A Disassembly Optimization Problem

by

Ajay Bhootra

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Master of Science
in
Industrial and Systems Engineering

Dr. Subhash C. Sarin, Chair
Dr. Hanif D. Sherali
Dr. Kyle Y. Lin

September 20, 2002
Blacksburg, VA

Keywords: Disassembly Optimization, Integer Programming, Lagrangian Relaxation, Branch and Bound Method, Asymmetric Traveling Salesman Problem

A Disassembly Optimization Problem

by

Ajay Bhootra

Dr. Subhash C. Sarin, Chairman

Industrial and Systems Engineering

ABSTRACT

The rapid technological advancement in the past century resulted in a decreased life cycle of a large number of products and, consequently, increased the rate of technological obsolescence. The disposal of obsolete products has resulted in rapid landfilling and now poses a major environmental threat. The governments in many countries around the world have started imposing regulations to curb uncontrolled product disposal. The consumers, too, are now aware of adverse effects of product disposal on environment and increasingly favor environmentally benign products.

In the wake of imminent stringent government regulations and the consumer awareness about ecosystem-friendly products, the manufacturers need to think about the alternatives to product disposal. One way to deal with this problem is to disassemble an obsolete product and utilize some of its components/subassemblies in the manufacturing of new products. This seems to be a promising solution because products now-a-days are made in accordance with the highest quality standards and, although an obsolete product may

not be in the required functional state as a whole, it is possible that several of its components or subassemblies are still in near perfect condition.

However, product disassembly is a complex task requiring human labor as well as automated processes and, consequently, a huge amount of monetary investment. This research addresses a disassembly optimization problem, which aims at minimizing the costs associated with the disassembly process (namely, the costs of breaking the joints and the sequence dependent set-up cost associated with disassembly operations), while maximizing the benefits resulting from recovery of components/subassemblies from a product. We provide a mathematical abstraction of the disassembly optimization problem in the form of integer-programming models. One of our formulations includes a new way of modeling the subtour elimination constraints (SECs), which are usually encountered in the well-known traveling salesman problems. Based on these SECs, a new valid formulation for asymmetric traveling salesman problem (ATSP) was developed. The ATSP formulation was further extended to obtain a valid formulation for the precedence constrained ATSP. A detailed experimentation was conducted to compare the performance of the proposed formulations with that of other well-known formulations discussed in the literature. Our results indicate that in comparison to other well-known formulations, the proposed formulations are quite promising in terms of the LP relaxation bounds obtained and the number of branch and bound nodes explored to reach an optimal integer solution. These new formulations along with the results of experimentation are presented in Appendix A.

To solve the disassembly optimization problem, a three-phase iterative solution procedure was developed that can determine optimal or near optimal disassembly plans for complex assemblies. The first phase helps in obtaining an upper bound on our maximization problem through an application of a Lagrangian relaxation scheme. The second phase helps to further improve this bound through addition of a few valid inequalities in our models. In the third phase, we fix some of our decision variables based on the solutions obtained in the iterations of phases 1 and 2 and then implement a branch and bound scheme to obtain the final solution. We test our procedure on several randomly generated data sets and identify the factors that render a problem to be computationally difficult. Also, we establish the practical usefulness of our approach through case studies on the disassembly of a computer processor and a laser printer.

ACKNOWLEDGEMENTS

I wish to express my utmost gratitude for my advisor, Dr. Subhash Sarin, for his guidance and support over the past years. He is always available to advise and discuss research issues and his ideas and suggestions have helped me in tackling several difficult problems I faced during the course of this research. I sincerely appreciate the support of Dr. Hanif Sherali, whose numerous suggestions on modeling and algorithmic issues have led to a considerable improvement in this thesis. I thank Dr. Kyle Lin for agreeing to serve on my committee and taking interest in my research.

I cannot thank my family enough for all their love and affection. My parents have been a constant source of encouragement and I would not have achieved this milestone without their blessings. Thanks to my brother, Abhay, and my sister-in-law, Richa, for making me feel at home during my stay in US. I am forever indebted to them for their care and support and I can always count on them for any help and guidance I need in my life. Last but not the least; I thank my sister, Archana, for always being there when I need someone to talk to and cheering me up during times of frustration.

Finally, I would like to thank my friends from Virginia Tech, SGSITS, Indore and St. Mary's Convent School, Dewas. I will always cherish the memories of time I have spent with them and I hope they achieve success in their endeavors.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 The Hazards to Environment – Background and Statistics

The past century witnessed tremendous technological advancements in practically every field of human endeavor. Although the mankind has reaped enormous benefits from these advancements in terms of improvement in the standard of living, the damage that has been done to the environment in the process cannot be underestimated. The adverse impacts on the environment seem to be an inevitable byproduct of technological progress. It all started in early 1960s when the major environmental concerns were limited to air and water pollution in the highly populated or industrialized areas. In the late 1960s, however, people realized the inherent dangers associated with tropical deforestation, which resulted in extinction of forests that were once considered as self-replenishing natural resources (Gungor and Gupta, 1999). After 1974, when the first ozone depletion hypothesis was put forward, the depletion of ozone layer and the consequent global warming due to Greenhouse effect became the most important concerns of environmental scientists and researchers. More recently, the focus has not only been on the ways to protect the stratosphere, but also on preventing damage to the lithosphere resulting from the entry of harmful substances in the waste stream due to land filling.

The rapid advances in technology have resulted in reduced product life cycle, not due to degraded operational condition of the product itself, but rather due to increased rate of technological obsolescence and very high cost of service (Allenby, 1994). The example of computers is a case in the making. It has been observed that a PC becomes obsolete in less than 12 months after it leaves the factory and the figures indicate that there are more than seventy million obsolete computers sitting in the basements of various organizations (Moore and Anhalt, 1995). An estimated 150 million discarded personal computers will have been land filled by the year 2005 (Chen *et al.*, 1993). Studies indicate that almost half a million home and office appliances are disposed on a yearly basis (Renich, 1991). More than 10 million vehicles reach the end of their useful lives every year in United States (Chen *et al.*, 1993). According to the National Academy of Sciences, 94% of the raw material extracted from the limited-earth inventory enters the waste stream within 9 months (Moore and Anhalt, 1995). According to the US Environmental Protection Agency (EPA), the amount of waste generated in US has reached a whopping 196 million tons from 88 million tons in 1960's (Gungor and Gupta, 1995). This not only results in rapid depletion of valuable resources, but also adversely affects the environment. The problem is further intensified by ever-increasing demand for consumer products, which boosts the disposal rates and associated costs. During the 1980s, a tenfold (per unit volume) increase in these costs has been observed in United States and a large portion of landfill sites is expected to reach capacity within the next few years (Brooke, 1991). All these statistics outline the urgent need for resorting to product recycling. Until recently, the manufacturers were reluctant to consider recycling as a possible option for two main reasons. Firstly, the recycling costs were considered unjustifiable when compared to the gains resulting from subsequent material recovery; and secondly, the resale value of the recycled product cannot possibly match the market value of the virgin product (Gungor and Gupta, 1995). However, the government regulations and consumer awareness about the environmentally benign products have now enhanced the significance of recycling and added further incentives and pressures on manufacturers for producing recyclable products. Product recycling, which, until recently, was considered infeasible due to cost considerations, may actually become inevitable in near future. The label of *green* will definitely become a strong competitive advantage for enterprises and products in the world market.

All these concerns have prompted the campaigns such as design for environment (DFE), design for recycling (DFR) and design for disassembly (DFD). These three concepts are briefly described below:

1.2 Design for Environment (DFE)

Design for Environment covers a wide range of product development activities including selection of appropriate materials, examination of the product usage phase to reduce environmental impact, designing for energy efficiency, minimization of industrial residues during manufacturing, designing for end-of-life, improvement in packaging and reduction in the use of environmentally relevant substances (Rose, 2000). Fiksel (1996) defines DFE as: "a systematic consideration of design performance with respect to environmental, health, and safety objectives over the full product and process life cycle." According to the author, DFE can be broken down into many stages including manufacturing, consumer use, and the end-of-life of the product. Throughout these stages, different forms of design strategies can be envisioned as the pieces of DFE. For example, in order to minimize the effect of the product on the environment at the manufacturing stage, design objectives may include design for energy conservation to reduce the energy use in production and to be able to use renewable forms of energy, and design for minimizing the discharge of hazardous byproduct during production. Similar concerns are also valid during the distribution of the product. Finally, during the end-of-life stage of the product, there are design objectives to increase the output of the product recovery. These include design for material and product recovery, design for disassembly, design for waste minimization, design under legislation and regulations, etc.

1.3 Design for Recycling (DFR)

Design for recycling (DFR) suggests making better choices for material selection such that the processes of material separation and material recovery become more efficient. Some general characteristics of DFR are as follows (Gungor and Gupta, 1995):

- long product life with the minimized use of raw materials (source reduction),
- easy separation of different materials,
- fewer number of different materials in a single product while maintaining compatibility with the existing manufacturing infrastructure**,**
- fewer components within a given material in an engineered system,
- increased awareness of life cycle balances and reprocessing expenses,
- increased number of parts or subsystems that are easily disassembled and reused without refurbishing,
- more adaptable materials for multiple product applications, and
- fewer 'secondary operations' reducing the amount of scrap and simplifying the recovery process.

1.4 Design for Disassembly (DFD)

DFD is one of the several aspects of DFE. The issue of design for disassembly was first addressed by Boothroyd and Alting (1992). At that time, the Appliance Recycling Centers of America (ARCA) had initiated the efforts to research in this field to facilitate recycling. Integrated design, certain fastening and assembly principles, and surface coatings can make it very difficult to disassemble the product and to separate hazardous materials from the useful ones. DFD initiatives lead to the correct identification of design specifications in order to minimize the complexity of the structure of the product by minimizing the number of parts, increasing the use of common materials and choosing the fastener and joint types which are easily removable (Gungor and Gupta, 1995).

1.5 Motivation for Current Research

Despite the necessity of using eco-friendly processes and products, an average American company has not implemented environmentally conscious practices, lagging behind its European competitors (Boothroyd and Alting, 1992). In fact, many European countries, notably Germany and Netherlands, have legislations pending for placing the primary responsibility for safe product disposal at the end of its life on the primary producer (Meacham *et al.*, 1999). In Germany, the Decree on Electronic Waste and the Decree on Used Cars will force the manufacturers to reclaim the product after its useful life, to reuse the recyclable portion and to dispose of the residue (Jovane *et al.*, 1993). In 1993, European Union introduced a set of guidelines, the Eco Management and Audit Scheme (EMAS), which, although voluntary, has signaled that the environmental responsibility should lie within the industry (Kriwet *et al.*, 1995). This trend is not limited to the

European community only. Japan is in the process of legislating the "Home Appliance Recycling Law", which may become effective this year, and which will require mandatory recycling of televisions, refrigerators, washing machines, and air conditioners after the end of their useful lives (Huang *et al.*, 2000). The government and the industries in United States need to take the "environmental initiative" in order to change the status quo. There is an urgent need to develop the guidelines and procedures to aid the manufacturers in adopting environmental standards in product design and processes. These policies should be based on concrete scientific and engineering studies and should be developed and implemented in such a manner that a high level of industrial compliance of environmental standards is achieved without immense and unbearable economic implications.

At the same time, it must be observed that in order to remain competitive, the manufacturers use processes and systems that ensure high level of product quality. Consequently, even though the product as a whole may not be in the required operational condition after the end of its conventional useful life, there might be components and subassemblies in near perfect condition, which can be reused in new products or in the remanufacture of old products (Dewhurst, 1993). The efficient retrieval of these parts/subassemblies will not only cut down the production cost, but will also reduce the associated disposal costs and consequent environmental hazards. It is clear that in order to retrieve such useful components and subassemblies from the product after the end of its useful life, the product needs to be disassembled in such a manner that the benefits resulting from reusing these parts and subassemblies overweigh the combined costs of disassembly and disposal of the complete product. It must be observed, however, that disposal of the entire product without possible recycling may no longer be a viable option in wake of the imminent government regulations. The motivation for this research stems from the above considerations.

1.6 Problem Statement and Research Objectives

After the useful life of the product is over, it is returned to the manufacturer for de-manufacturing, which includes disassembly for recovery of some or all components or subassemblies, remanufacture of the final assembly or of its most valuable components for resale, or disposal. Disassembly is the backward process of dismantling final products through successive steps that may follow the reverse order of assembly operations. The disassembly optimization problem that we address in this research can be defined as follows: Given a product to be disassembled, determine a disassembly sequence and the depth of disassembly so as to:

- maximize the net revenue resulting from the benefits accrued from the recovery of components and subassemblies, and
- minimize the costs incurred while disassembling a product and/or disposing of the product or its components.

Realizing the importance of cost-effective disassembly of the product after the end of its useful life and the resulting implications in the development of ecosystem-friendly standards, this research is aimed at developing a mathematical abstraction of the above mentioned disassembly problem using mathematical programming techniques and subsequently, developing a solution approach for the above problem. This approach will result in minimizing the disassembly costs (cost of breaking the joints and removing the fasteners and the set-up costs) while realizing maximum benefits from the recovered parts/subassemblies.

1.7 Organization of Thesis

This thesis is organized as follows:

In Chapter 2, we present a comprehensive review of literature pertaining to disassembly modeling, techniques of estimating recovery values and disassembly costs, and solution approaches for disassembly optimization.

In Chapter 3, we present new integer programming models for disassembly optimization and validate our models through a case study on disassembly of a computer processor. One of the formulations includes a new way of modeling the subtour elimination constraints (SECs), which are usually encountered in the well-known traveling salesman problems. Based on these SECs, a new formulation of the asymmetric traveling salesman problem is developed. This formulation is further extended to obtain a new formulation for the precedence constrained ATSP (PCATSP). The formulations of ATSP and PCATSP along with the associated results of experimentation are presented in Appendix A.

In Chapter 4, we present a three-phase solution methodology for the model obtained in Chapter 3. In the first phase, we use the Lagrangian relaxation technique for obtaining a tighter bound for the problem. This bound is further improved in the second phase by incorporating valid equalities as cutting planes in the model in an iterative manner. Finally, in the third phase, we switch to the branch and bound method for solving the integer program.

We conclude this research work in Chapter 5 with the results of our experimentation to solve the disassembly optimization problem. We also present a case study on the disassembly of a computer printer and apply our solution procedure to obtain optimal disassembly plan. Finally, we provide some insights about future research directions.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

During the last few years, the increasing concern towards environmental safety has resulted in substantial research towards the development of eco-friendly end-of-life alternatives for various products. In the wake of imminent stringent government regulations and the growing environmental consciousness among consumers, it becomes imperative for the manufacturers to adopt eco-friendly practices. The academia has responded promptly to these concerns and several authors have addressed the issue of developing feasible, cost-effective and environmentally benign end-of-life alternatives for the products. Over the years, the related research has evolved mainly along three axes. The first deals with product data modeling suitable for disassembly sequence generation and material recovery. The second addresses the economic aspects associated with disassembly comprising of disassembly profitability analyses based on the evaluation of disassembly effort, the value of recycled materials and the disposal costs. The final axis concerns the development of models and solution approaches for disassembly optimization. The following literature review describes the pertinent research that has been done in these interrelated areas.

2.2 Representation of Product Data for Disassembly

A clear and precise representation of various components, subassemblies and their mating relationships in the final assembly aids in the development of suitable model for disassembly analysis and optimal sequence generation. Such a representation usually results in the generation of all feasible disassembly sequences and provides a basis for deriving a search algorithm to obtain the optimal sequence. Consequently, several authors have addressed the issue of product data modeling. One of the earliest and well known methods of capturing geometrical and relational product data for disassembly studies, and generate disassembly trees from the basic part mating relationships is through the AND/OR graphs proposed by Homem de Mello and Sanderson (1991). The AND/OR graph representation of an assembly gives all possible disassembly sequences for the product, while reducing the number of nodes used in the graph by combining the common nodes in different sequences. Consequently, there is just one node corresponding to a given part/subassembly with possibly multiple edges (called hyper arcs) incident on it as well as emanating from it. The basic idea behind the development of these graphs is to view each subassembly as a configuration comprising of subsets of individual parts. Thus, a single disassembly process can be viewed as a transition from one configuration to another configuration with a reduced number of subsets. The strength of an AND/OR graph lies in the fact that it is not just a graph showing a precedence relationship among mating components and subassemblies but a representation that reveals different sequences comprising of exactly similar configurations of parts/subassemblies. Although an AND/OR graph provides a compact representation of all feasible assembly sequences, it will still have large number of nodes in the case of products with large number of parts. Also, the subassemblies included in AND/OR graphs may not correspond to actual subassemblies in the product's Bill of

Material (BOM). As a result, they may not be of much practical value to industrial problems.

Zussman *et al*. (1994) extended the idea of AND/OR graph by adding a decision tree of feasible recycling operations to each node of the graph. The resultant graph was called the "Recovery Graph". The idea behind such an inclusion was to make the representation more complete from the recycling point of view. Thus, a recovery graph will not only represent all technically feasible disassembly sequences, it will also show the possible end-of-life options for individual components and subassemblies. However, this representation suffers from the same limitations as the AND/OR graph and the additional decision trees at each node makes the representation too complex to aid the development of rigorous mathematical models.

Subramani and Dewhurst (1991) also used the AND/OR graph representation with some extensions. The relational model constructed by them consists of four entities namely: Parts, Contacts, Attachments and Relations. Parts represent the discrete components of the assembly. Contacts represent the mating configuration between the surfaces of parts and may be planar, cylindrical, polygonal, slotted or threaded cylindrical contact. Attachments are entities like screws, press fit attachment, glue attachment and clip attachment which are used to bind the contacts. Finally, relations are used in the model to establish association between parts, contacts and attachments. A disassembly diagram (DAD) is constructed based on this relational model for the generation of all possible disassembly sequences. The DAD establishes precedence relationships between various components along the six possible coordinate directions and therefore accounts for any constraints to the movement of parts along these directions. Such a representation, although complex, provides a direct means for generation of optimal disassembly sequences.

In his seminal paper addressing the issue of recovery problem in product design, Navin-Chandra (1994) outlined the development of a novel computer-based decision-making tool called *ReStar* for DFD/DFR analysis. The optimization algorithm embedded in this comprehensive tool can operate with any tree-like disassembly diagram. Navin-Chandra used the disassembly diagram suggested by Subramani and Dewhurst (1991) with some extensions. These extensions were incorporated to handle:

1. various types of fasteners
2. subassemblies
3. tool/hand access

The disassembly graph is represented in form of a table, and all the joints and fasteners are included as parts. The disassembly table provides information regarding the accessibility of individual components along the coordinate directions of motions. For each part, all other parts that obstruct the motion of that part in a particular direction are specified. The disassembly optimization algorithm is based on following two rules:

1. If the direction of motion of a part is unobstructed in a given direction, then it can be removed in that direction.
2. If a part is held only by a joint in a particular direction, then it can be removed by undoing that joint.

The recursive application of these rules results in complete disassembly of the product. *Restar* keeps track of the associated disassembly costs and recovery revenues during the disassembly process and also keeps track of the sub-assemblies comprising of compatible materials, which can be used as such without further disassembly.

