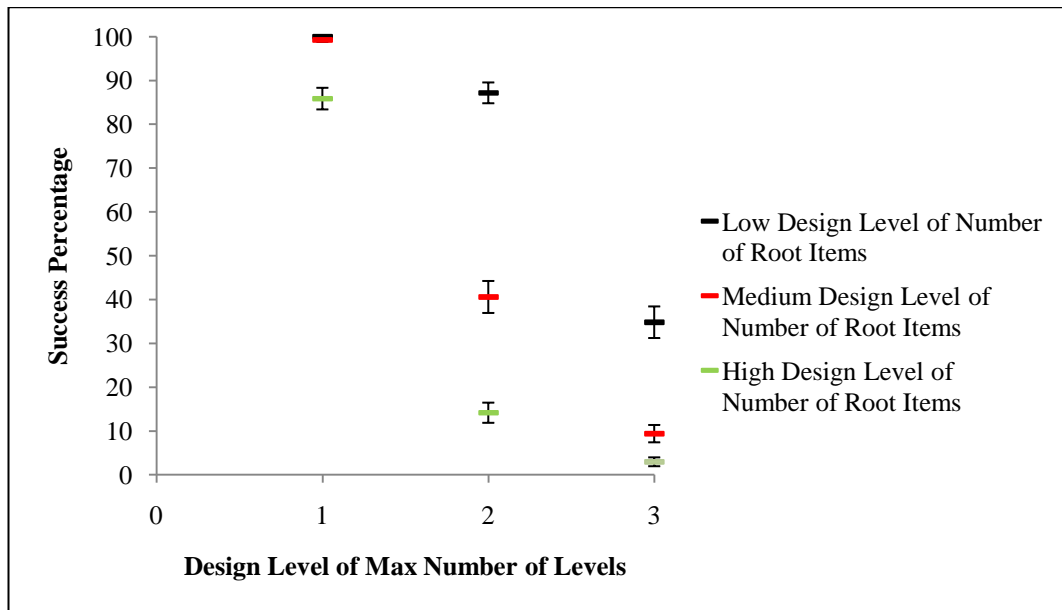


Figure 4.18 Interaction between number of root items and maximum number of levels in success percentage

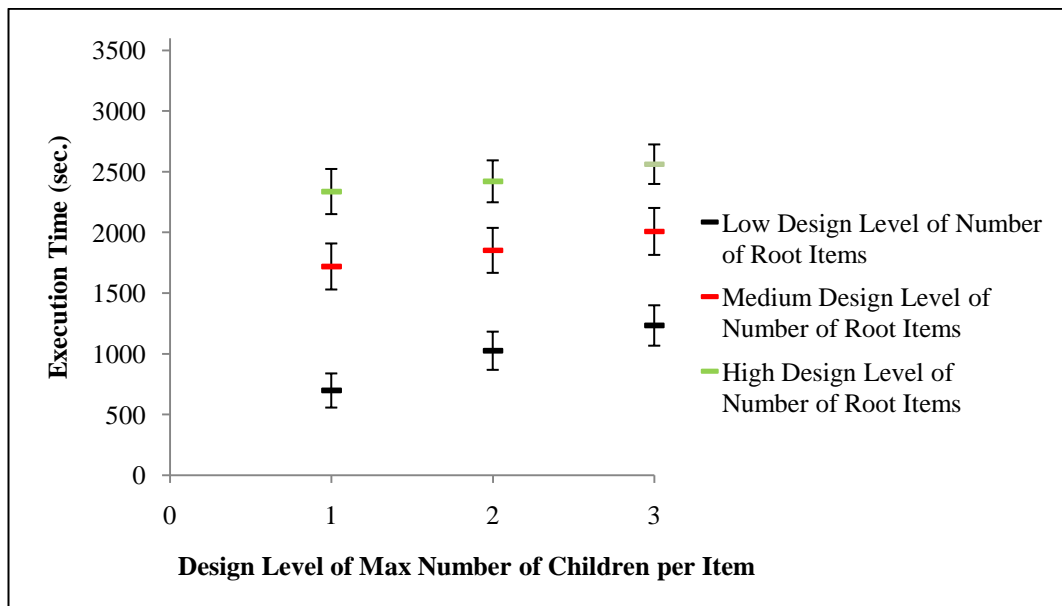


Figure 4.19 Interaction between number of root items and maximum number of children per item in execution time

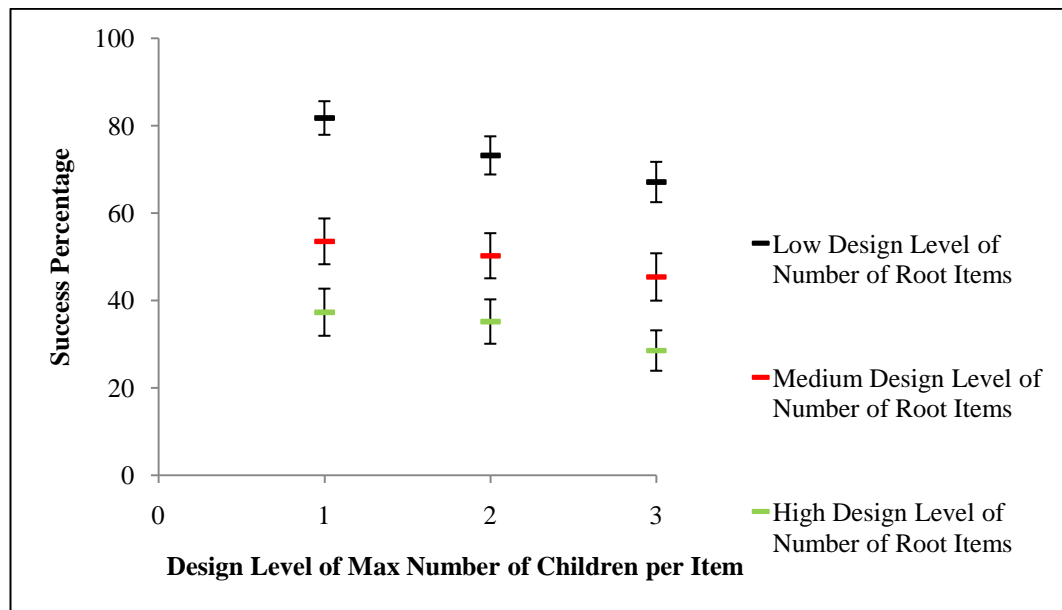


Figure 4.20 Interaction between number of root items and maximum number children per item in success percentage

4.3.1.9. Interaction between Number of Root Items and Maximum Number of Children per Item

Figures 4.19 and 4.20 indicate that maximum number of children per item parameter and number of root items parameter seems to have effect on execution time and success percentage. From these figures, it can be said that the number of root items parameter has more impact on solution performance than maximum number of children per item parameter. For example, at each design level of maximum number of children per item, effect of increasing number of root items by one design level has much effect in contrast to effect of increasing maximum number of children per item by one design level.

4.3.1.10. Interaction between Number of Root Items and Probability Distribution of number of children

Figures 4.21 and 4.22 indicate that probability distribution of number of children seems to have less effect than number of root items for execution time and success

percentage than maximum number of levels. At each level of number of root item parameter, execution time tends to increase whereas success percentage tends to decrease as level of probability distribution of number of children increase. However, amount of increase is not as much as maximum number of levels parameter.

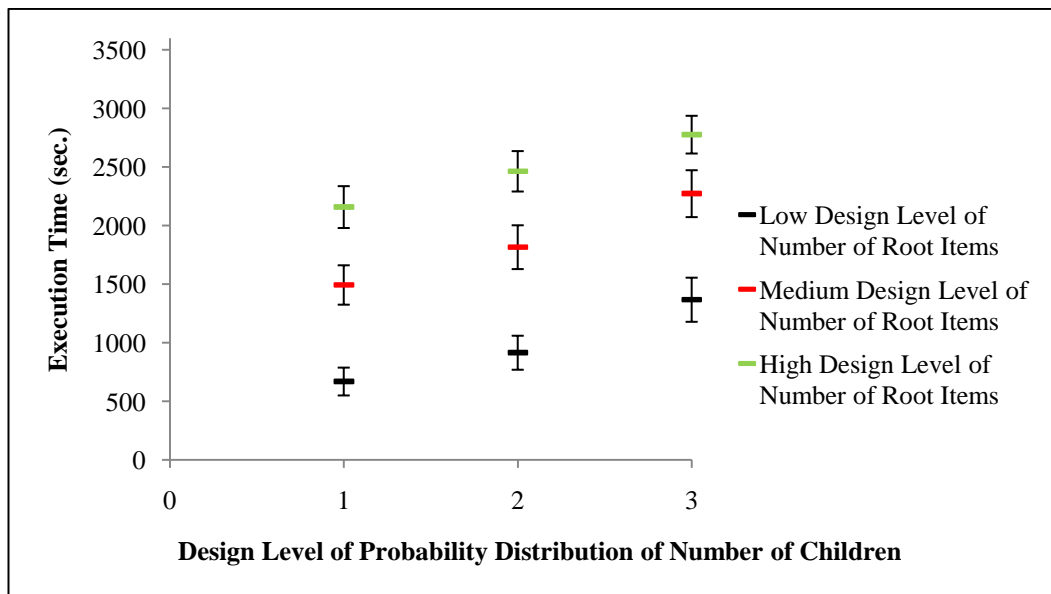


Figure 4.21 Interaction between number of root items and probability distribution of number of children in execution time