Laperriere and ElMaraghy (1992) proposed a directed graph network representation for generating optimal assembly and disassembly plans. Unlike the previous methods discussed, this representation aims to improve the efficiency of search algorithms by reducing the number of candidate disassembly sequences. Hence, this representation does not give all the possible sequences and this reduction is achieved through imposition of geometric feasibility and accessibility constraints. The geometric feasibility constraints are derived on the basis of a part/subassembly's translational degrees of freedom. These constraints account for the fact that during a certain disassembly process, it might not be possible to achieve translational motion of a part along all the six possible directions (three translations each along the positive and negative x, y and z coordinate directions). The number of possible sequences is substantially reduced by considering the fact that at a certain step in disassembly process, all the six degrees of freedom for a given part might be restricted, thereby eliminating the possibility of its removal; and consequently, eliminating a possible disassembly sequence. Similarly, accessibility constraints account for the fact that at a certain step in the disassembly process, the restricted access of the tool to hold and remove a certain part might prevent the execution of that particular disassembly operation, thereby eliminating a possible disassembly sequence. Although, this representation seems to be promising, especially for complex assemblies, the scheme does not provide for any rigorous mathematical conceptualization of the accessibility constraints. Also, it might not always be possible to foresee the possibility of a restricted tool access or a restricted translational motion of a part at a particular step in disassembly process.

Pu and Purvis (1995) suggest a different kind of approach based on case based reasoning and case adaptation techniques and use it for assembly sequence generation. The case based reasoning approach is used to solve new problems by adapting solutions that were obtained from a previous similar problem. In this methodology, previous experiences are stored in memory and then new problems are solved through the following steps (Bergmann, 1998):

1. retrieval of similar experiences about similar situations from the memory.
2. partial or complete reuse of the past experience in context of the new problem, or adaptation of the past experience to accommodate the differences.
3. storing of new experience in the memory (for future use)

The key idea in the scheme suggested by Pu and Purvis is the curtailment of search space in assembly sequence generation by utilizing the information obtained from previous cases, thus reducing the complexity of search. Rather than using a single previous experience as a basis (called case framing), they suggest a case combination approach, which stores and retrieves information from several previous similar cases. In order to make sure that all the cases are compatible with respect to their individual constraints i.e. none of the constraints are violated when the information from all the cases is used simultaneously, each individual case is viewed as a constraint satisfaction problem (CSP). A Constraint Satisfaction Problem (CSP) can be formally defined by:

1. a set of variables, each of which has a discrete and finite set of possible values (their domain), and
2. a set of constraints among these variables.

The solution to a CSP is to find a value for each variable, from their respective domain, which satisfy all the constraints.

A relational model is developed, which serves as the basis of formulating the assembly sequence generation problem as a CSP. The relational model captures the geometric, spatial, mechanical, and stability-related constraints associated with the disassembly of a particular product. Each connection between the two parts in a relational model is defined as a CSP variable. The solution to the CSP (and consequently, to the sequence generation problem) is feasible as none of the constraints are violated.

Molloy *et al.* (1991) used a combination of boundary representation (B-rep) CAD data structures and simple mating relationships to model final assemblies and generate relationship graphs. The B-rep of products has been prevalent in CAD systems for years. It uses parametric equations to represent a product in terms of its surface boundaries i.e. vertices, edges and surfaces together with connectivity and adjacency information relating them. The components in this representation are specified by their B-reps, and their relationships in an assembly are specified by mating conditions like against, fits, contact and tight-fits between all the components. The "against" condition applies to two planar surfaces and merely ensures that they are in contact at all times and also permits relative motion between them. The "contact" condition is similar to the "against" condition except that no movement is permitted here. The "fits" condition holds between the centerlines of a solid cylinder and a hole, and permits rotational motion as well as translational motion along the centerline. The "tight-fits" condition is similar to "fits" condition except for the fact that rotational motion is not permitted in this case. The relationship graph is obtained through systematic analysis of the part and subassembly information, and the mating conditions that exist at various contact surfaces.

Another hierarchical network representation scheme called LINKER has been suggested by Ishii *et al*. (1994). The nodes in this representation correspond to components and subassemblies and the links represent the physical or geometrical connections between the nodes. The LINKER representation simply gives the mating relationships between components and subassemblies and does not actually show all possible disassembly sequences in the network. Hence, this representation is not suitable for search-based optimization algorithms. The disassembly evaluation is based on "clumping" the components together on the basis on similarity in designer specified characteristics. The analysis takes into account the time to remove fasteners and components as well as the set-up time. Although such a representation might aid in quick disassembly analysis, the solution obtained is not necessarily optimal. Also, the "clumps" may not be similar to actual subassemblies in product's Bill of Material.

Moore *et al.* (1998) proposed the automatic generation of a disassembly petri net (DPN) from disassembly precedence matrix (DPM). The elements of a DPM represent the geometric precedence relationships such as AND, OR and complex AND/OR relations among the components. An AND relationship exists between parts p1 and p2 in relation to part p3, if both p1 and p2 have to be removed before retrieving p3. An OR relationship exists between parts p1 and p2 in relation to part p3, if either p1 or p2 must be removed before p3. For n components, the DPM, DP = $\{dp_{gh}\}$, g = 1….n, h =1,…., n is defined as follows:

$$dp_{gh} = \begin{cases} 0, \text{ if removal of part g does not precede removal of part h} \\ 1, \text{ if removal of part g (AND) precedes the removal of part h} \\ d, \text{ if removal of part g (OR) precedes removal of part h} \end{cases}$$

Here, d represents the direction (one of the possible 6 coordinate directions) along which a part is removed when it has an OR relationship with its successor. Hence, dp_{gh} equals 1 when part g has an AND precedence relationship with part h and equals d if it has an OR precedence relationship with part h. The proposed algorithm automatically generates the DPN for the given final assembly. The disassembly petri net can then be used to generate all feasible disassembly process plans (DPPs) and select the best out of those; or alternatively, some heuristic may be used for generating a near-optimal DPP.

2.3 The Economics of Disassembly and End-of-Life-Cycle Alternatives

Disassembly of a product may turn out to be a value-added function, if the final assembly contains recyclable components or subassemblies of significant value such that benefits resulting from recovering such items overweigh the costs associated with the disassembly process itself and with the disposal of the product. Clearly, for determining the usefulness of disassembly and subsequently proceeding to develop methodologies to optimize the disassembly process, it is imperative to have an estimate of the various costs and recovery values associated with the disassembly process. Generally, there are no direct means of determining the exact values of these measures. Hence, the values are estimated indirectly based on the available information for a product and its disassembly process. A precise estimate of such values will ensure that subsequent steps of disassembly optimization will produce accurate results, which will consequently help in the selection of the best option among the various end-of-life alternatives for individual components and subassemblies. Thus, cost-benefit analysis associated with a disassembly process is of critical nature and vast amount of research has been done to address these issues.

Before the cost and benefit studies are initiated, it is important to analyze the various end-of-life options carefully. Kriwet *et al.* (1995) identified different possible recycling options for a component or subassembly. These include further disassembly, shredding and separation processes, reusing or using on, utilizing and dumping (or disposal). If the shape of the original product is maintained for future tasks, it is termed as “using”, while making use of the material obtained after dissolving the original shape is defined as “utilization”. If the function of the recycled product is same as the original product, then it is said to be “reused”, otherwise the process is termed as “using on”. Material recycling is the simplest form of value recovery, and has been employed for years in the production cycle of steel, aluminum and other metals ("Design for aluminum recycling" in "Automotive Engineering" [October, 1993]). However, because of the various costs and benefits associated with disassembly, disposal and reuse, material recycling may not be the best option in all cases.

Boothroyd and Alting (1992) addressed the importance of design for assembly and disassembly and implementing these ideas in the early conceptual stage of product design to achieve simplification of products, and reduction in assembly and manufacturing costs. They described various prevalent approaches for analyzing the assembly including the Hitachi Assembly Evaluation Method (AEM), Boothroyd Dewhurst DFA method, the Assembly-Oriented Product Design method, the "Lucas" method, and the design for assembly cost effectiveness method (DAC) developed by Sony corporation.

The authors also developed the rules and guidelines for product design for ease of disassembly (DFD). They proposed a new approach in which all the phases of a product life cycle (development, production, distribution, consumption and recycling or disposal) are considered simultaneously from conceptual design stage to detailed design stage. Various criteria were suggested that would guide the design of a product. These include environmental protection, working conditions, resource utilization, life-cycle costs, manufacturing properties, product properties and company policy. They emphasized the importance of using life-cycle design as a framework for DFD; they suggested that DFD must fulfill the requirements of all phases of product life-cycle rather than just the recycling phase. Finally, a detailed description is provided of the various research and development efforts underway in the area of DFD.

Navin-Chandra (1994) stipulated the motivating factors behind the development of green design paradigm. These factors include pro-active market leadership, holism and simplification, safety and liability, remanufacture and reuse, materials mortgage, and stepped obsolescence. These are "pull" factors which indicate the advantages of green design, and which are of greater significance as compared to "push" factors like government legislations, which might have serious business and economic implications. Pro-active market leadership requires initiative in development of environment-friendly products, which will help the companies establish a favorable image among consumers. Many governments now have official ecolabelling schemes, identifying a particular product as ecofriendly (Jovane *et al.*, 1993). A holistic view of product, in terms of recycling and material compatibility, can lead to product simplification, which can result in reduction in number of vendors, assembly operations and inventories. Due to safety and liability considerations, some manufacturers, especially in electronics industry, might prefer to dispose their product themselves rather than having the consumers to do it. The reason is that the original manufacturer is held financially liable in case of groundwater contamination due to the disposed product; and the resulting lawsuits may run into millions of dollars. Remanufacturing and reuse are obviously advantageous due to possible savings in manufacturing costs. Materials mortgage implies that consumers can actually lease or mortgage some of the expensive materials of a product for a longer duration of time, and in the meantime can buy and return (for recovery) the original product several times. In this way, the manufacturers can include some of the expensive materials in the product to improve quality and at the same time, can ensure that product is not priced at an unreasonably high value. Another useful concept is that of stepped obsolescence. A new hi-tech product in its early years is usually bought only by enthusiastic early adopters. A trade-in scheme is developed, wherein manufacturers reclaim this product from the early adopters within 2 years, and sell it to late adopters at a lower price after remanufacturing. Thus, the scheme of stepped obsolescence automatically ensures product recovery and remanufacturing.

Lund (1984) highlighted the importance of remanufacturing in view of current scarcity of materials. The author proposed a complete scheme for undertaking remanufacturing, from the point where it is relinquished by the user due to faulty or worn-out condition, to the point where it restored to a new-like condition and ready for reuse. He discussed remanufacturing practices in automotive, industrial equipment, and consumer and residential product sectors, and pointed out the need for developing remanufacturing technologies and investing in the study of disassembly process.

Holt (1993) outlined the current status of materials and components recycling in automotive industry. He provides data regarding the recyclable metal and plastic composition of passenger cars, gives a synopsis of various recycling initiatives underway in this area and provides details of different recycling procedures applicable for individual interior and exterior components of passenger cars. Two other articles in the same issue of *Automotive Engineering* (October, 1993) reported on the costs and benefits of plastics and aluminum recycling, emphasizing the important role of disassembly in determining the effectiveness of recycling.

Jovane *et al.* (1993) emphasized the importance of disassembly as a key issue in product life cycle. The author provides an overview of disassembly research being conducted at various universities, industries and research institutions. His survey is focused on ongoing research areas such as recycling techniques, the disassembly problem, life cycle design (including estimation of various life cycle costs), product design for ease of disassembly (DFD) and recycling (DFR), process design for selection of disassembly strategies, and system design to develop methodologies for the:

- configuration of manual and automated disassembly systems,
- economic justification of disassembly systems, and
- organization of logistic network for reclaiming, dismantling and recycling.

Tipnis (1994) advocated environmental stewardship on part of manufacturers in order to remain competitive in the future market. He proposed a paradigm ‘E’ (E stands for environment) for manufacturers to handle imminent government regulations, consumer preference for green products and competition from other manufacturers. The paradigm ‘E’ acts as starting point for design for sustainability, and emphasizes designing of products and processes with specific sustainable growth rates for the control of pollution. Specific cases from automotive and electronic industries are presented, which illustrate the successful application of paradigm ‘E’.

Boks *et al.* (1996) presented the results of applying two approaches of disassembly modeling to a television set. Disassembly modeling is a collective term to represent disassembly sequence planning, disassembly operations planning and disassembly evaluation. The results presented in the paper did not address the issues of determination of disassembly sequence and disassembly depth. One of the approaches for disassembly modeling is based on theoretical work, while other methodology is derived based on actual observations. Theoretically derived time estimates for disassembly operations are obtained through standard motion-time studies. Measurements from actual disassembly operations are derived by timing disassembly operations in situations where these operations are performed on a daily basis. The first disassembly model used was the Kroll's disassembly evaluation method. This method of quantifying the ease-of-disassembly was developed after several years of experimentation on electrical appliances. The method consists of compiling two lists – the tasks list and the tools list. The tasks list includes 16 standard disassembly operations like unscrew, cut etc. The tools list consists of 24 tools that are used in disassembly. Four different sources of difficulty in disassembly were identified. These include:

1. Accessibility, which measures the ease with which a tool or hand can reach a part,
2. Positioning, which measures the degree of precision required to position the hand or tool,
3. Force, which is the measure of amount of effort for disassembly, and
4. Base time, the time required to perform the basic task movements (excluding the time required in accessing the part, positioning the tool, and exerting the force).

All these factors were measured on a common scale of time and were rated on a scale of 1-10. These times were estimated using Maynard's Operation Sequence Technique (MOST), which is one of the widely used techniques in work-measurement studies.

The second model that was employed for analyzing the television set disassembly was the Philips ECC end-of-life cost model, which was developed at the Environment Competence Center (ECC) of Philips Consumer Electronics. This model uses a cost-benefit analysis to determine the optimal depth of disassembly. The best end-of-life strategy is determined based on these results. The results of the application of these two models and their comparison was provided in the concluding section of the paper.

Steinhilper (1994) addressed the issues of design for recycling and design for remanufacturing in case of mechatronic and electronic products from the European viewpoint. Several case studies are presented in the paper, which demonstrate the layout of disassembly lines for certain products at various disassembly facilities in Germany. The author also presents some estimates of the contributions of disposal expenses in the total life-cycle costs of the products.

Corbet (1996) introduces a new concept called design for value maximization (DFVM), which is intended to integrate environmental concepts (like DFE, DFR and DFD) with business concepts (like activity-based costing and process mapping). The key idea behind this methodology is that even after incorporating various environmentally dedicated concepts into product and process design (which will result in additional capital investment initially), it is possible to achieve increased profitability in the long run if other familiar business concepts (activity based costing in particular) are also employed simultaneously. The author provides a framework for implementation of DFVM and discusses the associated barriers.

Stuart *et al.* (1995) describe mathematical modeling approach to evaluate the environmental impacts and yield trade-offs in electronics manufacturing over the life cycle of the products studied. The objective of the model is to maximize the revenues generated from the final product and the recycled materials and to minimize the costs associated with the current assembly and future disassembly of the product. Besides the traditional production constraints such as production capacity, this model includes constraints that account for government regulations and recyclable product market limits.

Low and Williams (1996) presented a family of linear financial models for managing the various end-of-life options. The end-of-life alternatives considered are – resale, remanufacture, upgrade, recycling and scrap. The financial models serve to estimate the overhead cost elements (such as those associated with product recovery, transportation, packaging, disassembly etc.) associated with each of these options. These models help in better understanding the relative costs and benefits associated with each of these options.

Dewhurst (1993) also derived generic parametric equations to approximate the costs and benefits from disassembly operations, and an efficiency index for recycling.

Zussman *et al.* (1994) employed utility theory for assessment of end-of-life alternatives, which also takes into account the uncertainties in future economic and technical conditions. The authors considered three objectives in this evaluation:

1. profit maximization through selection of the best end-of-life option,
2. maximization of the number of reusable parts, and
3. minimization of the amount of landfill waste.

Relationships for evaluation of disposal costs and recovery benefits are developed and an example of a washing machine assembly is presented to illustrate the concepts.

Ishii *et al.* (1994) introduce the concept of 'clumps' for their analysis of design for product retirement (DFPR). A 'clump' is a collection of components/subassemblies that share common characteristics as per the designer's specifications. The clumps in a product can be recycled and reprocessed without further disassembly. The total retirement cost comprises of the collection and transportation cost, the disassembly cost and the total reprocessing cost of all the clumps in the product. The authors derive the expression for total system disassembly cost, which comprises of the costs associated with time to remove the fasteners and components, as well as the time required to reach for the removal or undo process.

Chen *et al.* (1993), Johnson and Wang (1995), McGlothin and Kroll (1995), and Shu and Flowers (1995) defined recycling cost functions that are linear with respect to the disassembly time. Chen *et al.* present a cost benefit analysis model of recycling. The cost function for recycling is defined as the total of disassembly cost, shredding cost, material recovery cost and dumping cost. The benefit of recycling is defined as the total of revenue from used parts, revenue from recovered materials and benefit of emission reduction from energy saving. Individual expressions for each of these costs are derived. The cost-benefit analysis method is illustrated by application of the model to the dashboard of a compact car.

Johnson and Wang (1995) present a methodology for optimal disassembly sequence generation for material recovery opportunities (MRO). MRO is a term used to represent recycling, remanufacture and reuse options at the end of product's life-cycle. They derive equations for profit-loss margin to decide further continuance of disassembly at each stage of the disassembly process. The authors consider various material value cost factors (such as total reprocessing/upgrade costs, the disposal state of the components, disposal fees and availability of recycling infrastructure) for quantitative assessment of the costs associated with recycling.

Mcglothin and Kroll (1995) describe a disassembly evaluation scheme for assessing the ease of disassembly. The method consists of preparing a worksheet called disassembly evaluation chart. The various column entries in the chart include:

1. Part number: The part number is used for identifying individual components/subassemblies.
2. Theoretical minimum number of parts: Each component is evaluated to determine whether it should exist as a separate part (because it has to move relative to other parts, or due to material incompatibility).

3. Number of repetitions: The number of times each disassembly task is performed is noted.
4. Task type: These include 16 different types of disassembly tasks such as cut, unscrew, push/pull etc.
5. Direction: This indicates coordinate axis along which the hand or the tool accesses the part.
6. Required tool: These are one of the 28 different tools commonly used in disassembly operations.

   Note: Each of the above information constitutes a column of the chart.
7. Columns 7-11 contain difficulty rating on a scale on 1-4 for a particular disassembly task. The difficulty due to accessibility, positioning, force and additional time is recorded. The first three have been discussed earlier in this section. The additional time accounts for any other time not included in the other three (such as removal of a long screw as compared to a shorter one). The last column records the total difficulty rating.

The ease of disassembly is evaluated by analyzing the entries in the worksheet. The authors illustrate the methodology by application to a computer monitor disassembly. Two separate papers (Kroll *et al.*, 1996 and Kroll and Hanft, 1998) outline the same approach and illustrate it through a hair dryer assembly and a computer CPU assembly respectively.

Shu and Flowers (1995) address the issue of selection of fastening and joining methods to facilitate remanufacturing. They present several case studies to illustrate the difficulties that may arise in remanufacturing due to certain fastening and joining methods. A linear cost model for estimating the life cycle costs is presented. The various costs considered include the first cost (production and assembly costs), the recycling cost, costs associated with failures during assembly and disassembly, maintenance costs and remanufacture cost.

Das *et al.* (2000) propose a disassembly effort index (DEI) to represent the total operating cost to disassemble a product. Based on their survey of a variety of disassembly facilities, the authors found that disassembly effort and cost were a function of several factors such as time, tools, fixture, access, instructions, hazard protection and force requirements. The DEI was computed using a multi-factor model that incorporated all these factors. The effort index was defined on a scale ranging from 0 to 100 and the range was assigned to each of the seven factors on a weighted basis. The DEI can be compared with the projected market value of disassembled parts and subassemblies to get an insight into the economic implications of disassembly.