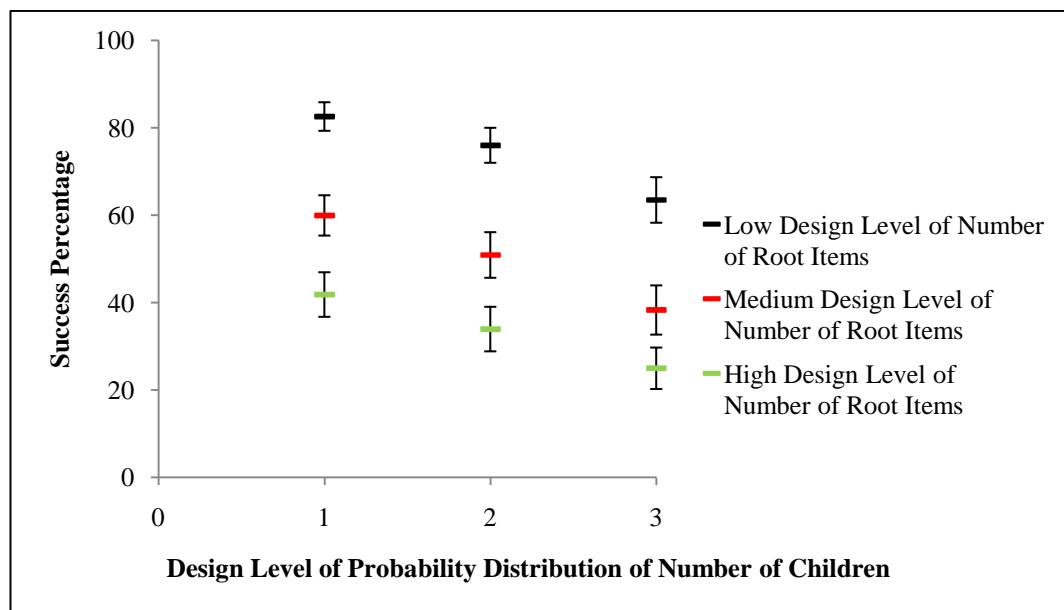


Figure 4.22 Interaction between number of root items and probability distribution of number of children in success percentage

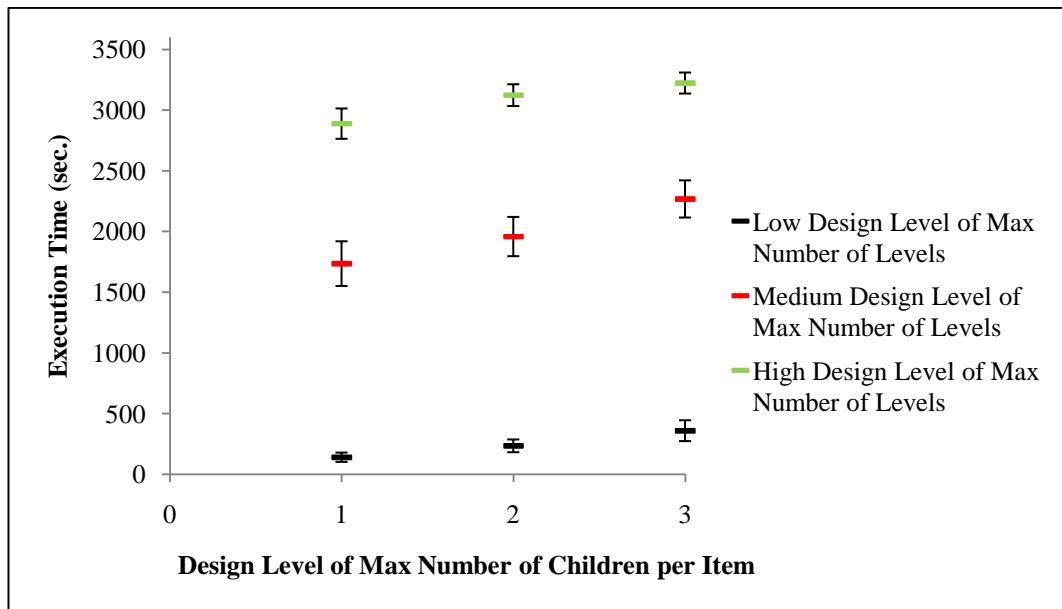


Figure 4.23 Interaction between maximum number of levels and maximum number of children per item in execution time