2.4 Disassembly Optimization

After estimating the various costs associated with disassembly operations and disposal, as well as the benefits resulting from material recovery and recycling, the next step is to address the issue of disassembly optimization i.e. generating optimal disassembly sequences and determining the optimal depth of disassembly. A limited number of research studies have considered the disassembly optimization problem. The problem of optimal disassembly sequence generation has been proved to be NP-Complete (Moyer and Gupta, 1997). Consequently, the various research efforts are aimed at developing heuristic procedures for obtaining near-optimal solutions. Navin-Chandra (1994) represented final assembly in the form of a disassembly tree, with each node representing a component/subassembly. At each node of this tree, the following options are considered: dismantle further, send to shredder, sell, remanufacture and check for hazardous materials. In his first formulation, the author considers the revenues due to reuse, material recovery and shredding at each node. Of these three options, the one that provides maximum revenue is considered. For each node, the cost incurred to remove all the joints for reaching that node is also considered. The objective is to select the disassembly path and a particular node at which the net profit is maximized. The second formulation considers the disassembly problem as a traveling salesman problem (TSP). The analogy consists of considering each node as a city and the breaking of a joint is then similar to visiting a city. However, the problem differs from the traditional TSP in three respects:

1. it is not necessary to visit all the nodes in a feasible solution,
2. there is no need to return to the original node, and
3. besides cost, there are also revenues associated with visiting each node and hence the objective is to maximize the net benefit rather than minimizing the cost.

In order to account for these differences, the problem is formulated as a prize collecting traveling salesman problem (Balas, 1989) with simple variations. The solution methodology consists of exploring the disassembly tree in greedy fashion (i.e., exploring the nodes of maximal recovery first) through total enumeration. The major drawback of this scheme is that the set-up costs resulting from successive disassembly operations are not considered in the formulation. Also, since this problem is a restricted form of the prize collecting TSP, the algorithms that effectively solve the unrestricted case cannot be employed in this case.

Johnson and Wang (1995) also outline a similar search method that explores the disassembly tree to obtain an optimal sequence and stops at a node with maximum overall profitability. The disassembly tree is automatically generated through a software called EDIT (environmental design industrial template). At each node, the profit loss margin (PLM) values are calculated, which are obtained through cost-benefit analysis. The authors suggest a reduction in search space through clustering of compatible materials, clustering negative PLM value nodes for disposal, clustering similar disassembly operations, and maximizing parallelism and yield.

Penev and de Ron (1996) use AND/OR graph representation of product and use it as a basis for determining the optimal disassembly strategy. For the determination of a disassembly sequence, a dynamic programming approach is suggested. Each node in the AND/OR graph represents the state of the dismantled components. Each level in the hierarchical representation is a stage and arcs represent the disassembly operations required to arrive at the next stage. Since the net revenue of a state does not depend on the disassembly operations used to arrive at the previous state (and hence the principal of optimality holds), a state of the next stage is chosen so as to maximize the net revenue. The authors provide no rigorous mathematical formulation and practical illustration to validate their approach. For determination of optimal disassembly level, the authors consider various possible end-of-life options such as reuse, recycling, shredding and disposal and the costs and benefits associated with them. Again, as in previous approaches, a search scheme based on total enumeration is suggested. The method is illustrated through disassembly of a bearing unit. The main drawback of this scheme is that the interrelationship between disassembly sequences and costs is ignored; the problems of determining the optimal disassembly sequence and disassembly depth are segregated, which may not lead to an overall optimal disassembly strategy.

Kuo *et al.* (2000) present a graph based heuristic approach for obtaining the disassembly tree, which can be subsequently used for disassembly sequence generation. The scheme consists of the following steps:

1. At first, a component fastener graph for the given product is obtained, with components represented by the nodes while fasteners by the edges. It must be noted that this graph is different from the ones discussed previously in that it does not have subassemblies as its nodes, but consists of individual components only. An adjacency matrix is prepared based on this graph. An element m_{ij} of this matrix is equal to 1 if the component i is connected to component j, and 0 otherwise.
2. The next step consists of modularity analysis via cut-vertex search. This step consists of decomposing the final assembly into modular subassemblies to facilitate disassembly. These subassemblies comprise of components with common characteristics or close physical relationship. A standard cut-vertex

algorithm is used to determine the cut-vertices of the component-fastener graph and obtain the modular subassemblies.

3. The third step involves disassembly precedence analysis. For this purpose, a disassembly precedence matrix is prepared for each of the six possible coordinate directions. In a given direction, if component i cannot be disassembled and removed before component j, then the element p_{ij} of the precedence matrix is 0 for that particular direction.
4. The disassembly tree is generated by combining the results from previous steps to obtain a disassembly constraint matrix. The disassembly constraint matrix provides information about directions in which the modular subassemblies can be removed and establishes an ordering in which these subassemblies have to be removed.

The authors have developed a computer-based tool for disassembly ability analysis from the recycling point of view. The details of the tool and the underlying methodology are not discussed. The approach is aimed at analyzing electromechanical devices with complex product structure.

Lambert (1999) proposed a linear programming approach for disassembly/cluster sequence generation. The author first addresses the issue of sequence generation in case of complete disassembly. The proposed product data representation is again a graphical representation as in several of the previous cases. The graph consists of subassemblies as nodes aligned from left to right in descending order of the number of individual components they comprise of. Therefore, the final assembly is at the leftmost end of the graph. The individual components should be at the rightmost end, but the proposed scheme does not require their inclusion in the graph. Two sets S (of subassemblies) and A (of actions, or disassembly operations) are defined. Individual components are numbered starting from 1, and a subassembly is numbered in the form a/b, if the subassembly consists of components from numbers a through b. A flow variable x_i is assigned to each action i, with $x_i = 1$, if an action occurs and $x_i = 0$, otherwise. The variable x_i can be defined as a continuous variable between 0 and 1, since the calculation always results in a basic solution. For each node in the graph, the constraints are defined in the following manner:

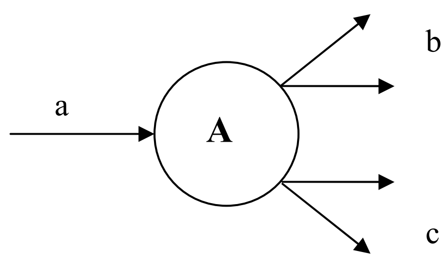

Fig 2.1 – Node Constraints for LP approach to disassembly sequence generation

Consider the node A as shown above, with arc **a** incident on it. It is possible to disassemble the node in two different ways, denoted by actions **b** and **c** respectively. For this node, the node equation is given by:

$$x_b + x_c \leq x_a$$

Similar node equations are obtained for each individual node. The objective function is to maximize the total net revenue over all the nodes. The formulation of model for sequence generation in case of clusters (collection of components with similar characteristics) is obtained similarly. However, such a model assumes a prior knowledge of the fact that it is beneficial to retrieve such clusters without further disassembly. Thus, the model is actually based on the assumption that the depth of disassembly is known a priori.

2.5 Concluding Remarks

In this chapter, we provided a comprehensive overview of pertinent literature in the fields of product data modeling, economic aspects of product disassembly and disassembly optimization. We have found that mathematical programming techniques have not been widely utilized in disassembly optimization. Furthermore, there are no rigorous mathematical models that address the issue of sequence dependent set-up costs in disassembly optimization. In the subsequent chapters, we develop integer-programming models to incorporate the sequence dependent set-up costs and develop a solution procedure to determine disassembly sequence and depth of disassembly.

CHAPTER 3

MODELING FOR DISASSEMBLY OPTIMIZATION

3.1 Introduction

This chapter is organized along the following lines. In the section 3.2, we provide details of our network representation that captures the mating relationships between the components/subassemblies of the product. The network representation also provides information about the joints/fasteners, which must be removed in order a retrieve a particular component/subassembly. In section 3.3, we provide models for sequence generation in case of complete disassembly. We show that the problem of minimizing sequence dependent set-up costs in case of complete disassembly is equivalent to asymmetric traveling salesman problem with precedence constraints (also referred to as sequential ordering problem or SOP). We discuss three models - the first one (DSGM1) is available in the literature; we have developed the second (DSGM2) and the third model (DSGM-RLT) during the course of this research. We again emphasize that the objective of our research is to develop disassembly plans that not only provide an optimal sequence for breaking the joints, but also help to determine the optimal depth of disassembly. The models discussed in 3.3 assume complete disassembly; they do not serve our objective of determining an optimal depth of disassembly. We present these models to emphasize that our problem resembles the SOP and we'll use this fact in the development of our models in section 3.4 and again in the development of our solution methodology in chapter four. In section 3.4, we develop two different 0-1 integer-programming models for disassembly optimization. As mentioned in chapter 1, the objective of disassembly optimization is to determine the sequence of disassembly operations as well as to determine an optimal depth of disassembly such that recovery value of components/subassemblies is maximized, while the costs associated with breaking the joints and sequence dependent set-up are minimized. In section 3.5, we present new formulations for ATSP and PCATSP and establish their validity. In the last section, we present a case study for generating optimal disassembly plan for a computer processor using our disassembly optimization models.

3.2 Network Representation of Product Assembly

3.2.1 Description of the Network Representation

The development of network representation of a product assembly assumes a prior knowledge of product's bill of material (BOM) and the set of joints and fasteners (such as threading, welding, snap fit, screws, rivets etc.) that hold the parts together. Only those components/subassemblies, which are identified in the product's BOM, form the nodes of our network.

The network representation of the disassembly operations starts from the final assembly and explodes the part BOMs in a tree-like fashion until the leaves are reached. The representation consists of three sets of nodes:

1. nodes that represent parts or subassemblies,
2. nodes that represent joints or fasteners, and
3. dummy nodes that represent terminal disassembly points.

The following two sets of directed arcs are present in our representation:

1. arcs that define the set of joints to be broken to retrieve a part or subassembly, and
2. arcs between the joints or components at the same level, which specify a precedence relationship between particular joints or components.

We illustrate the development of network representation through a small example of a roller assembly (This example is from Subramani and Dewhurst (1991)). Figure 3.1 depicts a roller assembly comprising of six components. Parts L1 and L3 are welded on part L2 through joints (welds) W1 and W2, respectively. The resulting subassembly is identified as part R1. Two through-holes on parts H1 and H2 support R1. H1 and H2 are threaded on the base P1 by joints (screws) S2, S3 (H1) and S1 (H2).

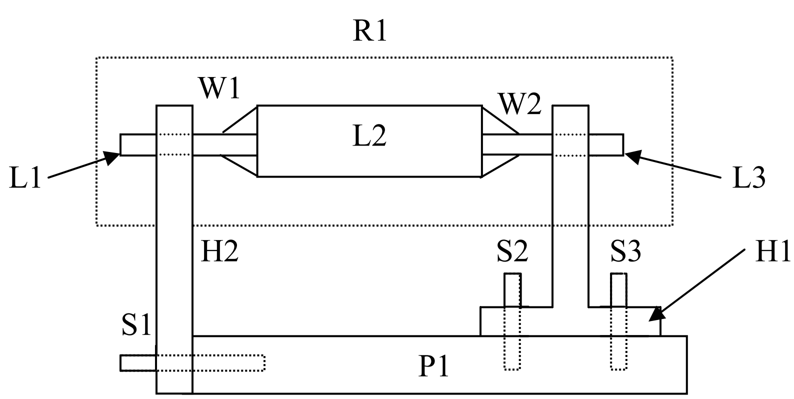

Fig 3.1 - Example of Roller Assembly

The network representation for this roller assembly is shown in Fig. 3.2. The parts/subassemblies and terminal nodes are represented as circular/oval nodes, while the joints and fasteners are represented as rectangular nodes. The dashed arrows in the figure indicate that parts H1 and H2 must be removed before R1 can be recovered.

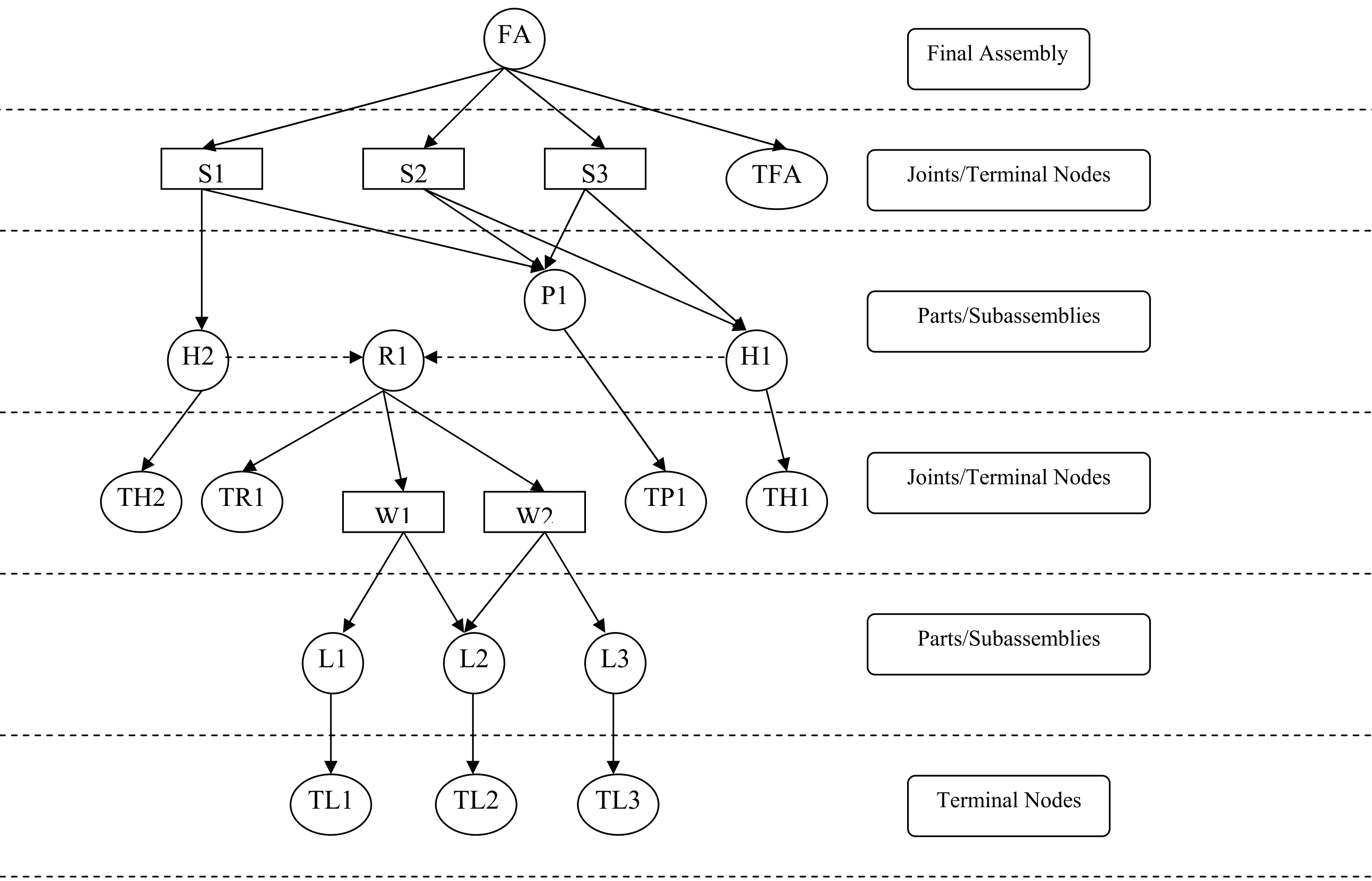

Figure 3.2 – Network Representation of the Roller Assembly

3.2.2 Usefulness of the Network Representation

The network representation scheme described in the previous section offers several advantages over those available in literature. These include:

- Our network representation is based on the information derived directly from the product's bill of material (BOM) and the assembly plans (AP). Since information regarding a product's BOM and AP can be readily extracted from the appropriate MRP databases, it is easy to develop this network representation. In the development of automated systems for generating disassembly plans, such a representation is of great practical significance since such systems can be directly linked to the MRP database. As pointed out in section 2.2, several popular representation schemes such as AND/OR graphs do not necessarily have subassemblies included in BOM as their nodes.
- Our scheme not only shows the mating relationships between components and subassemblies, but also includes the joints and fasteners that hold these parts/subassemblies together. This is a very useful feature in disassembly planning, since it provides information about the joints that must be removed to retrieve a particular part/subassembly and helps in the development of rigorous mathematical models. The LINKER representation due to Ishii *et al.* (1994) also separates fasteners from parts and subassemblies; however, the scheme does not follow the hierarchical nature of BOMs and as in the case of AND/OR graphs, the subassemblies included may not correspond to the subassemblies in a product's BOM.

3.3 Modeling for Sequence Generation in Case of Complete Disassembly

3.3.1 Introduction to PCATSP

The precedence constrained asymmetric traveling salesman problem (also known as Sequential Ordering Problem or SOP) was first introduced by Escuredo (1988). The SOP can be stated as follows: Given a directed graph comprising of *n* nodes, the corresponding edge weights associated with the arcs, and the precedence relationships among the nodes, find a minimum weight Hamiltonian path comprising of all *n* nodes such that the precedence relationships are satisfied. Each precedence relationship requires that a certain node *i* be visited before a certain node *j* in a feasible solution. Ascheuer *et al* (1990) described the problem in graph theoretic terminology as follows: Given a complete digraph (V, A_n) of *n* nodes with edge weights (or costs) c_{ij} for all $(i, j) \in A_n$ and a transitively closed acyclic digraph (V, R), find a feasible Hamiltonian path in (V, A_n) that has minimum cost. The transitively closed acyclic graph represents the precedence relationships among the nodes in *V*. The graph is transitively closed because if node *i* precedes node *j* and node *j* precedes node *k*, then node *i* must precede node *k*.

The model for SOP is similar to that of asymmetric traveling salesman problem (ATSP) with additional precedence constraints. In the next section, a model of SOP, that is discussed in the context of the disassembly sequence generation problem, is presented.

3.3.2 Development of the Model DSGM1

The following notation is used for developing the model DSGM1, which represents the disassembly sequence generation problem:

K – set of joints to be broken

KPM_j – set of joints, which must be broken before joint j (preceding joints)

KSM_j – set of joints, which must be broken after joint j (succeeding joints)

s_{ij} – set-up cost associated with breaking joint j immediately after joint i

F – first joint to be broken in the sequence

L – last joint to be broken in the sequence

Note that if F and L are not known in advance, dummy joints can be used instead.

The binary decision variable p_{ij} is defined in the following manner:

$$p_{ij} = \begin{cases} 1, \text{ if joint } j \text{ is broken immediately after joint } i \\ 0, \text{ otherwise} \end{cases}$$

The problem can be represented in the form of a graph in the following manner:

- The joints in set K represent the nodes of the graph. The arc set A represents all possible arcs between the pairs of nodes. Hence, a complete digraph (K, A) is obtained.
- The precedence relationships between the joints are specified by an acyclic transitively closed precedence graph (K, R), where the arc set R defines the precedence relationships among the joints. This graph is transitively closed because if a joint i precedes joint j and a joint j precedes joint k, then joint i must precede joint k.