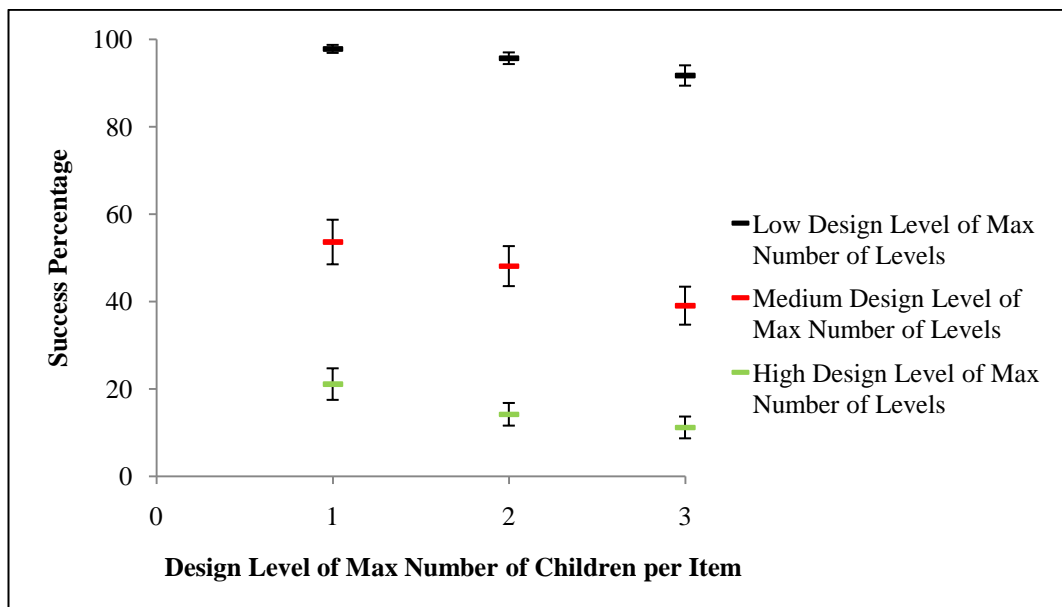


Figure 4.24 Interaction between maximum number of levels and maximum number of children per item in success percentage

4.3.1.11. Interaction between Maximum Number of Levels and Maximum Number of Children per Item

As it can be seen from Figures 4.23 and 4.24, effect of increasing design level of maximum number of levels by one level seems to have much more effect than a one level increase in maximum number of children per item.

4.3.1.12. Interaction between Maximum Number of Levels and Probability Distribution of number of children

Probability distribution of number of children parameter seems to have much less effect than maximum number of levels parameter. Figures 4.25 and 4.26 present interaction between these parameters on average execution time and average success percentage. From Figures 4.25 and 4.26, it can be seen that increasing maximum number of levels parameter by one design level, for example from low to medium design level, increases execution time more than 1000 seconds on the average. However, a one level increase in probability distribution of number of children increases execution time by 100 seconds on the average. From these figures, it can be seen that maximum number of levels parameter is dominant over probability distribution of number of children.

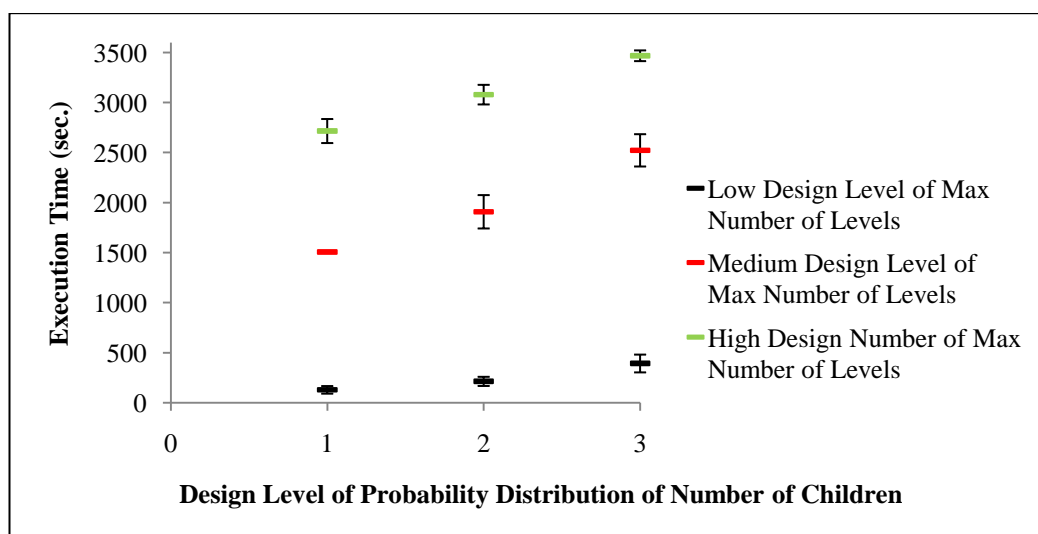


Figure 4.25 Interaction between maximum number of levels and probability distribution of number of children in execution time