- The arc set $A\ (W) = \{(i, j) \in A\colon i, j \in W\}$, where $W \subseteq K$
- If G is a subset of A, then the sum $\sum_{(i,j)\in G} p_{ij}$ is abbreviated as $p(G)$
- Also, $(j{:}W) = \{(j, k) \in A\colon k \in W\}$ and $(W{:}j) = \{(i, j) \in A\colon i \in W\}$

The model for disassembly sequence generation is presented below:

Minimize $\sum_{i\in K} \sum_{\substack{j\in K \\ j\neq i}} s_{ij} p_{ij}$

subject to:

$$\sum_{\substack{j\in K \\ j\notin KPM_i \\ j\neq i,F}} p_{ij} = 1, \forall i \in K : i \neq L \tag{3.1a}$$

$$\sum_{\substack{i\in K \\ i\notin KSM_j \\ i\neq j,L}} p_{ij} = 1, \forall j \in K : j \neq F \tag{3.1b}$$

$$p_{ij} = 0, \forall i \in K, \forall j \in (KPM_i \cup \{i\}) \tag{3.1c}$$

$$p_{ij} = 0, \forall j \in K, \forall i \in (KSM_i \cup \{j\}) \tag{3.1d}$$

$$u_i - u_j + |K| p_{ij} \leq |K| - 1, \forall i, \forall j \in K : j \neq i \tag{3.1e}$$

$$p(j:W) + p(A(W)) + p(W:i) \leq |W|, \forall (i,j) \in R, \forall W \subseteq K \setminus \{i,j\} : W \neq \phi \tag{3.1f}$$

$$p_{ij} \in \{0,1\}, \forall i \in K, \forall j \in K \quad (3.1g)$$

Constraints (3.1a) ensure that after every joint $i \in K$ (except the last joint L), exactly one joint is broken and this joint should not belong to the set of joints which must precede i (i.e. KPM_i).

Constraints (3.1b) ensure that before every joint $j \in K$ (except the first joint F), exactly one joint is broken and this joint should not belong to the set of joints, which must succeed j (i.e. KSM_j).

Constraint (3.1c) and (3.1d) are variable-fixing constraints which ensure that if a joint pair (i, j) have a predecessor-successor relationship, then the corresponding decision variable p_{ji} should be 0.

Constraints (3.1e) are well-known Miller-Tucker-Zemlin subtour elimination constraints.

Constraints (3.1f) are due to Ascheuer *et al* (1990) and are known as the *precedence forcing constraints*. These constraints ensure that the precedence relationships between the joints are not violated. The number of precedence forcing constraints is exponential in the size of the problem.

Finally, constraints (3.1g) restrict the decision variables p_{ij} to be binary.

3.3.3 Development of the model DSGM2

In this section, an alternative model for disassembly sequence generation is presented. Except for the decision variable, the notation is the same as in the model discussed in the previous section. In this case, we define the decision variable as follows:

$$z_{kn} = \begin{cases} 1, \text{ if joint } k \text{ is removed in the n-th position of the disassembly sequence.} \\ 0, \text{ otherwise} \end{cases}$$

The model for disassembly sequence generation is presented below:

Minimize $\sum_{k \in K} \sum_{\substack{l \in K \\ l \neq k}} \sum_{n=1}^{|K|-1} s_{kl} z_{kn} z_{l,n+1}$

subject to:

$$\sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} z_{kn} = 1, \forall k \in K \qquad (3.2a)$$

$$\sum_{k \in K} z_{kn} = 1, \forall n = 1,....,|K| \qquad (3.2b)$$

$$z_{kn} = 0, \forall k \in K, \forall n = 1,....,|KPM_k| \qquad (3.2c)$$

$$z_{kn} = 0, \forall k \in K, \forall n = (|K| - |KSM_k| + 1),....,|K| \qquad (3.2d)$$

$$z_{kn} \le \sum_{m=1}^{n-1} z_{lm}, \forall k \in K, \forall l \in KPM_k, \forall n = 2,....,|K| \tag{3.2e}$$

$$z_{kn} \le \sum_{m=n+1}^{|K|} z_{lm}, \forall k \in K, \forall l \in KSM_k, \forall n = 1,....,(|K|-1) \tag{3.2f}$$

$$z_{kn} \in \{0,1\}, \forall k \in K, \forall n = 1,....,|K| \tag{3.2g}$$

The objective function in the above model is obtained as follows: if the joint *k* is broken n-th in the disassembly sequence (z_{kn}=1) and the joint *l* is broken at (n+1)th position ($z_{l,n+1}$=1), then the associated set-up cost (s_{kl}) is included in the computation of total set-up cost. The objective is to obtain a sequence in which the joints must be broken to minimize the total set-up cost.

Constraints (3.2a) ensure that each joint is broken at exactly one position in the sequence. The number of possible positions at which a particular joint can be broken is further constrained by observing that for each joint *k*, all its predecessors (joints in set KPM_k) must be broken before that joint, hence the joint *k* cannot be broken at any of the positions from 1 to $|KPM_k|$ (note that constraints (3.2c) enforce this requirement). Also, for each joint *k*, its successors (joints in set KSM_k) can be broken only after the joint *k* is broken. Hence, joint *k* cannot be broken at any of the last $|KSM_k|$ positions (note that constraints (3.2d) force this requirement). Consequently, for each joint *k*, the only available positions are from ($|KPM_k|+1$) to ($|K|-|KSM_k|$) and it must be broken at one of these positions.

Constraints (3.2b) ensure that at each position from 1 to $|K|$, exactly one joint is broken.

Constraints (3.2e) and (3.2f) are the precedence forcing constraints, which ensure the feasibility of joint sequence by precluding possible precedence violations. Constraints (3.2e) ensure that if joint *k* is broken at position n, its predecessor joints (joints in set KPM_k) must be broken at one of the positions from 1 to (n-1). Similarly, constraints (3.2f) ensure that if joint *k* is broken at position n, then its successors joints (joints in set KSM_k) must be broken at one of the positions from (n+1) to $|K|$.

Finally, constraints (3.2g) enforce the decision variables z_{kn} to be binary.

The non-linearity of the objective function complicates the above problem, making it difficult to solve using the well-known methods for linear integer programs. However, it can be linearized in the following manner:

A new binary variable $w_{k,l,n}$ is introduced and is defined as follows:

$$w_{k,l,n} = \begin{cases} 1, \text{ if joint } k \text{ is broken at the n-th and joint } l \text{ at the (n+1) th position of the disassembly sequence.} \\ 0, \text{ otherwise} \end{cases}$$

The product $z_{kn} z_{l,n+1}$ in the objective function is replaced by this new variable $w_{k,l,n}$. That is,

$$z_{kn} z_{l,n+1} = w_{k,l,n}.$$

It is clear that $w_{k,l,n}$ should be 1 only when both z_{kn} and $z_{l,n+1}$ are 1 and should be 0, otherwise. This restriction can be enforced through the following set of inequalities (Sherali and Adams, 1999) :

$$z_{kn} \geq w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,...,|K| \quad (3.2h)$$

$$z_{l,n+1} \geq w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,...,(|K|-1) \quad (3.2i)$$

$$z_{kn} + z_{l,n+1} - 1 <= w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,....,(|K|-1) \quad (3.2j)$$

Thus, a linear 0-1 integer-programming model can be obtained by incorporating the above constraints in the model.

Next, we introduce several equality constraints, which might tighten the linear programming relaxation of the formulation described above and help in achieving a tighter lower bound on the objective function value. It must be noted that constraints (3.2a) to (3.2j) described previously represent the disassembly sequence generation problem completely; however, the following constraints might curtail the solution space of the linear programming relaxation without affecting the integer programming solution.

$$\sum_{\substack{l \in K \\ l \neq k}} \sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} w_{k,l,n} = 1, \forall k \in K : k \neq L \qquad (3.2k)$$

$$\sum_{\substack{k \in K \\ k \neq l, L}} \sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} w_{k,l,n} = 1, \forall l \in K : l \neq F \qquad (3.2l)$$

$$\sum_{\substack{k \in K \\ k \neq L}} \sum_{\substack{l \in K \\ l \neq k}} w_{k,l,n} = 1, \forall n = 1,....,(|K|-1) \qquad (3.2m)$$

$$w_{k,l,n} = 0, \forall k \in K : k \neq F, \forall l \in K, \forall n = 1,....,|KPM_k| \qquad (3.2n)$$

$$w_{k,l,n} = 0, \forall k \in K, \forall l \in K, \forall n = (|K| - |KSM_k| + 1),....,(|K|-1) \qquad (3.2p)$$

$$w_{k,l,n} = 0, \forall k \in K, \forall l \in (KPM_k \cup \{k\}), \forall n = (|KPM_k| + 1),....,(|K| - |KSM_k|) \qquad (3.2q)$$

$$\sum_{\substack{k \in K \\ k \neq l, F}} w_{k,l,n} = \sum_{\substack{m \in K \\ m \neq l, I}} w_{l,m,n+1}, \forall l \in K : l \neq F, L, \forall n = 1,....,(|K|-2) \qquad (3.2r)$$

We now provide an explanation of constraints (3.2k) to (3.2r), also mentioning alongside how the constraints (3.2a) to (3.2j) actually enforce the restrictions implied by these constraints.

Constraints (3.2k) ensure that each joint $k \in K$ (except the last joint L) is immediately succeeded by exactly one joint. Similarly, constraints (3.2*l*) ensure that each joint $k \in K$ (except the first joint F) is preceded by exactly one joint in the disassembly sequence. Furthermore, they also ensure that each joint is broken at exactly one position in the sequence and the number of feasible positions at which a particular joint is broken range from ($|KPM_k|$+1) to ($|K|$-$|KSM_k|$). As explained earlier, this restriction is imposed by constraints (3.2a). Constraints (3.2n) and (3.2p) ensure that a joint is not broken at a position other than the feasible one. These restrictions also follow from constraints (3.2c) and (3.2d) described previously.

Constraints (3.2m) ensure that at each position in the sequence, exactly one joint is broken. This restriction is imposed by constraints (3.2b).

Constraints (3.2q) ensure that for each joint $k \in K$, the immediately succeeding joint is not the one that must precede k. The precedence forcing constraints (3.2e) and (3.2f) incorporate these restrictions.

Constraints (3.2r) imply that if a joint k is broken at position n and is immediately succeeded by joint l (broken at position (n+1)), then joint l must be immediately succeeded by a joint m broken at position (n+2), unless joint l is the last joint in the sequence. That is, if $w_{k,l,n}$ is equal to 1 for some joint pair (k, l), then $w_{l,m,n+1}$ must be 1 for some $m \in K$. These constraints ensure that there is no break in the sequence of joints from position 1 to $|K|$. This condition is imposed by constraints (3.2a) and (3.2b). Constraints (3.2a) ensure that each joint is broken at exactly one position in the sequence, while constraints (3.2b) imply that in each position of the sequence, exactly one joint is broken. Consequently, the possibility of a break in the joint sequence is precluded.

3.3.4 A formulation of DSGM2 using the Reformulation-Linearization Technique (DSGM-RLT)

In this section, we present an alternative formulation of disassembly sequence generation problem, which is obtained by using the reformulation-linearization technique (RLT) of Sherali and Adams (1999).

For the application of RLT to DSGM2, we multiply each of the constraints (3.2b) (except when $k = L$) with $z_{l,n+1}$ for all $l \in K{:}l \notin (KPM_k \cup \{k\})$ and for all $n = (|KPM_k|+1),\ldots,(|K| - |KSM_k|)$. Thus, we obtain the following constraints:

$$\sum_{\substack{k \in K \\ k \neq L}} z_{kn} z_{l,n+1} = z_{l,n+1}, \forall n = (|KPM_k|+1),\ldots,(|K|-|KSM_k|), \forall l \in K : l \notin (KPM_k \cup \{k\}) \quad (3.2s)$$

Now, we can replace the product term $z_{kn}z_{l,n+1}$ with the binary variable $w_{k,l,n}$ in both the objective function and the constraints (3.2s). Note that the variable $w_{k,l,n}$ is defined in the same manner as in DSGM2. The constraints (3.2s) can then be rewritten as:

$$\sum_{\substack{k \in K \\ k \neq L}} w_{k,l,n} = z_{l,n+1}, \forall n = (|KPM_k|+1),....,(|K|-|KSM_k|), \forall l \in K : l \notin (KPM_k \cup \{k\}) \qquad (3.2t)$$

The model DSGM-RLT comprises of constraints (3.2a) through (3.2k) alongwith constraints (3.2q), (3.2r) and (3.2t) and the same objective function as in DSGM2 with the product term replaced by $w_{k,l,n}$. The complete formulation is presented below:

Minimize $\sum_{k \in K} \sum_{\substack{l \in K \\ l \neq k}} \sum_{n=1}^{|K|-1} s_{kl} w_{k,l,n}$

subject to:

$$\sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} z_{kn} = 1, \forall k \in K \qquad (3.2a)$$

$$\sum_{k \in K} z_{kn} = 1, \forall n = 1,....,|K| \qquad (3.2b)$$

$$z_{kn} = 0, \forall k \in K, \forall n = 1,....,|KPM_k| \qquad (3.2c)$$

$$z_{kn} = 0, \forall k \in K, \forall n = (|K|-|KSM_k|+1),....,|K| \qquad (3.2d)$$

$$z_{kn} \leq \sum_{m=1}^{n-1} z_{lm}, \forall k \in K, \forall l \in KPM_k, \forall n = 2,....,|K| \tag{3.2e}$$

$$z_{kn} \leq \sum_{m=n+1}^{|K|} z_{lm}, \forall k \in K, \forall l \in KSM_k, \forall n = 1,....,(|K|-1) \tag{3.2f}$$

$$z_{kn} \in \{0,1\}, \forall k \in K, \forall n = 1,....,|K| \tag{3.2g}$$

$$z_{kn} \geq w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,...,|K| \tag{3.2h}$$

$$z_{l,n+1} \geq w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,...,(|K|-1) \tag{3.2i}$$

$$z_{kn} + z_{l,n+1} - 1 <= w_{k,l,n}, \forall k \in K, \forall l \in K : l \neq k, \forall n = 1,....,(|K|-1) \tag{3.2j}$$

$$\sum_{\substack{l \in K \\ l \neq k}} \sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} w_{k,l,n} = 1, \forall k \in K : k \neq L \tag{3.2k}$$

$$w_{k,l,n} = 0, \forall k \in K, \forall l \in (KPM_k \cup \{k\}), \forall n = (|KPM_k|+1),....,(|K|-|KSM_k|) \tag{3.2q}$$

$$\sum_{\substack{k \in K \\ k \neq l,F}} w_{k,l,n} = \sum_{\substack{m \in K \\ m \neq l,I}} w_{l,m,n+1}, \forall l \in K : l \neq F, L, \forall n = 1,...., (|K|-2) \tag{3.2r}$$

$$\sum_{\substack{k \in K \\ k \neq L}} w_{k,l,n} = z_{l,n+1}, \forall n = (|KPM_k| + 1), \ldots, (|K| - |KSM_k|), \forall l \in K : l \notin (KPM_k \cup \{k\}) \qquad (3.2t)$$

The constraints (3.2t) imply the constraints (3.2*l*) to (3.2p) of the model DSGM2.

Surrogating constraints (3.2t) over all *n*, for all $l \in K$, we get:

$$\sum_{\substack{k \in K \\ k \neq l, L}} \sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} w_{k,l,n} = 1, \forall l \in K : l \neq F,$$

which is the same as constraints (3.2*l*).

Similarly, surrogating (3.2t) over all *l*, for each *n*, we get:

$$\sum_{\substack{k \in K \\ k \neq L}} \sum_{\substack{l \in K \\ l \neq k}} w_{k,l,n} = 1, \forall n = 1, \ldots, (|K| - 1),$$

which is same as (3.2m).

Also, (3.2c) and (3.2d) force the joints to be broken in a feasible position in the sequence. Hence, $z_{l,n+1}$ will be 0 for all *n* other than the feasible ones. Consequently, (3.2t) imply that for all positions other than the feasible ones, we have:

$$\sum_{\substack{k \in K \\ k \neq l}} w_{k,l,n} = 0, \forall l \in K, \forall n = 1, \ldots, |KPM_k|,$$

and

$$\sum_{\substack{k \in K \\ k \neq l}} w_{k,l,n} = 0, \forall l \in K, \forall n = (|K| - |KSM_k| + 1), \ldots, (|K| - 1)$$

Since the summation of w variables is 0, it implies that each individual variable will be 0 for all $k \in K$, which is the condition enforced by (3.2n) and (3.2p).

Hence, constraints (3.2t) subsume all the constraints (3.2l) to (3.2p) of formulation DSGM2.

3.4 Models for Disassembly Optimization

The breaking of joints within a final assembly is a complex task that requires both human labor as well as automated processes. The sequence of breaking the joints is critical, since sequence dependent set-up costs may form a major component of the total disassembly cost. For example, if two screws and a weld are to be removed, the total cost of removing the screws first and then the weld is obviously smaller than the cost of removing the weld in between the screws. This property is similar to sequence dependent set-up times encountered in scheduling problems (Das and Sarin, 1994 and Sherali *et al*, 1990) and severely complicates the problem of determining optimal disassembly sequence. We effectively capture the effect of sequence dependent set-up in our formulations. To that end, we use the following notation:

Let the network representation of the product be denoted by a digraph $G=(K, A)$, where K is the set of joints and A is the set of directed arcs in the network. Let I be the set of component indices (individual parts and subassemblies) in the final assembly. Let f_k be the cost of breaking the joint k and s_{kl} be the set-up cost when disassembly operations related to joints $(k, l) \in K$ are performed in sequence. Let b_i be the composite recovered value of part i, and c_i be the disposal cost of part i.

Let x_i model the depth of disassembly i.e.

$$x_i = \begin{cases} 1, \text{ if disassembly stops at node } T_i \\ 0, \text{ otherwise} \end{cases}$$

(Here T_i is the terminal node for part $i \in I$)

Also, we define a variable y_k for each joint $k \in K$ in the following manner:

$$y_k = \begin{cases} 1, \text{ if joint } k \text{ is broken} \\ 0, \text{ otherwise} \end{cases}$$

Next, we define the sequencing variable p_{kl} in the following manner:

$$p_{kl} = \begin{cases} 1, \text{ if joint } l \text{ is broken immediately after joint } k \\ 0, \text{ otherwise} \end{cases}$$

Also, we define the following:

P_i – set of parts at the BOM level of part i that have to be removed before i can be recovered

SOS – set of subassemblies (including the complete final assembly)

SOJ_i – set of joints within subassembly i

KP_i – set of preceding joints of part i

KS_i – set of succeeding joints of part i

KPM_k – set of preceding joints of joint k

KSM_k – set of succeeding joints of joint k

F – dummy joint, which precedes the removal of all the joints (this joint is always broken irrespective of the status of other joints)

L – dummy joint, which succeeds the removal of all the joints (if all the joints are not broken, then this dummy joint is also not broken). We use the dummy joint L in our first model, which we call DOM1.

D – dummy joint, which succeeds the removal of every other joint *that is broken* (joint D is always broken in a feasible solution and is the last joint in the sequence). We use joint D in our second optimization model, which we call DOM2.

For all the dummy joints used in our model, we set the cost of breaking the joint equal to zero. Also, we set the sequence dependent set-up of dummy joints with respect to every other joint equal to zero.

Finally, we use the binary variable z_{kn} in our first model (DOM1) which is defined as follows:

$$z_{kn} = \begin{cases} 1, \text{ if joint } k \text{ is removed in the n-th position of the} \\ \quad \text{disassembly sequence} \\ 0, \text{ otherwise} \end{cases}$$

We use the binary variable q_{ij} in our second model. It is used to enforce precedence and subtour elimination constraints, as explained later. Before proceeding further, we first make the following important observation:

Observation: *In the disassembly optimization problem, not only do we want to eliminate the subtours from a solution, we also do not want the complete tour unlike a traveling salesman problem i.e we do not want that the last joint broken in the sequence be succeeded by the first joint. This is because the sequence of joints to be broken in different products is different and will not be repeated. Even in similar products (such as two computers of same configuration), it is reasonable to assume that the sequence in which the joints are broken will not be repeated because the net recovery values of individual parts/subassemblies in each of the products are likely to be different, and consequently, the joints that will be broken and their sequence will be different. In the absence of the repetition of a sequence in which the joints of a product are broken, it is clear that we should prevent the occurrence of a complete tour in the solution. From a disassembly point of view, each product is unique and should be treated as such.*

In the sequel, wherever we refer to subtours, we imply that a complete tour is also included, unless stated otherwise.