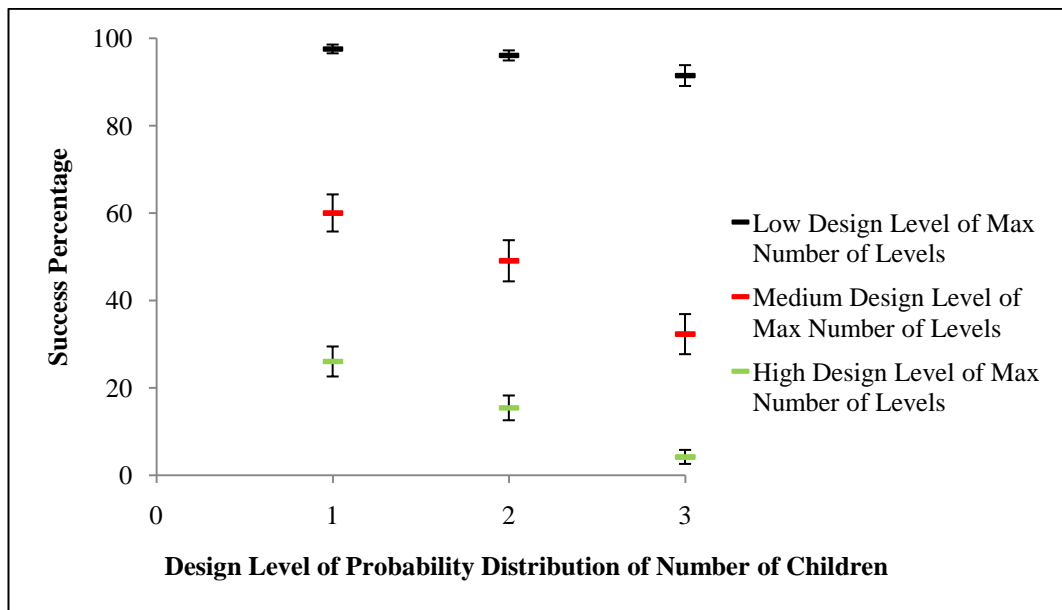


Figure 4.26 Interaction between maximum number of levels and probability distribution of number of children in success percentage

4.3.1.13. Interaction between Maximum Number of Children per Item and Probability Distribution of number of children

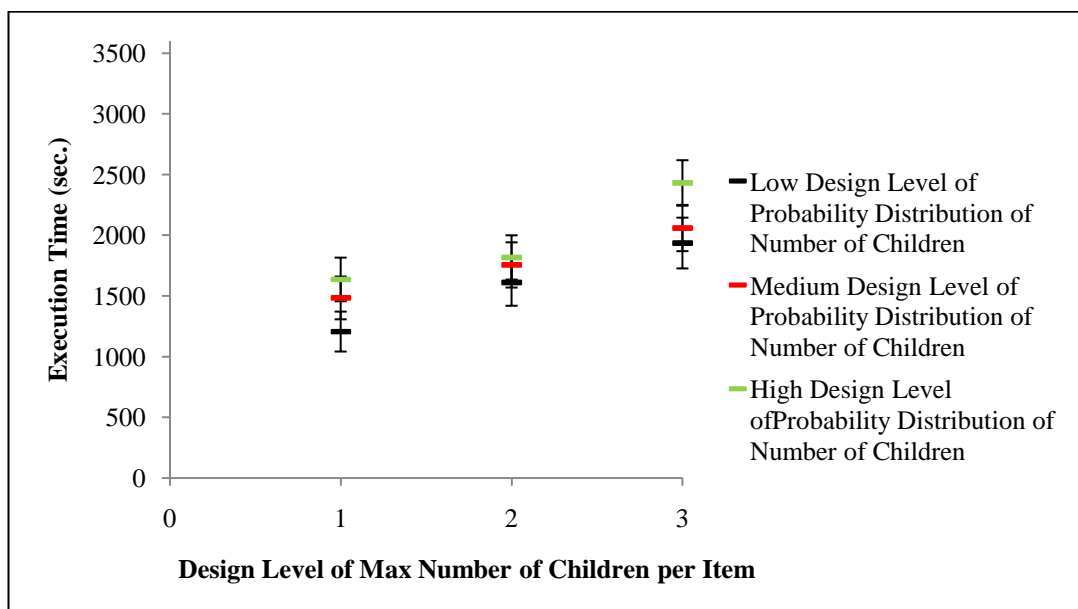


Figure 4.27 Interaction between maximum number of children per item and probability distribution of number of children

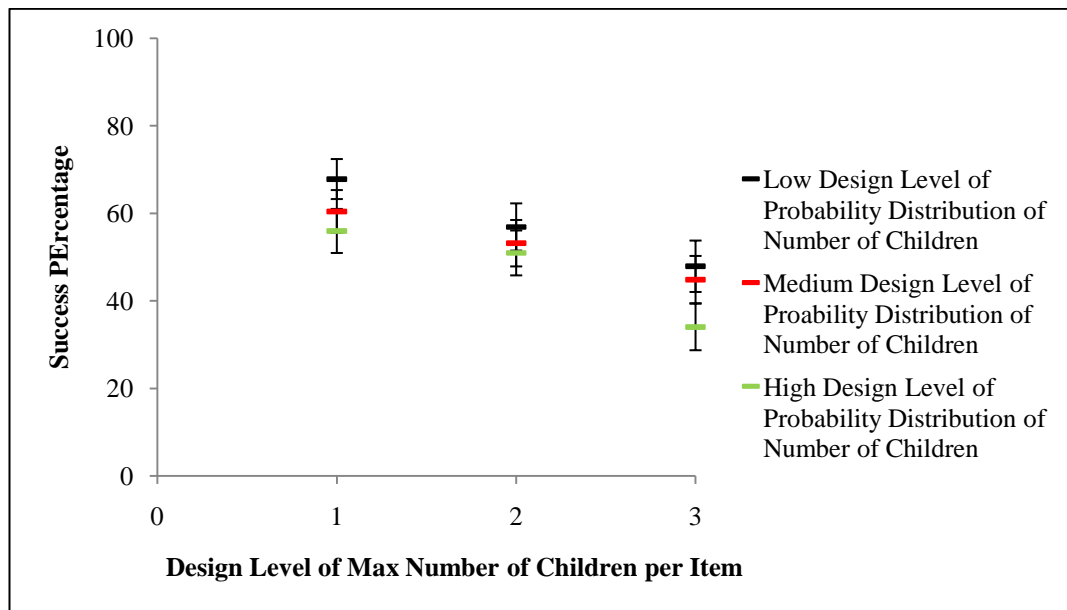


Figure 4.28 Interaction between maximum number of children per item and probability distribution of number of children in success percentage

According to Figures 4.27 and 4.28, almost both parameters respond in same manner for any change in level of maximum number of children per item and probability distribution of number of children. This may also be expected because maximum number of children per item parameter determines the maximum number of children that an item can have. Also, probability distribution of number of children gives probability level of any number of children up to maximum number of children specified by maximum number of children parameter.

Figures (4.17) – (4.28), we analyze that how dominant parameters, on average execution time and success percentage measures, affect each other. It seems that number of root items and maximum number of levels parameters have much more effect than other dominant parameters; maximum number of children per item and probability distribution of number of children. We can conclude from Figures (4.17) – (4.28) that the most dominant parameters are number of root items and maximum number of levels parameters on performance. Other parameters can be listed based on their effects on performance as; probability distribution of number of children and maximum number of children per item. The common attribute of these parameters is that they are product-tree based parameters and increasing/decreasing design level of these parameters directly affects the product-tree structure.

Table 4.10 also indicates mean execution time and (95%) confidence interval values of each level of dominant design parameter pairs calculated in interaction analysis. Likewise Table 4.9, parameters number used in Table 4-10 is used as in Table 4-3. From Table 4.10, it can be concluded that none of pair of design parameters push the execution time limit with 95% confidence level. In almost all design parameter pairs, as design level of pairs increases, confidence level with 95% decreases. For example, when parameters 1 and 2 are at high design levels, confidence interval gets narrower to $\pm 33,38$ seconds from $\pm 78,76$ seconds and mean value also increases to 3506,26 seconds from 3146,88 seconds.

Design Level of 1 st Parameter	Design Level of 2 nd Parameter		Design Parameter Pairs No					
			1-2	1-3	1-7	2-3	2-7	3-7
Low Design Level	Low Design Level	Mean	0,35	697,30	670,36	139,67	128,85	1205,91
		Confidence Level (95%)	$\pm 0,11$	$\pm 140,63$	$\pm 118,79$	$\pm 38,35$	$\pm 37,95$	$\pm 164,53$
Low Design Level	Medium Design Level	Mean	543,61	1024,76	916,19	233,53	213,36	1611,42
		Confidence Level (95%)	$\pm 86,48$	$\pm 157,39$	$\pm 145,1$	$\pm 52,71$	$\pm 45,64$	$\pm 192,84$
Low Design Level	High Design Level	Mean	2411	1232,89	1368,41	358,34	392,27	1935,36
		Confidence Level (95%)	$\pm 125,87$	$\pm 166,46$	$\pm 188,17$	$\pm 85,88$	$\pm 88,98$	$\pm 209,24$
Medium Design Level	Low Design Level	Mean	65,76	1719,08	1494,34	1734,52	1505,60	1483,18
		Confidence Level (95%)	$\pm 19,61$	$\pm 189,71$	$\pm 168,06$	$\pm 184,08$	$\pm 152,74$	$\pm 176,32$
Medium Design Level	Medium Design Level	Mean	2224,02	1851,78	1816,92	1957,81	1908,41	1754,86
		Confidence Level (95%)	$\pm 129,69$	$\pm 185,36$	$\pm 186,82$	$\pm 161,67$	$\pm 166,72$	$\pm 186,26$
Medium Design Level	High Design Level	Mean	3288,85	2007,76	2273,52	2267,55	2522,50	2059,02
		Confidence Level (95%)	$\pm 68,74$	$\pm 193,36$	$\pm 200,12$	$\pm 153,04$	$\pm 161,47$	$\pm 190,61$
High Design Level	Low Design Level	Mean	665,40	2336,31	2159,15	2888,07	2715,44	1635,34
		Confidence Level (95%)	$\pm 85,54$	$\pm 185,97$	$\pm 178,58$	$\pm 125,16$	$\pm 120,13$	$\pm 179,96$
High Design Level	Medium Design Level	Mean	3146,88	2420,51	2464,41	3122,71	3079	1816,82
		Confidence Level (95%)	$\pm 78,76$	$\pm 172,77$	$\pm 172,40$	$\pm 89,75$	$\pm 97,96$	$\pm 182,37$
High Design Level	High Design Level	Mean	3506,26	2561,74	2777,06	3222,42	3467,73	2430,22
		Confidence Level (95%)	$\pm 33,38$	$\pm 162,84$	$\pm 160,96$	$\pm 86,56$	$\pm 53,66$	$\pm 187,35$