Next, we formulate the disassembly optimization models DOM1 and DOM2 in the following sections.

3.4.1 Disassembly Optimization Model DOM1

Maximize $\sum_{i \in I}(b_i - c_i)x_i$ - $\sum_{k \in K} f_k y_k$ - $\sum_{k \in K}\sum_{\substack{l \in K \\ L \neq k}} s_{kl} p_{kl}$

Subject to:

$$x_i + y_k \leq 1, \forall i \in I, k \in SOJ_i \quad (3.3a)$$

$$x_i \geq y_k, \forall i \in I, k \in KS_i \quad (3.3b)$$

$$x_i \leq y_k, \forall i \in I, k \in KP_i \quad (3.3c)$$

$$x_i \geq \sum_{k \in KP_i} y_k - |KP_i| + 1, \forall i \in I : i \notin SOS, |P_i| = 0 \quad (3.3d)$$

$$x_i \geq \sum_{j \in P_i} x_j - |P_i| + 1, \forall i \in I : i \notin SOS, |P_i| \neq 0 \quad (3.3e)$$

$$\sum_{n=|KPM_k|+1}^{|K|-|KSM_k|} z_{kn} = y_k, \forall k \in K \quad (3.3f)$$

$$z_{kn} = 0, \forall k \in K : k \neq F, \forall n = 1,..,|KPM_k| \quad (3.3g)$$

$$z_{kn} = 0, \forall k \in K : k \neq L, \forall n = (|K| - |KSM_k| + 1),..,|K| \quad (3.3h)$$

$$\sum_{k \in K} z_{kn} \leq 1, \forall n = 1,..,|K| \quad (3.3i)$$

$$z_{kn} \leq \sum_{m=1}^{n-1} z_{lm}, \forall k \in K, l \in KPM_k, n = 2,..,|K| \quad (3.3j)$$

$$\sum_{l \in (K - KPM_k)} p_{kl} \leq y_k, \forall k \in K : k \neq L \quad (3.3k)$$

$$\sum_{k \in (K - KSM_l)} p_{kl} = y_l, \forall l \in K : l \neq F \quad (3.3l)$$

$$z_{kn} + \sum_{m=n+1}^{|K|} z_{lm} + p_{lk} \leq 2, \forall k \in K, l \in K, n = 1,..,(|K| - 1) \quad (3.3m)$$

$$p_{kl} = 0, \forall k \in K, l \in (KPM_k \cup \{k\}) \quad (3.3n)$$

$$p_{ki} = 0, \forall i \in K, j \in KPM_i, k \in KPM_j \quad (3.3p)$$

$$p_{iF} = 0, \forall i \in K \quad (3.3q)$$

The objective function represents the best trade-off between disassembly and disposal costs and benefits from the recovered components and subassemblies.

Constraints (3.3a) prohibit the breaking of joints within a subassembly i (i.e. joints $k \in SOJ_i$), if T_i is the terminal disassembly node. Note that set SOJ_i will be empty for all individual parts i.

Constraints (3.3b) are *accessibility* constraints, which ensure that the succeeding joints of a part i (i.e. $k \in KS_i$) cannot be broken unless that part is recovered. Note that for a given subassembly i, the set of succeeding joints does *not* include the joints within the subassembly (i.e. joints $k \in SOJ_i$).

Constraints (3.3c) state that if disassembly stops at node T_i, i.e. if $x_i = 1$, then all preceding joints $k \in KP_i$ have to be broken. Both constraints (3.3a) and (3.3b) are inactive if $x_i = 0$.

Constraint (3.3d) ensures that for a given part $i \in I$, which does not have any predecessor parts at its BOM level (i.e., set P_i is empty), if all the preceding joints $k \in KP_i$ are broken, then $x_i = 1$ necessarily, except when i is a subassembly (i.e., $i \in SOS$), since a subassembly can be broken down further into its component parts and hence, x_i can possibly be equal to zero.

Constraint (3.3e) ensures that for all parts $i \in I$, such that i is not a subassembly, and $|P_i| \neq 0$, if all the parts in P_i are recovered, then i is also recovered (i.e., $x_i = 1$). (Note that for a part $i \in I$, such that $|P_i| \neq 0$, the set $KP_i = \bigcup_{j \in P_i} KP_j$. Hence, instead of separate constraints (4) and (5), constraint (4) (without the restriction that $|P_i| = 0$) should suffice. However, we present them separately here to distinguish those parts which have other preceding parts at their BOM level from the other parts which do not have such precedence relationships).

Constraints (3.3f) are necessary to ensure that if a joint k is broken, it is broken at exactly one position in the sequence; otherwise, if the joint is not broken, then the associated sequencing variable z_{kn} is zero for all possible values of n. Also, since a joint k can be broken only after its predecessors have been broken, and before its successors are broken, we only need that the above stated condition is true for values of n in the range ($|KPM_K|+1$) to ($|K|-|KSM_k|$). For other values of n, z_{kn} is equal to zero and this restriction is imposed by constraints (3.3g) and (3.3h).

Constraints (3.3i) ensure that at a given position in the disassembly sequence, no more than one joint is broken.

Constraints (3.3j) are the precedence forcing constraints. They enforce the restriction that if a joint k is broken at a position n in the disassembly sequence, then all its preceding joints must be broken at some position from 1 to (n-1).

Constraints (3.3k) ensure that if a joint k is broken at some position other than the last position in the sequence, then there is exactly one joint that is broken immediately after joint k. The use of inequality in this constraint is to account for the fact that if k is the last joint broken, than p_{kl} will be zero. Since it is not known beforehand what joint will be broken last, hence use of inequality is imperative.

Constraints (3.3l) ensure that every joint (except the dummy joint F) that is broken during disassembly process is preceded by exactly one joint. Unlike the previous constraint, here it is known that the first joint to be broken is the joint F. This information can be usefully employed to obtain an equality constraint, which in general, will give a tighter formulation as compared to the inequality constraint.

Constraints (3.3m) in our formulation links the variables p_{kl} to variables z_{kn} in such a way that when precedence relationships are imposed on z_{kn}, they are automatically imposed on p_{kl}. Since we have already imposed precedence relationships on variables z_{kn} through constraints (3.3j), hence precedence on variables p_{kl} is ensured. Note that since p_{kl} are also sequencing variables, we need to impose precedence on them. Also, it must be noted that the use of z_{kn} variables in our formulation is solely for the purpose of imposing these precedence relationships in an indirect but simple manner. Constraints (3.3m) also prevent the occurrence of any subtours in the solution as explained in the following result:

Proposition 3.4.1a*: Constraints (3.3m) together with constraints (3.3j) enforce the precedence relationships on variable* p_{lk}

Proof: It follows from constraints (3.3m) that if joint k is broken at position n and joint l is broken at one of the positions in the range (n+1) to $|K|$, then the variable p_{lk} is zero. Now, since any joint that must succeed joint k can be broken only at positions in the range (n+1) to $|K|$ (this follows from constraints (3.3j)), we are ensured that, for all joints l that must succeed k, p_{lk} is zero. Thus, it follows from the definition of variables p_{lk} that a joint l that must succeed a joint k can never be broken before breaking joint k. Hence, the precedence relationships are imposed on variable p_{lk}. QED

Proposition 3.4.1b*: Constraints (3.3m) together with constraints (3.3i) prevent the occurrence of subtours in the solution of DOM1.*

Proof (by contradiction): Suppose the occurrence of a subtour is possible. Through constraints (3.3i), we have enforced the condition that each joint occupies a unique position in the sequence. Suppose there are m joints in the subtour. Further suppose that the first joint k in the subtour occupies the n^{th} position in the sequence and that l is the last joint in the subtour. Clearly, l occupies $(n+m)^{th}$ position in the sequence. Since we have a subtour with k as its first joint and l as its last joint, clearly, p_{lk} should be 1. However, it contradicts constraints (3.3m) which ensure that if joint k is broken at position n and joint l is broken at one of the positions in the range $(n+1)$ to $|K|$, then the variable p_{lk} is zero. Hence, p_{lk} cannot equal 1 in this case and a subtour is not possible. QED

Constraints (3.3n) ensure that if a joint *l* must precede a joint *k*, then the associated variable p_{kl} is zero. This follows directly from the definition of variable p_{kl}.

Constraints (3.3p) state that if a joint *j* precedes joint *i* and a joint *k* precedes joint *j*, then *i* cannot be broken immediately after *k*; joint *j* must be broken in the sequence between *k* and *i*. These constraints are not necessary in our model, but they may serve to tighten the LP relaxation and improve the performance of solution procedures.

Finally, constraints (3.3q) ensure that no joints are broken before the dummy starting joint *F*.

3.4.2 Disassembly Optimization Model DOM2

Maximize $\sum_{i \in I}(b_i - c_i)x_i - \sum_{k \in K} f_k y_k - \sum_{k \in K}\sum_{\substack{l \in K \\ L \neq k}} s_{kl} p_{kl}$

Subject to:

$$x_i + y_k \leq 1, \forall i \in I, k \in SOJ_i \tag{3.4a}$$

$$x_i \geq y_k, \forall i \in I, k \in KS_i \tag{3.4b}$$

$$x_i \leq y_k, \forall i \in I, k \in KP_i \tag{3.4c}$$

$$x_i + \sum_{k \in KS_i} y_k \geq \sum_{k \in KP_i} y_k - |KP_i| + 1, \forall i \in I : i \notin SOS \tag{3.4d}$$

$$x_i + \sum_{k \in KS_i} y_k \geq \sum_{j \in P_i}\left(\sum_{k \in KP_j} y_k - |KP_j| + 1\right) - |P_i| + 1, \forall i \in I : i \notin SOS, |P_i| \neq 0 \tag{3.4e}$$

$$\sum_{l \in (K - KPM_k)} p_{kl} = y_k, \forall k \in K : k \neq D \tag{3.4f}$$

$$\sum_{k \in (K - KSM_l)} p_{kl} = y_l, \forall l \in K : l \neq F \tag{3.4g}$$

$$p_{kl} = 0, \forall k \in K, l \in (KPM_k \cup \{k\}) \tag{3.4h}$$

$$q_{ij} = 1, \forall i \in K, j \in KSM_i \tag{3.4i}$$

$$q_{ij} + q_{ji} = 1, \forall i \in K, j \in K : i \neq j \tag{3.4j}$$

$$q_{ij} + q_{jk} + q_{ki} \leq 2, \forall i \in K, j \in K, k \in K : i \neq j, j \neq k, k \neq i \tag{3.4k}$$

$$p_{ij} \leq q_{ij}, \forall i \in K, j \in K \tag{3.4l}$$

$$p_{ki} = 0, \forall i \in K, j \in KPM_i, k \in KPM_j \tag{3.4m}$$

$$p_{iF} = 0, \forall i \in K \tag{3.4n}$$

$$\sum_{j \in K} p_{Fj} = 1 \tag{3.4p}$$

$$p_{Dj} = 0, \forall j \in K \tag{3.4q}$$

$$\sum_{i \in K} p_{iD} = 1 \tag{3.4r}$$

The objective function and the constraints (3.4a) to (3.4e) are exactly the same as constraints (3.3a) to (3.3e) in DOM1. Constraints (3.4f) and (3.4g) are similar to constraints (3.3k) and (3.3*l*) respectively, in DOM1, except that in constraints (3.4f), the inequality sign of (3.3k) has been replaced by an equality sign and we use dummy variable *D* instead of variable *L*. Note that, by definition, joint *L* is broken only if all other joints are broken in the optimal solution, whereas joint *D* is always broken irrespective of the status of other joints. Also, constraints (3.4h) are the same as constraints (3.3n) of DOM1 and constraints (3.4m) and (3.4n) are the same as constraints (3.3p) and (3.3q) respectively in DOM1.

Constraints (3.4i) to (3.4*l*) together enforce the subtour elimination and precedence constraints among joints. We defer the proof of these results to Appendix A, where we use these constraints to obtain new valid formulations for asymmetric traveling salesman problem (ATSP) and the precedence constrained asymmetric traveling salesman problem (PCATSP).

Note: In Appendix A, along with the valid formulations for ATSP and PCATSP, we also present the results of our experimentation, in which we compare the performance of our models with that of other well-known models discussed in literature using problems from the TSP library. Our results indicate that in comparison to other well-known formulations in literature, the proposed formulations are quite promising in terms of the LP relaxation bounds obtained and the number of branch and bound nodes explored to reach an optimal integer solution.

In the next section, we validate models DOM1 and DOM2 through a case study of a computer processor.

3.5 Case Study: Disassembly of a Computer Processor

We now illustrate the validity of our model through its application for disassembly optimization of a computer processor (The product description and details of mating relationships for the products used as examples in this thesis are obtained from the website of Disassembly Engineering Laboratory of the New Jersey Institute of Technology: http://dfqm.njit.edu).

We provide the following description of the products used as examples:

- a table showing the identification numbers of parts,
- a table showing the identification numbers and types of fasteners,
- a figure illustrating the mating relationships of components (the figure shows how the components are connected with each other without actually specifying their spatial arrangement in three dimensions. The joints are represented as double headed arrows.)
- composition of recoverable subassemblies (since the actual BOM of the product was not available, the subassemblies shown here are chosen just to illustrate the validity of the model)
- network representation of the product, and a table showing the preceding and succeeding joints of components/subassemblies.

Part name	**ID#**
Front cover	P1
Side Cover	P2
Voltage Source	P3
Mother Board	P4
LAN Card	P5
Sound Card	P6
Floppy Drive	P7
CD ROM	P8
Hard Disk Drive	P9
HDD Frame	P10
On Off Switch	P11
Reset Switch	P12
Red LED	P13
Green LED	P14
Main Frame	P15

Table 3.1 - Identification numbers of individual parts (computer processor)

Fastener ID	Number	Fastener type
F1	5	Screw
F2	10	Screw
F3	4	Screw
F4	4	Screw
F5	4	Screw
F6	4	Screw
F7	2	Screw
F8	2	Nail
F9	1	Quick Release Fasteners
F10	1	Quick Release Fasteners
F11	1	Quick Release Fasteners
I1	4	Snap Fit
I2	4	Snap Fit
I3	4	Snap Fit
I4	4	Snap fit
I5	2	Snap fit
I6	2	Snap fit

Table 3.2 - Identification numbers and types of fasteners (computer processor)

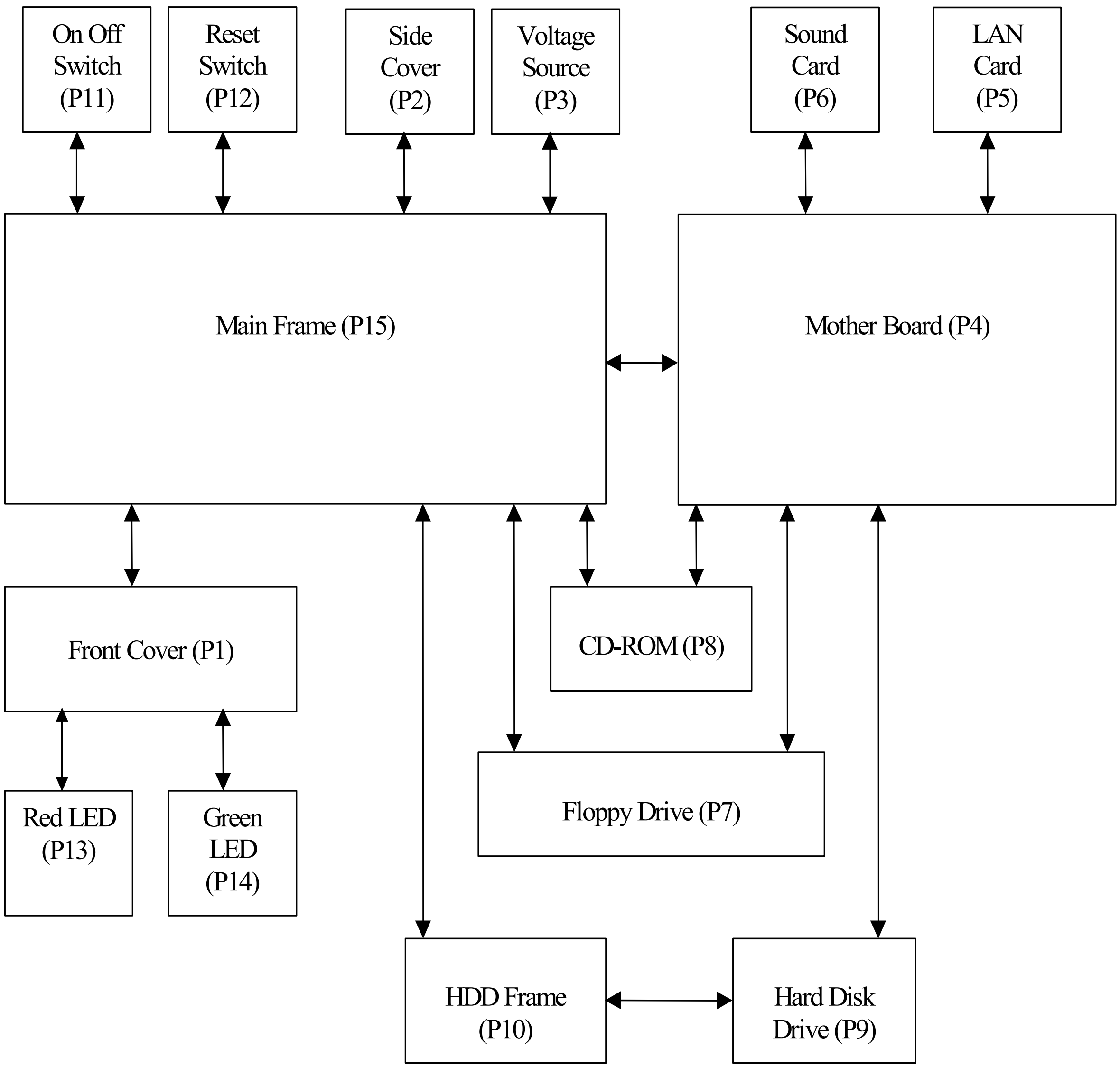

Fig 3.3 - Mating relations of the components (computer processor)

Subassembly	Parts	Joints
S1	P1, P13, P14	I5, I6
S2	P4, P5, P6, P7, P8, P9, P10, P11, P12, P15	I3, I4, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11
S3	P11, P12, P15	F7, F8
S4	P4, P5, P6, P7, P8, P11, P12, P15	I3, I4, F2, F7, F8, F10, F11
S5	P9, P10	F5

Table 3.3 – Composition of subassemblies (computer processor)