Table 4-10 - 95% Confidence Level and Mean Values of Design Parameter Pairs in Execution time (sec.) for Interaction Effect Analysis

This is because, 211 problem out of 243 problem sets (=86%) pushes the execution time limit at high design levels of parameters 1 and 2. Another interesting example is in interaction of parameters 2 and 7. When both parameters are at high design level, mean value increases to 3222,42 seconds and confidence level significantly decreases to 53,66 seconds from 97,96 seconds. The reason is that 208 problems out of 243 problem sets (=85%) cannot be solved within execution time limit. In other interactions shown in Table 4.10, ratio of problem sets that cannot be solved within execution time limit is much lower than interactions between parameters 1 and 2, 2 and 7.

From Tables 4.9 and 4.10, it can be concluded that when most dominant parameters, number of root items and maximum number of levels, at their high design levels the problem gets more complicated and above 80% of problem sets push execution time limit.

CHAPTER - 5

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

As environmental awareness and policies increases, the problem of disassembling of end-of-life (EOL) items becomes important. Also, many manufacturers are forced to collect their EOL items to involve remanufacturing processes by governments. Disassembly is one of the main point in remanufacturing systems because it deals with such vital questions that which and how many of an item is disassembled until which sub-assembly, how many of items should be kept in the inventory, how many of them should be sold or how many of items should be disposed of.

In this thesis, we looked at disassembly scheduling problem on parallel resource environment with new decision options such as; selling and disposing of a part. In literature, disassembly scheduling decisions are based on minimizing total cost. However, there is also need to find answer to questions like; how many of sub-assemblies must be sold to remanufacture items on time, how many of and which sub-assemblies in product structure should be disposed of and which sub-assembly should be assigned to which resource to use available capacity more efficient by considering cost profit analysis.

In Chapter 1, a general understanding of remanufacturing and disassembly systems is provided. The importance of disassembling operations is explained for remanufacturing systems. Different disassembly problem types; disassembly sequencing (planning) and disassembly scheduling are explained briefly. Similarities and differences between disassembly scheduling and material requirement planning (MRP) is pre-

sented. Also, reasons of why disassembly scheduling problem is more complicated than MRP are explained.

In Chapter 2, extensive literature review on disassembly scheduling is given. Similarities between majorities of study on disassembly scheduling are emphasized. Novelities and differences of this study are also briefly presented.

In Chapter 3, a basic model on capacitated disassembly scheduling (CDSP) is given for introduction to CDSP. Possible alternative product structures are introduced with disassembly notations. Parameters of proposed disassembly scheduling problem in this thesis are explained. Then, novel mathematical model is given with definitions of parameters, decision variables and constraints.

Computational analysis is made in Chapter 4. Design parameters defined and used in experimentation are introduced. Detailed steps of testbed generating mechanism are presented. Also, the reason of why there is a need for such a testbed generating mechanism is mentioned. Summary results of experimentation are given. Analysis of effect of each design parameter is made through main effect analysis to observe which parameter makes the problem more difficult to what extent. Then, to investigate impacts of how parameters affect each other, mean interaction analysis is conducted on parameters that affects problem strongly by itself.

The proposed problem is complicated such that it can be solved optimally with reasonable number of parts. For this reason, there is a need for new solution approaches to obtain good solutions in reasonable time for medium and large-sized instances. However, due to complexity of proposed problem, problem-specific solution approaches (local-search based) may not obtain good solutions. As a future research direction, population-based, MIP-based solution approaches may handle complexity of the problem and obtain good solutions. Also, future work should include consideration of environmental effects of disassembly and disposal as an objective or constraint.

REFERENCES

- [1] Gupta, S. M., and Taleb, K. N, (1994), Scheduling Disassembly, *International Journal of Production Research*, 32, (8), 1857-1866
- [2] Taleb, K. N., and Gupta, S.M., (1997), Disassembly of Multiple Product Structures, *Computers Industrial Engineering*, 32, (4), 949-961
- [3] Taleb, K. N., Gupta, S.M., and Brennan, L. (1997), Disassembly of Complex Product Structures with Parts and Materials Commonality, *Production Planning and Control*, 8(3), 255-269
- [4] Pnueli, Y. And Zussman, E., (1997) Evaluating the end-of-life value of a product and improving it by redesign, *International Journal of Production Research*, 35, (921-942)
- [5] Moore, K., Gungor, A., and Gupta, S.M., (1998) A Petri Net Approach to Disassembly Process Planning for Products with Complex AND/OR Precedence Relationships, *European Journal of Operational Research*, 135, (2), 428-449
- [6] Güngör, A., and Gupta, S.M., (1999) A Systematic Solution Approach to the Disassembly Line Balancing Problem, *Proceedings of the International Conference on Computers and Industrial Engineering*, 70-73
- [7] Lee, D.-H., Kang, J.-G. and Xirouchakis, P., (2001) Disassembly Planning and Scheduling: Review and Further Research, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 215, (5), 695-709
- [8] Lee, D.-H., Xirouchakis, P., and Zust, R., (2002) Disassembly Scheduling with Capacity Constraints, *CIRP Annals - Manufacturing Technology*, 51, (1), 387-390
- [9] Kim, H.-J., Lee, D.-H., Xirouchakis, P., and Zust, R., (2003) Disassembly Scheduling with Multiple Product Types, *CIRP Annals-Manufacturing Technology*, 52, (1), 403-406

-
- [10] De Brito, M. P. And Dekker, R., (2004) A framework for reverse logistics. In Dekker, R., Fleishmann, M., Inderfuth, K., and van Wassenhove, L. N., editors, *Reverse Logistics: Quantitative Models for Closed-Loop Supply Chains*, pages 3-27, Springer-Verlag, Berlin.
- [11] Lee, D.-H., and Xirouchakis, P., (2004), A Two-Stage Heuristic for Disassembly Scheduling with Assembly Product Structure, *The Journal of the Operational Research Society*, 55, (3), 287-297
- [12] Lee, D.-H., Kim, H.-J., Choi, G., and Xirouchakis, P., 2004, Disassembly Scheduling: Integer Programming Models, *Journal of Engineering Manufacture*, 218, (10), 1357-1372
- [13] Kang, J.-G. and Xirouchakis, P., (2006) Disassembly Sequencing for Maintenance: A Survey, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 220
- [14] Kim, H.-J., Lee, D.-H., and Xirouchakis, P., (2006) A Lagrangean Heuristic Algorithm for Disassembly Scheduling with Capacity Constraints, *The Journal of the Operational Research Society*, 57, (10), 1231-1240
- [15] Kim, J.-G., Jeon, H.-B., Kim, H.-J., Lee, D.-H., and Xirouchakis, P., (2006) Disassembly Scheduling with Capacity Constraints: Minimizing the Number of Products Disassembled, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 220, 1473-1481
- [16] Kim, H.-J., Lee, D.-H., and Xirouchakis, P., (2007) Disassembly Scheduling: Literature Review and Future Research Directions, *International Journal of Production Research*, 45, (18), 4465-4484
- [17] Barba-Gutierrez, Y., Adenso-Diaz, B. and Gupta, S.M., (2008) Lot Sizing in Reverse MRP for Scheduling Disassembly, *International Journal of Production Economics*, 111, (2), 741-751
- [18] Kim, H.-J., Lee, D.-H., Xirouchakis, P., and Kwon, O., (2009) A Branch and Bound Algorithm for Disassembly Scheduling with Assembly Product Structure, *Journal of the Operational Research Society*, 60, 419-430

- [19] Sheldon M. Ross, (2009) Introduction to Probability and Statistics for Engineers and Scientists, *Academic Press*

- [20] Kim, H.-J., Lee, D.-H., and Xirouchakis, P., (2010) Disassembly Scheduling: Literature Review and Future Research Directions, *International Journal of Production Research*, 45, 4465-4484

- [21] Ilgin, M.A. and Gupta, S.M., (2010) Environmentally Conscious Manufacturing and Product Recovery (ECMPRO): A Review of the State of the Art, *Journal of Environmental Management*, 91, 563-591

- [22] Prakash, P., Ceglarek, D., and Tiwari, M., (2012) Constraint-based Simulated Annealing (CBSA) Approach to Solve the Disassembly Scheduling Problem, *The International Journal of Advanced Manufacturing Technology*, 60, (9-12), 1125-1137

- [23] Morgan, S.D., Roger, J.G., (2013) A systematic literature review of remanufacturing technology, *International Journal of Production Research*