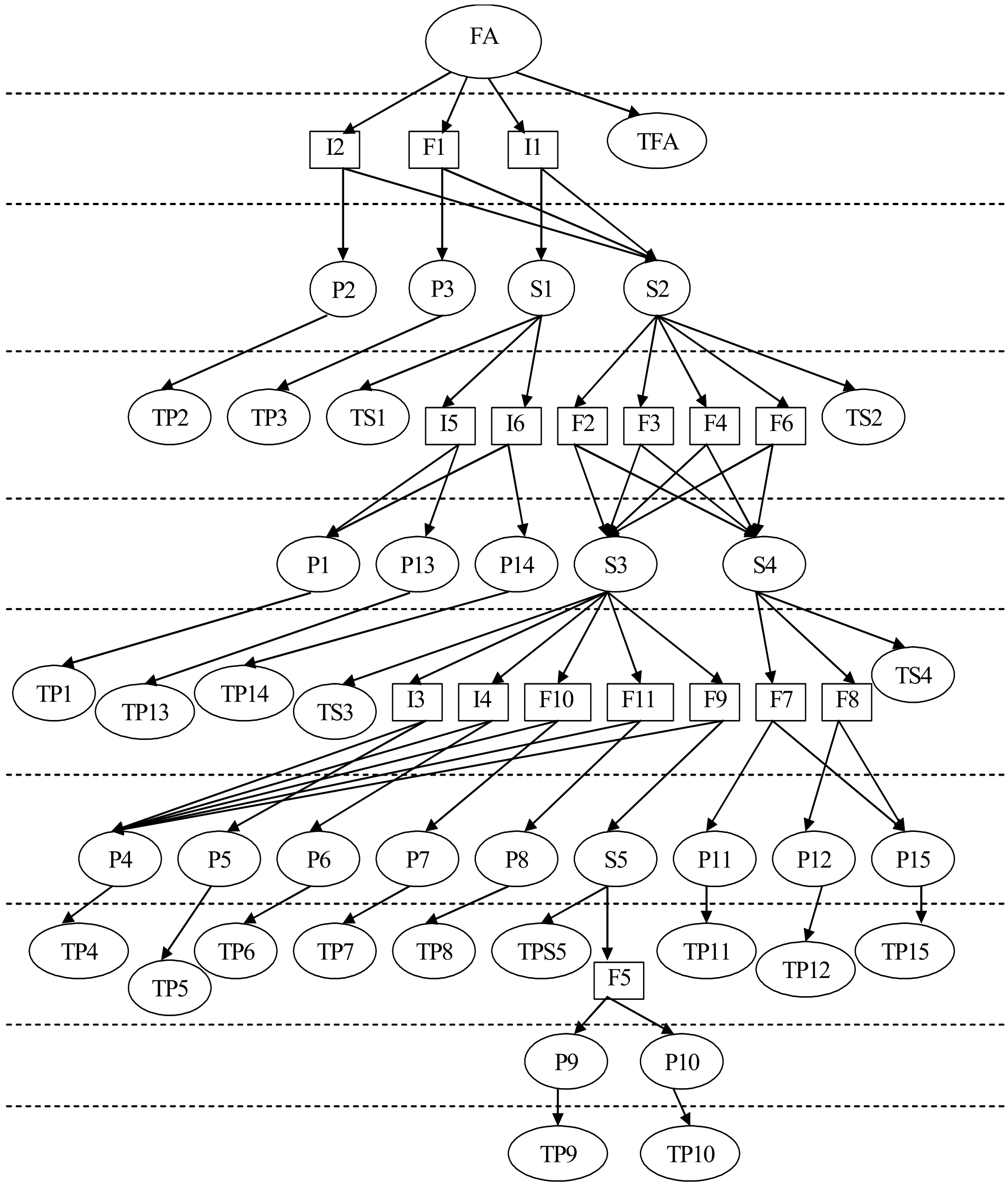

Fig 3.4 - Network representation of the processor assembly based on the procedure proposed in Sec. 3.2

Part/Subassembly	Preceding joint (s)	Succeeding joint (s)
P1	I1, I5, I6	___
P2	I2	I3, I4, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11
P3	F1	I3, I4, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11
P4	I1, I2, I3, I4, F1, F2, F3, F4, F6, F9, F10, F11	___
P5	I1, I2, I3, F1, F2, F3, F4, F6	___
P6	I1, I2, I4, F1, F2, F3, F4, F6	___
P7	I1, I2, F1, F2, F3, F4, F6, F10	___
P8	I1, I2, F1, F2, F3, F4, F6, F11	___
P9	I1, I2, F1, F2, F3, F4, F5, F6, F9, F11	___
P10	I1, I2, F1, F2, F3, F4, F5, F6, F9, F11	___
P11	I1, I2, F1, F2, F3, F4, F6, F7	___
P12	I1, I2, F1, F2, F3, F4, F6, F8	___
P13	I1, I2, I5, F1	___
P14	I1, I2, I6, F1	___
P15	I1, I2, F1, F2, F3, F4, F6, F7, F8	___
S1	I1	___
S2	I1, I2, F1	___
S3	I1, I2, F1, F2, F3, F4, F6	I3, I4, F5, F9, F10, F11
S4	I1, I2, F1, F6	___
S5	I1, I2, F1, F2, F3, F4, F6, F9	___

Table 3.4 - Preceding and succeeding joints for parts and subassemblies of a computer processor

3.6 Results of Experimentation

In Appendix B, we provide the complete data and description of our experimental set-up. In this section, we provide the results of our experimentation and a brief explanation, wherever required. In the results, we provide the objective function value, the optimal solution (parts/subassemblies recovered and joint sequence in the optimal solution) and the computational time (in CPU seconds) required for obtaining the solution for each data set using both DOM1 and DOM2.

Subcategory	**Parts/Subassemblies Recovered**	**Joint Sequence**	**Objective Function Value**	**CPU time (seconds)**	
				DOM1	**DOM2**
Subcategory 1	P2, P3, S1, S2	I2-I1-F1	81.71	4.6	4.2
Subcategory 2	FA (final assembly)	No joints broken	389.99	0.36	0.34
Subcategory 3	P2, P3, P4, P5, P6, P7, P8, S1, S3, S5	I2-I1-F1-F3-F4-F2-F6-F11-F10-I3-F9-I4	426.89	7.0	6.8

Table 3.5 – Results of experimentation: data set 1 (computer processor)

Subcategory	**Parts/Subassemblies Recovered**	**Joint Sequence**	**Objective Function Value**	**CPU time (seconds)**	
				DOM1	**DOM2**
Subcategory 1	FA (final assembly)	No joints broken	42.02	5.5	5.3
Subcategory 2	P2, P3, P4, P5, P6, P7, P8, S1, S3, S5	I2-F1-I1-F2-F4-F6-F3-I4-F10-I3-F11-F9	230.56	6.0	6.1
Subcategory 3	P2, P3, P8, P9, S1, S3	I2-F1-I1-F2-F4-F3-F9-F11-F5	209.64	1.5	1.5

Table 3.6 – Results of experimentation: data set 2 (computer processor)

For both the data sets, the values of parameters have been randomly generated from a uniform distribution. However, the range of these values is different in the two cases. We provide the range of various parameter values for the two data sets in table below:

Parameter	**Range (data set 1)**	**Range (data set 2)**
b_i (for individual parts)	15-25	30-40
b_i (for subassemblies)	35-60	60-70
c_i (for individual parts)	5-10	10-15
c_i (for subassemblies)	10-15	15-25
f_k	2-7	10-15
s_{kl}	1-20	20-40

Table 3.7 – Range of parameter values for data set1 and data set 2

For both the data sets and in all the cases, we find that the precedence relationships are satisfied and the results obtained are as expected from the data values.

3.7 Concluding Remarks

In this chapter, we validated our disassembly optimization models through their application to the disassembly of a computer processor. Since the number of component indices was small, we were able to solve the problem to optimality within a very few CPU seconds. However, for large and complex assemblies, it might take prohibitively large time to solve these models directly. Hence, in the next chapter, we discuss a methodology for solving difficult problems to obtain optimal or near optimal solutions.

CHAPTER 4

A METHODOLOGY FOR SOLVING THE DISASSEMBLY OPTIMIZATION PROBLEM

4.1 Introduction

In chapter 3, we presented two models for generating optimal disassembly plans to maximize the composite recovered value of components/subassemblies and minimize the sequence dependent set-up costs. Small problem instances can be easily solved using these models as we saw in our experimentation on the disassembly of a computer processor. However, for larger problems, direct solution approaches may take prohibitively large time. In this chapter, we discuss the details of a methodology to solve larger problem instances in reasonable amount of computational time. Our solution procedure uses the model DOM2 discussed in the previous chapter. After experimenting with several different approaches to solve the integer programming problems, especially those based on Lagrangian relaxation scheme, we found that DOM2 is more suitable for developing an effective solution procedure.

4.2 Description of a Procedure to Solve DOM2

The methodology for solving DOM2 comprises of three phases. In the first phase, we use the Lagrangian relaxation procedure to obtain an improved upper bound for our problem. The second phase involves an addition of valid inequalities as cuts to our relaxed problem. Finally, in the last phase, we restore the relaxed constraints and switch to a branch and bound procedure to obtain an optimal or near optimal solution. The details of this procedure are presented next.

4.2.1 Phase 1: Lagrangian Relaxation

4.2.1.1 A brief introduction to Lagrangian relaxation

The Lagrangian relaxation approach is based on the observation that the complexity of many difficult integer-programming problems can be attributed to a few side constraints and that relaxing these constraints makes the resulting problem relatively easier to solve. However, the resulting problem will most likely be infeasible in the absence of these complicating constraints. In the Lagrangian relaxation scheme, these complicating constraints are added to the objective function with a penalty term whose value is iteratively varied based on the amount of violation of these constraints and their dual variables. This penalty term clearly plays a significant role in fast convergence of the procedure and a vast amount of research on Lagrangian relaxation is devoted to the determination of appropriate penalty terms. It has been shown that the resulting Lagrangian problem provides an upper bound (for a maximization problem) on the optimal solution value of the original problem (Fisher, 1985). Consequently, as soon as a feasible solution is obtained in the Lagrangian implementation, we know that an optimal solution has been found. However, convergence to an optimal solution is rare and in most circumstances, one has to settle with an improved bound on the problem or if possible, use some heuristic procedure to derive a feasible solution.

4.2.1.2 Choice of constraints to be relaxed

In chapter 3, we pointed out that constraints (3.4i) through (3.4*l*) of DOM2 enforce the required precedence relationships and the subtour elimination constraints (SECs). As discussed in the previous chapter, our problem closely resembles the asymmetric traveling salesman problem with precedence constraints and it is well known that the precedence forcing constraints and subtour elimination constraints are responsible for inducing complexity in such problems (Ascheuer *et al*, 1993). Consequently, in applying the Lagrangian relaxation technique to our problem, the choice of constraints to be dualized should be such that the precedence forcing constraints and SECs are relaxed.

The precedence forcing constraints and SECs are actually enforced on variables q_{ij} and that constraints (3.4*l*) help us achieve the desired result for variables p_{ij}. Thus, in our model, constraints (3.4*l*) are instrumental in enforcing both precedence and subtour elimination constraints. Relaxing these constraints in a Lagrangian scheme separates out the subproblems in the q-variables and in the (x, y, p)-variables, both of which are relatively much easier to solve. Note that we solve the Lagrangian problem with integer restrictions on (x, y, p)-variables, while the q-variables are non-negative continuous variables.

4.2.1.3 Determination of Lagrangian multipliers

Having determined the constraints to be relaxed, the next question is how to compute the value of Lagrangian multipliers. For this purpose, we use the subgradient optimization scheme due to Held *et al* (1974). We now provide an overview of this scheme. Most of the following material is from Fisher (1981, 1985).

We start with an integer programming problem in the following form:

$$Z = \text{maximize } cx$$

Subject to:

$Ax \leq b$,

$Dx \leq e$,

$x \geq 0$ and integral,

where x is n × 1, b is m × 1 and all other matrices have conformable dimensions. We assume that constraints $Ax \leq b$ are complicating constraints and dualize these constraints using non-negative Lagrangian multipliers u to obtain the following Lagrangian problem:

$Z_D(u)$ = maximize $cx + u(b - Ax)$

Subject to:

$Dx \leq e$,

$x \geq 0$ and integral.

Ideally, the multipliers u should be such that they solve the following dual problem:

$Z_D = \min Z_D(u),\ u \geq 0.$

In the subgradient optimization scheme, a sequence of values for u is determined iteratively starting with an initial point u^0 and applying the formula:

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

In this formula, t_k is a scalar step size at any iteration k and x^k is an optimal solution to ($LR_u{}^k$), the Lagrangian problem with multiplier set to u^k.

For determining the scalar step size t_k, a formula that has proved effective in practice is:

$$t_k = \frac{\lambda_k (Z_D(u^k) - Z^*)}{\sum_{i=1}^{m} (b_i - \sum_{j=1}^{n} a_{ij} x_j{}^k)^2}, \tag{4.1b}$$

where Z^* is the objective function value of the best known feasible solution to the original problem, $\lambda_k \in [0, 2]$ is a scalar, and $Z_D(u_k)$ represents the Lagrangian value at iterate k, which yields an upper bound on the objective function value. As the Lagrangian relaxation scheme is not guaranteed to converge to optimality, the method is usually terminated either upon reaching a specified iteration limit or when the step size becomes lower than a specified limit.

4.2.1.4 Implementation details of Lagrangian scheme for DOM2

As we mentioned earlier, we have chosen constraints (3.4*l*) for relaxation in our Lagrangian scheme. We restate constraints (3.4*l*) below:

$$p_{ij} \le q_{ij}, \forall i \in K, j \in K, i \ne j$$

We initially set $\lambda_k = 2$. If the objective function value of the Lagrangian problem failed to improve over three consecutive iterations, we halved the value of λ_k. For obtaining an initial feasible solution, we considered the simple situation in which no disassembly takes place, i.e., no joints are broken ($y_k = 0$, $\forall\ k \in K$), and no parts/subassemblies are recovered. In this case, the p_{kl} variables are all equal to 0. Also, $x_i = 0$, $\forall\ i \in I$ except for the final complete assembly, for which $x_{FA} = 1$. Noting the objective function of DOM, this solution yields an initial value of :

$$Z^* = (b_{FA} - c_{FA}). \tag{4.1c}$$

The values of dual variables are obtained using (4.1a) and are given by:

$$u^{k+1} = \max\ \{0, u^k - t_k\,(q_{ij} - p_{ij})\},\ \forall\ i \in K, j \in K \tag{4.1d}$$

The Lagrangian multipliers u_{ij} were initially set to 0, $\forall\ i, j$

Similarly, the step size t_k is determined using the formula (4.1b) and is given by:

$$t_k = \frac{\lambda_k (Z_D(u^k) - Z^*)}{\sum_{i \in K} \sum_{j \in K} (p_{ij} - q_{ij})^2} \tag{4.1e}$$

We terminated the procedure when any one of the following conditions was satisfied:

- The step size becomes less than 0.1;
- The number of iterations reaches 10, or
- The square-root of the denominator in (4.1e) becomes (near) zero (less than 10^{-4}, i.e., a near-zero subgradient is obtained), or the difference between the lower and upper bounds in the numerator of (4.1e) becomes sufficiently small (less than 10^{-4}).

4.2.2 Phase 2: Addition of Valid Inequalities

In our experiments, we have observed that, although the Lagrangian scheme discussed above works quite well in providing a succession of improving upper bounds, it does not converge to an optimal solution to the underlying dual problem in a reasonable computational time. Hence, we use the foregoing heuristic stopping criteria. These restrictions have worked quite well in the overall scheme of our implementation.

Since we have relaxed (3.4*l*) that contributes towards imposing the precedence relations and avoiding subtours, the infeasibility as inherent in *p*-vector might either be in the form of the presence of subtours in the solution or a violation of some precedence constraints. Accordingly, in this phase, we identify the violated inequalities (subtours or precedence violations) as described below. Next, we include cuts in our model that explicitly eliminate the identified subtour or precedence violation. The augmented model is then resolved using the phase I scheme until we (fortuitously) obtain a provable optimal solution (where the lower and upper bounds in the numerator of (19) match), or we have added a total of 10 valid inequalities. The steps of identifying infeasibilities in the sequence and generating valid inequalities are addressed in turn below.

4.2.2.1 Identification of subtours and precedence violations

We first discuss a procedure to determine what subtour, if any, is present in the incumbent Lagrangian dual subproblem solution. Recall that dummy joints *F* and *D* are always broken, and that joint *F* is the first to be broken in the sequence of joints and joint *D* is the last. We can find a subtour in a solution by implementing the following steps:

1) We first determine the number of joints that are broken including the dummy joints. Let this number be *n*.
2) We then sequentially start to number the joints based on their position in the sequence starting with *F* as the first joint and *D* as the last.
3) If there are no subtours in the solution, then the position number of joint *D* should be *n* because a total of *n* joints are broken and *D* is always the last joint in the sequence of broken joints. If the position number of *D* is less than *n*, then we know that at least one subtour is present in the solution.
4) Now, we proceed to determine the actual subtour. Since we have numbered all joints starting from *F* and ending at *D*, any joint, which is part of a subtour, will not be assigned a number. Suppose that joint *k* is such a joint. We number *k* as 1.
5) We number joint *l* for which $p_{kl} = 1$ as 2. Proceeding in this fashion, we keep numbering the joints until we again reach joint *k*. Note that this process is

actually numbering the nodes on a subtour; hence, starting from joint k, we will, at some point, reach joint k again. Thus, we can find a subtour present in a solution.

We first run this procedure to find if a subtour is present. As soon as a subtour is found, we add the corresponding constraint to eliminate this subtour as discussed in the following subsection and reiterate by applying phase I. (Note that after addition of each inequality, we do not implement the complete phase 1, but employ just one iteration of the subgradient optimization procedure). Otherwise, if no subtour exists, we check for precedence violations. It is important to realize that we use this procedure only when no subtour is found in our solution. The following procedure determines a path $P(k, l)$ in the solution sequence, where $l \in KPM_k$, if one exists.

1) Suppose that n joints are broken in the solution including the dummy joints. Since no subtour is present in the solution, it must be true that joint F is numbered 1 and joint D is numbered n.
2) For all $k \in K$ and $l \in KPM_k$ (set of preceding joints of k) such that both l and k are broken, we check if the position number of joint l is greater than joint k. If that is the case, then we have a precedence violation because joint l belongs to set of preceding joints of joint k and it appears later than joint k in the sequence of broken joints.
3) Having found a precedence violation, we again number the joints starting with k as joint 1 and going up to joint l, to identify the required path $P(k, l)$.

As soon as one such path is found, we add the corresponding cut as described in the next subsection to eliminate this precedence violation.

4) If no such precedence violation is found, then the incumbent Lagrangian subproblem solution is feasible to DOM, and we can use it to update the incumbent solution value Z^*, using the objective function of DOM. If this also matches the upper bound obtained from the Lagrangian subproblem, then this solution is optimal for DOM and we terminate the overall procedure. Otherwise, we proceed to phase 3.

We now turn our attention to the description of valid inequalities used in our procedure.

4.2.2.2 Valid inequalities to eliminate subtours –the D_k^- - inequalities

In our solution procedure, we have used the D_k^--inequalities for subtour elimination, as first introduced by Grotschel and Padberg (1985) and proven to be facet-defining for the asymmetric traveling salesman problem polytope. These inequalities are obtained by sequentially lifting the cycle inequality on the n nodes $i_1,\ldots,i_n$, as given by:

$$\sum_{j=1}^{n-1} p_{i_j,i_{j+1}} + p_{i_n i_1} \leq n-1.$$

Depending on the order of lifting the variables, various classes of inequalities can be obtained. One of these is known as the D_k^- (D_k minus) inequalities, given by:

$$\sum_{j=1}^{n-1} p_{i_j i_{j+1}} + p_{i_n i_1} + 2\sum_{j=3}^{n} p_{i_1 i_j} + \sum_{j=4}^{n}\sum_{h=3}^{j-1} p_{i_j i_h} \leq n-1.$$

Once a subtour is found, we add the corresponding D_k^- inequality to the model and implement a single phase 1 iteration.

4.2.2.3 Valid inequalities to eliminate precedence violations

The precedence forcing constraints we have used are similar to those used in sequential ordering problem as shown in description of model DSGM1. For a path *P* (*k, l*), such a constraint can be written as:

$$\sum_{i \in W} p_{ki} + \sum_{i \in W} \sum_{j \in W} p_{ij} + \sum_{j \in W} p_{jl} <= |W|, \forall W \subseteq K\{k,l\}, W \neq \phi$$

In our implementation, we have *W* = set of joints in path *P* (*k, l*) excluding *k* and *l*.

4.2.2.4: Summary of implementation details of Phase 2

In phase 2, we identify the violated inequalities and add constraints to eliminate the violations. We first implement the procedure to find a subtour in the current solution. If a subtour is found, we add the corresponding D_k^- inequality in our model. If no subtour is found, then we implement our procedure to find path *P* (*k, l*) consisting of precedence violations. If such a path is found, we add the corresponding precedence forcing inequality to avoid the recurrence of this particular path. We add a total of 10 valid inequalities to our model in phase 2 and if an optimal solution is not detected, we implement phase 3 of our methodology, which is discussed next.

4.2.3 Phase 3: Variable Fixing Step and Branch and Bound Implementation

In this phase, we use the branch and bound procedure for solving integer-programming problems. But, before we implement the branch and bound scheme, we implement an additional variable-fixing step that might fix certain p_{ij} variables to 0. We first discuss the details of this step.

4.2.3.1 Variable fixing step

During our implementation of phases 1 and 2, we keep track of the solutions found after each iteration. For each ordered pair of joints (i, j), we maintain a counter g_{ij}. For all the ordered pairs, the counter g_{ij} is initialized to 0. If in any of the solutions, we find that p_{ij} =1, then the counter g_{ij} is set to 1. Hence, at the end of phase 2, g_{ij} will be 0 for all those pairs (i, j) for which joint j is not broken immediately after joint i in any of the solutions. For all such pairs (i, j), we fix the variable p_{ij} to 0. The logic behind such variable fixing is obvious. If during several iterations in phases 1 and 2, p_{ij} for a pair (i, j) never turned out to be 1, then it is unlikely that it will be 1 in the optimal solution of the problem. That does not imply that it cannot be 1; in fact, in the worst case, such variable fixing might render the problem infeasible. However, we will observe from the results of our experimentation in the next chapter that in none of the instances that we have tested, we have encountered the problem of infeasibility. In fact, in most cases, we have found the optimal solution to our problem in much smaller computational time. In certain cases, our solution procedure does not give the optimal solution, but again in all such cases, the solution obtained is very close to the optimal solution.

4.2.3.2 Branch and Bound scheme

After fixing the variables as described above, we restore the constraints (3.4*l*) that we had relaxed in phase 1. We then use the branch and bound procedure of CPLEX to solve the problem. We have used the default settings on CPLEX in our implementation.

4.3 Concluding Remarks

In this chapter, we provided details of our three-phase solution methodology to obtain solution for our disassembly optimization problem. The code for this methodology was written in AMPL command language. We have included the code in Appendix C. In the next chapter, we provide the results of our experimentation and compare the computational time and solution quality of our methodology with the optimal solution obtained by directly using the CPLEX solver.

CHAPTER 5
RESULTS AND CONCLUSION

In this chapter, we provide the details of our experimental set-up. We provide the results of our experimentation and present a case study on the disassembly of a computer printer. We conclude this chapter by indicating potential areas in which further research should be conducted.

5.1 Design of Experiment

The experimental set-up consists of two stages. In the first stage, the objective is to determine the effect of various factors on the performance of our model and solution procedure. We specifically considered the following two factors:

1. The relative number of joints and parts/subassemblies in the final assembly, and
2. The values of the parameters such as the set-up cost, the cost of breaking the joints and the net recovery value.

To study the effect of number of parts and subassemblies, we considered the parts-to-joints ratios of 0.5, 1 and 2. We will denote these ratios as P/J. For each ratio, we considered three data sets with different number of parts and joints. These are shown in the Table 5.1:

:

P/J ratio	Data Set 1		Data Set 2		Data Set 3	
	# of parts	# of joints	# of parts	# of joints	# of parts	# of joints
0.5	10	20	15	30	20	40
1	20	20	30	30	40	40
2	20	10	30	15	40	20

Table 5.1 - Different combinations of parts and joints in the first stage of experiments

For each data set, the preceding and succeeding joints of each part/subassembly were generated randomly using MS Excel. After generating these preceding relationships randomly, we eliminated any possible inconsistency in data by resolving the conflicts in precedence relationships. For example, we ensured that same joint does not belong to the set of preceding and succeeding joints of a part.

For each combination of parts and joints shown in Table 5.1, we experimented with several different combinations of parameter values. All the values were generated randomly from a uniform distribution using MS Excel. The range of values of set-up cost (s_{kl}), cost of breaking the joints (f_k) and the net recovery value of parts and subassemblies ($b_i - c_i$), which we used in experimentation, are shown in Table 5.2.

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)
15-30	1-10	1-15	15-30
	1-10	15-30	30-45
	1-10	30-45	45-60
	10-20	1-15	15-30
	10-20	15-30	30-45
	10-20	30-45	45-60
	20-30	1-15	15-30
	20-30	15-30	30-45
	20-30	30-45	45-60
30-45	10-20	15-30	30-45
	10-20	30-45	45-60
	10-20	45-60	60-75
	20-30	15-30	30-45
	20-30	30-45	45-60
	20-30	45-60	60-75
	30-40	15-30	30-45
	30-40	30-45	45-60
	30-40	45-60	60-75
45-60	20-30	30-45	45-60
	20-30	45-60	60-75
	20-30	60-75	75-90
	30-40	30-45	45-60
	30-40	45-60	60-75
	30-40	60-75	75-90
	40-50	30-45	45-60
	40-50	45-60	60-75
	40-50	60-75	75-90

Table 5.2 - Ranges of Parameter Values for Experimentation

For each of the 9 combinations of parts and joints shown in Table 5.1, we experimented with all the 27 combinations of parameter values shown in Table 5.2. Thus, we experimented with a total of 243 data sets in this stage. All the problems were solved through direct use of CPLEX and using our solution methodology.

The first stage helped us in identifying the factors that make the problems relatively difficult to solve. Since we solved all the problems using both the direct run in CPLEX and our three-phase solution procedure, we were also able to study the performance of our solution procedure and establish its superiority. In the second stage of experimental design, we applied our solution procedure to several problems generated to reflect the factors that cause a problem to be difficult to solve. The aim was to further test the performance of our procedure as compared to the direct runs and to substantiate our claim regarding its superiority. In this stage, we experimented with 12 test problems. For each problem, we ran 3 replications i.e. we randomly generated the values of parameters from the same range thrice. Thus, we had a total of 36 data sets in this stage. We present the results of our experimentation in section 5.2.

5.2 Results of Experimentation

5.2.1 Results of Experimentation in Stage 1

In this section, we provide the results of stage 1 experimentation. We used ILOG AMPL version 10.6.16 – Win 32 with CPLEX 7.0.0 as solver.

Tables 5.3 to 5.11 contain the results of the first stage of the experimentation. Along with the parameter values, we state the following information in Tables 5.3 to 5.11:

- Objective function value obtained by direct run in CPLEX - **obj_DR**
- Objective function value obtained using three-phase solution procedure - **obj_SP**
- Percentage difference in obj_SP and obj_DR i.e.(obj_SP – obj_DR)*100/obj_DR – **obj_PD** (Note that for our maximization problem, positive values of obj_PD imply that the solution procedure performed better than the direct run).
- CPU time for direct run using CPLEX – **ct_DR**
- CPU time for three-phase solution procedure in minutes and seconds – **ct_SP**

Note that the CPU time is stated in minutes and seconds. Also, for all problems, we set a time limit of 360 minutes for both the direct run and the solution procedure. We report the best solution obtained in this time limit. In Tables 5.3 to 5.11, the objective function values obtained after aborting the runs after the time limit of 360 minutes are marked with an asterisk (*). For those problems, which were aborted, the performance of our solution procedure can be judged by comparing the best solution given by the solution procedure and the direct run in the specified time limit.

Number of parts – 10

Number of joints – 20

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	20.95	20.95	0%	00:01	00:01
	1-10	15-30	30-45	65.67	65.67	0%	00:01	00:01
	1-10	30-45	45-60	135.67	135.67	0%	00:02	00:01
	10-20	1-15	15-30	20.95	20.95	0%	00:01	00:01
	10-20	15-30	30-45	50.12	50.12	0%	00:01	00:01
	10-20	30-45	45-60	106.75	106.75	0%	00:01	00:01
	20-30	1-15	15-30	20.95	20.95	0%	00:01	00:01
	20-30	15-30	30-45	31.47	31.47	0%	00:01	00:01
	20-30	30-45	45-60	83.00	83.00	0%	00:01	00:01
30-45	10-20	15-30	30-45	31.47	31.47	0%	00:01	00:01
	10-20	30-45	45-60	76.75	76.75	0%	00:02	00:01
	10-20	45-60	60-75	139.73	139.73	0%	00:01	00:01
	20-30	15-30	30-45	31.47	31.47	0%	00:01	00:01
	20-30	30-45	45-60	53.00	53.00	0%	00:01	00:01
	20-30	45-60	60-75	109.09	109.09	0%	00:01	00:01
	30-40	15-30	30-45	31.47	31.47	0%	00:01	00:01
	30-40	30-45	45-60	51.47	51.47	0%	00:01	00:01
	30-40	45-60	60-75	84.92	84.92	0%	00:01	00:01
45-60	20-30	30-45	45-60	51.47	51.47	0%	00:01	00:01
	20-30	45-60	60-75	71.20	71.20	0%	00:01	00:01
	20-30	60-75	75-90	119.17	119.17	0%	00:02	00:01
	30-40	30-45	45-60	51.47	51.47	0%	00:01	00:01
	30-40	45-60	60-75	65.26	65.26	0%	00:01	00:01
	30-40	60-75	75-90	95.00	95.00	0%	00:01	00:01
	40-50	30-45	45-60	51.47	51.47	0%	00:01	00:01
	40-50	45-60	60-75	65.26	65.26	0%	00:01	00:01
	40-50	60-75	75-90	79.55	79.55	0%	00:01	00:01

Table 5.3 - Results of stage 1 of experimentation (10 parts and 20 joints)

Number of parts – 15

Number of joints – 30

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	23.97	23.97	0%	00:01	00:01
	1-10	15-30	30-45	37.46	37.46	0%	00:03	00:01
	1-10	30-45	45-60	84.16	84.16	0%	4:16	2:46
	10-20	1-15	15-30	23.97	23.97	0%	00:01	00:01
	10-20	15-30	30-45	37.46	37.46	0%	00:01	00:01
	10-20	30-45	45-60	52.46	52.46	0%	00:02	00:01
	20-30	1-15	15-30	23.97	23.97	0%	00:01	00:01
	20-30	15-30	30-45	37.46	37.46	0%	00:01	00:01
	20-30	30-45	45-60	52.46	52.46	0%	00:02	00:01
30-45	10-20	15-30	30-45	37.46	37.46	0%	00:02	00:01
	10-20	30-45	45-60	52.46	52.46	0%	00:02	00:01
	10-20	45-60	60-75	72.83	72.83	0%	00:02	00:01
	20-30	15-30	30-45	37.46	37.46	0%	00:01	00:01
	20-30	30-45	45-60	52.46	52.46	0%	00:02	00:01
	20-30	45-60	60-75	72.83	72.83	0%	00:01	00:01
	30-40	15-30	30-45	37.46	37.46	0%	00:02	00:01
	30-40	30-45	45-60	52.46	52.46	0%	00:01	00:01
	30-40	45-60	60-75	72.83	72.83	0%	00:02	00:01
45-60	20-30	30-45	45-60	52.46	52.46	0%	00:01	00:01
	20-30	45-60	60-75	72.83	72.83	0%	00:01	00:01
	20-30	60-75	75-90	80.52	80.52	0%	00:02	00:01
	30-40	30-45	45-60	52.46	52.46	0%	00:01	00:01
	30-40	45-60	60-75	72.83	72.83	0%	00:01	00:01
	30-40	60-75	75-90	80.52	80.52	0%	00:02	00:01
	40-50	30-45	45-60	52.46	52.46	0%	00:01	00:01
	40-50	45-60	60-75	72.83	72.83	0%	00:01	00:01
	40-50	60-75	75-90	80.52	80.52	0%	00:01	00:01

Table 5.4 - Results of stage 1 of experimentation (15 parts and 30 joints)

Number of parts – 20

Number of joints – 40

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	28.81	28.81	0%	00:05	00:02
	1-10	15-30	30-45	35.38	35.38	0%	00:06	00:02
	1-10	30-45	45-60	163.5	161.23	-1.39%	68:58	18:44
	10-20	1-15	15-30	28.81	28.81	0%	00:03	00:02
	10-20	15-30	30-45	35.38	35.38	0%	00:04	00:01
	10-20	30-45	45-60	57.76	57.76	0%	25:12	00:02
	20-30	1-15	15-30	28.81	28.81	0%	00:03	00:01
	20-30	15-30	30-45	35.38	35.38	0%	00:02	00:01
	20-30	30-45	45-60	57.76	57.76	0%	00:04	00:01
30-45	10-20	15-30	30-45	35.38	35.38	0%	00:04	00:01
	10-20	30-45	45-60	57.76	57.76	0%	00:04	00:01
	10-20	45-60	60-75	69.05	69.05	0%	24:54	00:02
	20-30	15-30	30-45	35.38	35.38	0%	00:02	00:01
	20-30	30-45	45-60	57.76	57.76	0%	00:04	00:01
	20-30	45-60	60-75	69.05	69.05	0%	00:04	00:02
	30-40	15-30	30-45	35.38	35.38	0%	00:02	00:01
	30-40	30-45	45-60	57.76	57.76	0%	00:02	00:01
	30-40	45-60	60-75	69.05	69.05	0%	00:02	00:02
45-60	20-30	30-45	45-60	57.76	57.76	0%	00:04	00:01
	20-30	45-60	60-75	69.05	69.05	0%	00:04	00:01
	20-30	60-75	75-90	81.89	81.89	0%	00:04	00:01
	30-40	30-45	45-60	57.76	57.76	0%	00:02	00:01
	30-40	45-60	60-75	69.05	69.05	0%	00:04	00:02
	30-40	60-75	75-90	81.89	81.89	0%	00:04	00:01
	40-50	30-45	45-60	57.76	57.76	0%	00:02	00:01
	40-50	45-60	60-75	69.05	69.05	0%	00:03	00:02
	40-50	60-75	75-90	81.89	81.89	0%	00:03	00:01

Table 5.5 - Results of stage 1 of experimentation (20 parts and 40 joints)

Number of parts – 20

Number of joints – 20

s_{kl}	f_k	$(b_i – c_i)$ (parts)	$(b_i – c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	21.58	21.58	0%	00:01	00:01
	1-10	15-30	30-45	65.67	65.67	0%	00:01	00:03
	1-10	30-45	45-60	148.46	148.46	0%	00:02	00:05
	10-20	1-15	15-30	21.58	21.58	0%	00:01	00:01
	10-20	15-30	30-45	50.12	50.12	0%	00:01	00:02
	10-20	30-45	45-60	100.74	100.74	0%	00:02	00:04
	20-30	1-15	15-30	21.58	21.58	0%	00:01	00:01
	20-30	15-30	30-45	31.47	31.47	0%	00:01	00:01
	20-30	30-45	45-60	71.08	71.08	0%	00:01	00:03
30-45	10-20	15-30	30-45	31.47	31.47	0%	00:01	00:01
	10-20	30-45	45-60	64.83	64.83	0%	00:02	00:02
	10-20	45-60	60-75	130.63	130.63	0%	00:01	00:03
	20-30	15-30	30-45	31.47	31.47	0%	00:01	00:01
	20-30	30-45	45-60	47.59	47.59	0%	00:01	00:01
	20-30	45-60	60-75	95.66	95.66	0%	00:03	00:05
	30-40	15-30	30-45	31.47	31.47	0%	00:01	00:01
	30-40	30-45	45-60	47.59	47.59	0%	00:01	00:01
	30-40	45-60	60-75	71.49	71.49	0%	00:01	00:02
45-60	20-30	30-45	45-60	47.59	47.59	0%	00:01	00:01
	20-30	45-60	60-75	61.98	61.98	0%	00:01	00:01
	20-30	60-75	75-90	121.56	121.56	0%	00:05	00:03
	30-40	30-45	45-60	47.59	47.59	0%	00:01	00:01
	30-40	45-60	60-75	61.98	61.98	0%	00:01	00:01
	30-40	60-75	75-90	97.39	97.39	0%	00:05	00:03
	40-50	30-45	45-60	47.59	47.59	0%	00:01	00:01
	40-50	45-60	60-75	61.98	61.98	0%	00:01	00:01
	40-50	60-75	75-90	89.46	89.46	0%	00:01	00:01

Table 5.6 - Results of stage 1 of experimentation (20 parts and 20 joints)

Number of parts – 30

Number of joints – 30

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	17.12	17.12	0%	00:01	00:01
	1-10	15-30	30-45	91.4	91.4	0%	326:01	106:40
	1-10	30-45	45-60	540.64*	540.64	0%	360:00	186:14
	10-20	1-15	15-30	17.12	17.12	0%	00:01	00:01
	10-20	15-30	30-45	42.79	42.79	0%	00:18	00:01
	10-20	30-45	45-60	313.41	313.41	0%	352:46	156:50
	20-30	1-15	15-30	17.12	17.12	0%	00:02	00:01
	20-30	15-30	30-45	42.79	42.79	0%	00:03	00:01
	20-30	30-45	45-60	89.88	89.88	0%	102:26	106:10
30-45	10-20	15-30	30-45	42.79	42.79	0%	00:16	00:02
	10-20	30-45	45-60	52.46	52.46	0%	00:45	00:02
	10-20	45-60	60-75	364.17*	359.49	-1.29%	360:00	126:42
	20-30	15-30	30-45	42.79	42.79	0%	00:04	00:01
	20-30	30-45	45-60	52.46	52.46	0%	00:35	00:02
	20-30	45-60	60-75	141.8	141.8	0%	83:56	85:36
	30-40	15-30	30-45	42.79	42.79	0%	00:02	00:01
	30-40	30-45	45-60	52.46	52.46	0%	00:07	00:01
	30-40	45-60	60-75	66.68	66.68	0%	00:22	00:01
45-60	20-30	30-45	45-60	52.76	52.76	0%	00:37	00:02
	20-30	45-60	60-75	66.68	66.68	0%	00:30	00:02
	20-30	60-75	75-90	197.98	197.79	-0.0096%	12:55	8:11
	30-40	30-45	45-60	52.76	52.76	0%	00:04	00:01
	30-40	45-60	60-75	66.68	66.68	0%	00:13	00:02
	30-40	60-75	75-90	83.13	83.13	0%	3:01	00:02
	40-50	30-45	45-60	52.76	52.76	0%	00:02	00:01
	40-50	45-60	60-75	66.68	66.68	0%	00:15	00:02
	40-50	60-75	75-90	83.13	83.13	0%	00:16	00:01

* - Aborted after 6 hours of runtime

Table 5.7 - Results of stage 1 of experimentation (30 parts and 30 joints)

Number of parts – 40

Number of joints – 40

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	36.87	36.87	0%	00:02	00:01
	1-10	15-30	30-45	128.35*	196.95*	53.48%	360:00	360:00
	1-10	30-45	45-60	640.4*	745.88*	16.47%	360:00	360:00
	10-20	1-15	15-30	34.58	34.58	0%	00:04	00:02
	10-20	15-30	30-45	72.07	72.07	0%	121:20	00:18
	10-20	30-45	45-60	54.89*	517.13*	842.12%	360:00	360:00
	20-30	1-15	15-30	23:49	23:49	0%	00:03	00:03
	20-30	15-30	30-45	58.06	58.06	0%	00:04	00:01
	20-30	30-45	45-60	216.91*	240.65*	10.94%	360:00	360:00
30-45	10-20	15-30	30-45	69.23	69.23	0%	00:03	00:02
	10-20	30-45	45-60	87.77*	87.77*	0%	360:00	360:00
	10-20	45-60	60-75	640.6*	640.6*	0%	360:00	360:00
	20-30	15-30	30-45	58.06	58.06	0%	00:02	00:02
	20-30	30-45	45-60	76.60	76.60	0%	00:15	00:04
	20-30	45-60	60-75	381.21*	362.70*	-4.86%	360:00	360:00
	30-40	15-30	30-45	48.40	48.40	0%	00:01	00:01
	30-40	30-45	45-60	66.94	66.94	0%	00:13	00:04
	30-40	45-60	60-75	101.98*	127.01*	24.54%	360:00	360:00
45-60	20-30	30-45	45-60	76.60	76.60	0%	00:01	00:02
	20-30	45-60	60-75	91.70	91.70	0%	21:43	00:04
	20-30	60-75	75-90	547.28*	571.55*	4.43%	360:00	360:00
	30-40	30-45	45-60	66.94	66.94	0%	00:04	00:01
	30-40	45-60	60-75	101.98	101.98	0%	00:06	00:02
	30-40	60-75	75-90	306.72*	315.07*	2.72%	360:00	360:00
	40-50	30-45	45-60	60.43	60.43	0%	00:04	00:01
	40-50	45-60	60-75	95.47	95.47	0%	00:05	00:02
	40-50	60-75	75-90	132.25	132.25	0%	00:82	00:18

*- Aborted after 6 hours of runtime

Table 5.8 - Results of stage 1 of experimentation (40 parts and 40 joints)

Number of parts – 20

Number of joints – 10

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	75.95	75.95	0%	00:01	00:01
	1-10	15-30	30-45	174.37	174.37	0%	00:01	00:01
	1-10	30-45	45-60	400.23	400.23	0%	00:01	00:01
	10-20	1-15	15-30	67.67	67.67	0%	00:01	00:01
	10-20	15-30	30-45	166.09	166.09	0%	00:01	00:01
	10-20	30-45	45-60	343.05	343.05	0%	00:01	00:01
	20-30	1-15	15-30	56.60	56.60	0%	00:01	00:01
	20-30	15-30	30-45	155.02	155.02	0%	00:01	00:01
	20-30	30-45	45-60	292.77	292.77	0%	00:01	00:01
30-45	10-20	15-30	30-45	168.23	168.23	0%	00:01	00:01
	10-20	30-45	45-60	303.84	303.84	0%	00:01	00:01
	10-20	45-60	60-75	471.20	471.20	0%	00:01	00:01
	20-30	15-30	30-45	158.34	158.34	0%	00:01	00:01
	20-30	30-45	45-60	292.45	292.45	0%	00:01	00:01
	20-30	45-60	60-75	407.12	407.12	0%	00:01	00:01
	30-40	15-30	30-45	146.00	146.00	0%	00:01	00:01
	30-40	30-45	45-60	283.75	283.75	0%	00:01	00:01
	30-40	45-60	60-75	371.67	371.67	0%	00:01	00:01
45-60	20-30	30-45	45-60	294.56	294.56	0%	00:01	00:01
	20-30	45-60	60-75	380.69	380.69	0%	00:01	00:01
	20-30	60-75	75-90	506.56	506.56	0%	00:01	00:01
	30-40	30-45	45-60	287.87	287.87	0%	00:01	00:01
	30-40	45-60	60-75	372.76	372.76	0%	00:01	00:01
	30-40	60-75	75-90	484.51	484.51	0%	00:01	00:01
	40-50	30-45	45-60	269.36	269.36	0%	00:01	00:01
	40-50	45-60	60-75	357.28	357.28	0%	00:01	00:01
	40-50	60-75	75-90	470.12	470.12	0%	00:01	00:01

Table 5.9 - Results of stage 1 of experimentation (20 parts and 10 joints)

Number of parts – 30

Number of joints – 15

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	242.32	242.32	0%	00:09	00:02
	1-10	15-30	30-45	344.83	344.83	0%	00:18	00:03
	1-10	30-45	45-60	752.07	752.07	0%	00:11	00:02
	10-20	1-15	15-30	164.73	164.73	0%	00:13	00:05
	10-20	15-30	30-45	203.44	203.44	0%	00:10	00:06
	10-20	30-45	45-60	610.68	610.68	0%	00:15	00:03
	20-30	1-15	15-30	82.54	82.54	0%	00:11	00:04
	20-30	15-30	30-45	104.47	104.47	0%	00:17	00:04
	20-30	30-45	45-60	511.71	511.71	0%	00:17	00:06
30-45	10-20	15-30	30-45	64.48	64.48	0%	00:04	00:02
	10-20	30-45	45-60	471.72	471.72	0%	00:05	00:05
	10-20	45-60	60-75	895.85	895.85	0%	00:03	00:01
	20-30	15-30	30-45	35.03	35.03	0%	00:01	00:01
	20-30	30-45	45-60	372.75	372.75	0%	00:03	00:05
	20-30	45-60	60-75	796.88	796.88	0%	00:03	00:03
	30-40	15-30	30-45	35.03	35.03	0%	00:01	00:01
	30-40	30-45	45-60	238.14	238.14	0%	00:04	00:01
	30-40	45-60	60-75	662.27	662.27	0%	00:04	00:02
45-60	20-30	30-45	45-60	210.11	210.11	0%	00:03	00:01
	20-30	45-60	60-75	634.24	634.24	0%	00:01	00:02
	20-30	60-75	75-90	1031.90	1031.90	0%	00:02	00:03
	30-40	30-45	45-60	75.50	75.50	0%	00:01	00:01
	30-40	45-60	60-75	499.63	499.63	0%	00:01	00:03
	30-40	60-75	75-90	897.29	897.29	0%	00:01	00:01
	40-50	30-45	45-60	51.88	51.88	0%	00:03	00:01
	40-50	45-60	60-75	386.32	386.32	0%	00:01	00:02
	40-50	60-75	75-90	783.98	783.98	0%	00:01	00:01

Table 5.10 - Results of stage 1 of experimentation (30 parts and 15 joints)

Number of parts – 40

Number of joints – 20

s_{kl}	f_k	$(b_i - c_i)$ (parts)	$(b_i - c_i)$ (subassemblies)	obj_DR	obj_SP	obj_PD (%)	ct_DR	ct_SP
15-30	1-10	1-15	15-30	130.72	130.72	0%	00:03	00:02
	1-10	15-30	30-45	466.58	466.58	0%	00:33	00:07
	1-10	30-45	45-60	1056.08	1056.08	0%	00:26	00:10
	10-20	1-15	15-30	69.94	69.94	0%	00:01	00:05
	10-20	15-30	30-45	334.21	334.21	0%	00:05	00:08
	10-20	30-45	45-60	898.94	898.94	0%	00:04	00:07
	20-30	1-15	15-30	57.59	57.59	0%	00:03	00:02
	20-30	15-30	30-45	306.26	306.26	0%	00:01	00:03
	20-30	30-45	45-60	771.36	771.36	0%	00:01	00:02
30-45	10-20	15-30	30-45	304.21	304.21	0%	00:01	00:01
	10-20	30-45	45-60	673.94	673.94	0%	00:03	00:03
	10-20	45-60	60-75	1265.34	1265.34	0%	00:04	00:02
	20-30	15-30	30-45	276.26	276.26	0%	00:01	00:01
	20-30	30-45	45-60	546.36	546.36	0%	00:10	00:07
	20-30	45-60	60-75	1132.91	1132.91	0%	00:11	00:04
	30-40	15-30	30-45	243.28	243.28	0%	00:01	00:01
	30-40	30-45	45-60	451.15	451.15	0%	00:01	00:01
	30-40	45-60	60-75	959.87	959.87	0%	00:03	00:02
45-60	20-30	30-45	45-60	441.84	441.84	0%	00:01	00:01
	20-30	45-60	60-75	909.29	909.29	0%	00:01	00:01
	20-30	60-75	75-90	1430.09	1430.09	0%	00:01	00:01
	30-40	30-45	45-60	408.86	408.86	0%	00:01	00:01
	30-40	45-60	60-75	736.25	736.25	0%	00:04	00:01
	30-40	60-75	75-90	1257.05	1257.05	0%	00:01	00:01
	40-50	30-45	45-60	379.75	379.75	0%	00:01	00:01
	40-50	45-60	60-75	648.32	648.32	0%	00:04	00:01
	40-50	60-75	75-90	1090.68	1090.68	0%	00:01	00:01

Table 5.11 - Results of stage 1 of experimentation (40 parts and 20 joints)

5.2.2 Inferences from Stage 1 Experimentation

Based on the results of our experimentation, we can infer the following:

1. The values of parameters have a significant impact on computational time. This can be observed from the computational time values, particularly in Tables 5.7 and 5.8. It can be seen that while some problems couldn't be solved in the pre-specified time limit of 360 minutes, several others problems were solved in less than 1 minute. We believe that such a significant difference can be attributed to the random nature of parameter values. In real world scenarios, we would expect that the values of parameters are not completely random. For example, the same type of joints will have the same cost of breaking. Similarly, the set-up cost between the identical joints will be very small. We expect that these factors will significantly reduce the computational time.

2. It can be observed that several problem instances were solved very quickly through direct runs using CPLEX. We have already highlighted the effectiveness of our model for small instances earlier through the case study on disassembly of a computer processor in chapter 3. The real strength of our three-phase solution procedure lies in solving large problem instances, which require large amount of computational time. We can observe this fact from our results. As compared to direct runs in CPLEX, the three-phase solution procedure was able to solve these problem instances in much smaller computational in almost all the cases. Also, for those instances, which ran to completion within the specified time limit, the quality of solution obtained by the

three-phase solution procedure was extremely good, with optimal solution obtained in all but three instances. In these three instances, where the three-phase procedure did not produce an optimum, the solution obtained was within 1.5% of the optimal solution. For those cases in which the runs were aborted, the solution given by three-phase procedure was better than the one produced by direct runs in most cases. We will further emphasize these points through a case study on the disassembly of a laser printer presented section 5.4.

3. An interesting observation is that for most problem instances with P/J ratio of 0.5, the optimal solution was "no disassembly". Such a result can be explained intuitively. Note that P/J ratio of 0.5 implies that a large number of joints need to be broken to retrieve a small number of parts/subassemblies. This, in turn, implies that the costs associated with breaking the joints and set-up will outweigh the benefits associated with component recovery. Hence, the best solution in most of these cases is to avoid the breaking of any joint in the final product, which implies no disassembly.

4. On the other extreme, in case of P/J ratio of 2, the optimal solution in several cases is complete disassembly. Again, this can be attributed to the fact that a large number of component indices can be recovered by breaking a fewer number of joints. Hence, the depth is disassembly is relatively higher in these instances.

5. Another significant observation in the case of a "no disassembly" solution is that while the direct runs may require several minutes to arrive at an optimal solution, the

three-phase solution procedure solves the problem very quickly. This results from the fact that the relaxed problem used in the three-phase procedure itself provides this optimal solution. Note that the constraints that we have relaxed are the ones that enforce precedence and subtour elimination among the joints in the final solution. In the case of “no disassembly”, as none of the joints are broken in the optimal solution, the question of subtours and precedence violations does not arise. Hence, the first Lagrangian relaxation subproblem satisfies the primal feasibility and complementary slackness conditions, yielding the optimal solution after the first iteration in phase 1.

6. It appears that the most difficult problems are those which have equal number of parts and joints (P/J = 1). This can be observed from Tables 5.7 and 5.8, which show the results for 30 parts/30 joints and 40 parts/40 joints problems respectively. In stage 2 of experimentation, we focussed on problem instances in which P/J ratios are close to 1.

7. Another important observation from the results in Tables 5.3 to 5.11 is that the most challenging problems were those in which the parameter values were in the following range:

   s_{kl}: 15-30, f_k: 1-10, $(b_i$-$c_i)$(parts): 30_45, $(b_i$-$c_i)$(subassemblies): 45-60

   It appears that while for small problem instances, the range of parameter values is irrelevant, for larger instances, especially those with P/J ratio of 1, the problems with the parameter values in this range proved to be the most difficult ones to solve.

Again, since we wanted to focus on computationally difficult problems in stage 2, we used parameter values from this range.

We now present the results from our experimentation in stage 2.

5.2.3 Results of Experimentation in Stage 2

Based on our inferences in the previous section, we compared the performance of our solution procedure with that of the direct run for several computationally difficult problems. For each combination of parts and joints, we generated three different sets of parameter values from a uniform distribution. We refer to these data sets as DS1, DS2 and DS3 respectively. The results are presented in Tables 5.12 and 5.13.

$\lvert I \rvert$	$\lvert K \rvert$	DS1		DS2		DS3	
		obj_DR	obj_SP	obj_DR	obj_SP	obj_DR	obj_SP
30	27	517.21*	514.70*	492.13	492.13	546.22	546.22
30	35	474.07*	479.08*	510.67*	510.67	535.39*	535.39*
32	40	332.49*	354.02*	344.76*	339.27	388.72	382.36
34	32	569.99	569.99	525.52	525.52	569.34	569.34
35	35	153.41*	153.41*	520.15*	536.06*	615.27*	622.92
35	38	494.73*	494.73*	504.11*	498.36	479.22*	479.22
37	43	589.71*	615.23*	592.89*	592.89*	574.23*	572.44*
38	36	732.20*	735.01	710.59*	732.23*	694.02*	698.00*
38	38	685.34*	681.94*	709.25*	714.82*	675.29*	712.54*
40	32	892..63*	892.63	916.43*	998.29*	926.53*	924.17
40	43	54.74*	54.74*	830.61*	837.62*	785.02*	788.37*
43	40	889.96*	892.48*	757.66*	853.24*	944.05*	943.01*

* - aborted after 6 hours of run time

Table 5.12 –Objective function values obtained by the procedure developed and CPLEX

|I|	|K|	DS1		DS2		DS3	
		ct_DR	**ct_SP**	**ct_DR**	**ct_SP**	**ct_DR**	**ct_SP**
30	27	360:00	360:00	37:36	33:06	93:30	33:50
30	35	360:00	360:00	360:00	225:36	360:00	360:00
32	40	360:00	360:00	360:00	244:27	344:16	227:34
34	32	131:49	103:42	88:21	43:11	290:11	174:34
35	35	360:00	360:00	360:00	360:00	360:00	221:13
35	38	360:00	360:00	360:00	287:54	360:00	256:28
37	43	360:00	360:00	360:00	360:00	360:00	360:00
38	36	360:00	277:00	360:00	360:00	360:00	360:00
38	38	360:00	360:00	360:00	360:00	360:00	360:00
40	32	360:00	315.27	360:00	360:00	360:00	314:16
40	43	360:00	360:00	360:00	360:00	360:00	360:00
43	40	360:00	360:00	360:00	360:00	360:00	360:00

Table 5.13 –Computational time required by the procedure developed and CPLEX

In Table 5.14, we provide a comparison of percentage differences in the objective function values and computation times of the direct run and the solution procedure. As in stage 1, the relative difference in objective function values is calculated as:

$$obj\_PD = \frac{Value(SP) - Value(DR)}{Value(DR)} * 100,$$

while the relative difference in computational time values is calculated as:

$$ct\_PD = \frac{Time(DR) - Time(SP)}{Time(DR)} * 100$$

Thus, positive values of obj_PD and ct_PD indicate that the solution procedure performed better than the direct run using CPLEX.

|I|	|K|	DS1		DS2		DS3	
		obj_PD	ct_PD	obj_PD	ct_PD	obj_PD	ct_PD
30	27	-0.49%*	0%	0%	11.97%	0%	63.81%
30	35	1.06%*	0%	0%	37.33%	0%*	0%
32	40	6.48%*	0%	-1.59%	32.10%	-1.63%	33.90%
34	32	0%	21.33%	0%	51.12%	0%	39.84%
35	35	0%*	0%	3.06%*	0%	1.24%	38.55%
35	38	0%*	0%	-1.14%	20.03%	0%	28.76%
37	43	4.33%*	0%	0%*	0%	-0.32%*	0%
38	36	0.38%	23.06%	3.05%*	0%	0.57%*	0%
38	38	-0.50%*	0%	0.79%*	0%	5.51%	0%
40	32	0%	12.38%	8.93%*	0%	-0.25%	12.70%
40	43	0%*	0%	0.08%*	0%	0.43%*	0%
43	40	0.28%*	0%	12.62%*	0%	-0.11%*	0%

* - aborted after 6 hours of run time for both the direct run and the solution procedure

Table 5.14 – Comparison of the objective function values and the computational time of the procedure developed with direct run using CPLEX

From the results in the above tables, we observe in most cases, the three-phase solution procedure was able to solve the problems in much smaller computation time, giving solutions which were very close to those obtained using direct run. Note that in all cases, the solution produced by our solution procedure was within 2% of that produced by the direct run. Also, in several cases, large reductions in computational time adequately compensated for less-than-optimal solution. For example, in case of 32 parts/40joints problem, the solution obtained by direct run was better than that of our solution procedure by 1.63% for data set DS3. However, the solution procedure took 33.90% less time in solving the problem. In cases where the runs were aborted, the solution procedure usually gave a better solution as compared to the direct run in CPLEX. In few instances where the direct run performed better, the relative difference in solution quality never exceeded 0.5%.

We now demonstrate the usefulness of our solution procedure through a real world application of the disassembly of a laser printer.

5.3 Case Study: Disassembly of a Laser Printer

In this section, we determine optimal disassembly plans for a laser printer. (The product description and details of mating relationships for the products used as examples in this thesis are obtained from the website of Disassembly Engineering Laboratory of the New Jersey Institute of Technology: http://dfqm.njit.edu).

As in the previous case study presented in chapter 3, we provide the following description of the product used in this example:

- a table showing the identification numbers of parts,
- a table showing the identification numbers and types of fasteners,
- a figure illustrating the mating relationships of components (the figure shows how the components are connected with each other without actually specifying their spatial arrangement in three dimensions. The joints are represented as double headed arrows.)
- composition of recoverable subassemblies (since the actual BOM of the product was not available, the subassemblies shown here are chosen just to illustrate the validity of the model)

- network representation of the product, and a table showing the preceding and succeeding joints of components/subassemblies.

Part Name	ID #
Paper input tray	P1
Cartridge	P2
Side casing	P3
PCB subassembly	P4
PCB subassembly casing	P5
PCB	P6
Steel frame	P7
Motor current board	P8
Side black casing	P9
Power current board	P10
Motor casing	P11
Flap cover	P12
Thermal unit	P13
Cooling unit	P14
Motor	P15
Top cover	P16
Support frame	P17
Roller	P18
Black casing	P19
Bottom steel frame	P20
PCB 1	P21
PCB 2	P22
Fan	P23
Busbar	P24
Back subassembly	P25
Bottom assembly	P26

Table 5.15 - Identification numbers for individual parts (laser printer)

Fastener ID	Number	Fastener type
F1	8	Screw
F2	4	Screw
F3	8	Screw
F4	10	Screw
F5	4	Screw
F6	8	Screw
F7	10	Screw
F8	2	Screw
F9	1	Screw
F10	4	Screw
F11	4	Screw
F12	2	Screw
F13	2	Screw
F14	4	Screw
F15	6	Nail
F16	4	Screw
F17	2	Screw
F18	13	Screw
F19	4	Screw
F20	4	Screw
F21	2	Screw
F22	2	Screw
F23	8	Screw
I1	2	Snap fit
I2	2	Snap fit
I3	2	Snap fit
I4	2	Snap fit

Table 5.16 - Identification numbers for fasteners (laser printer)

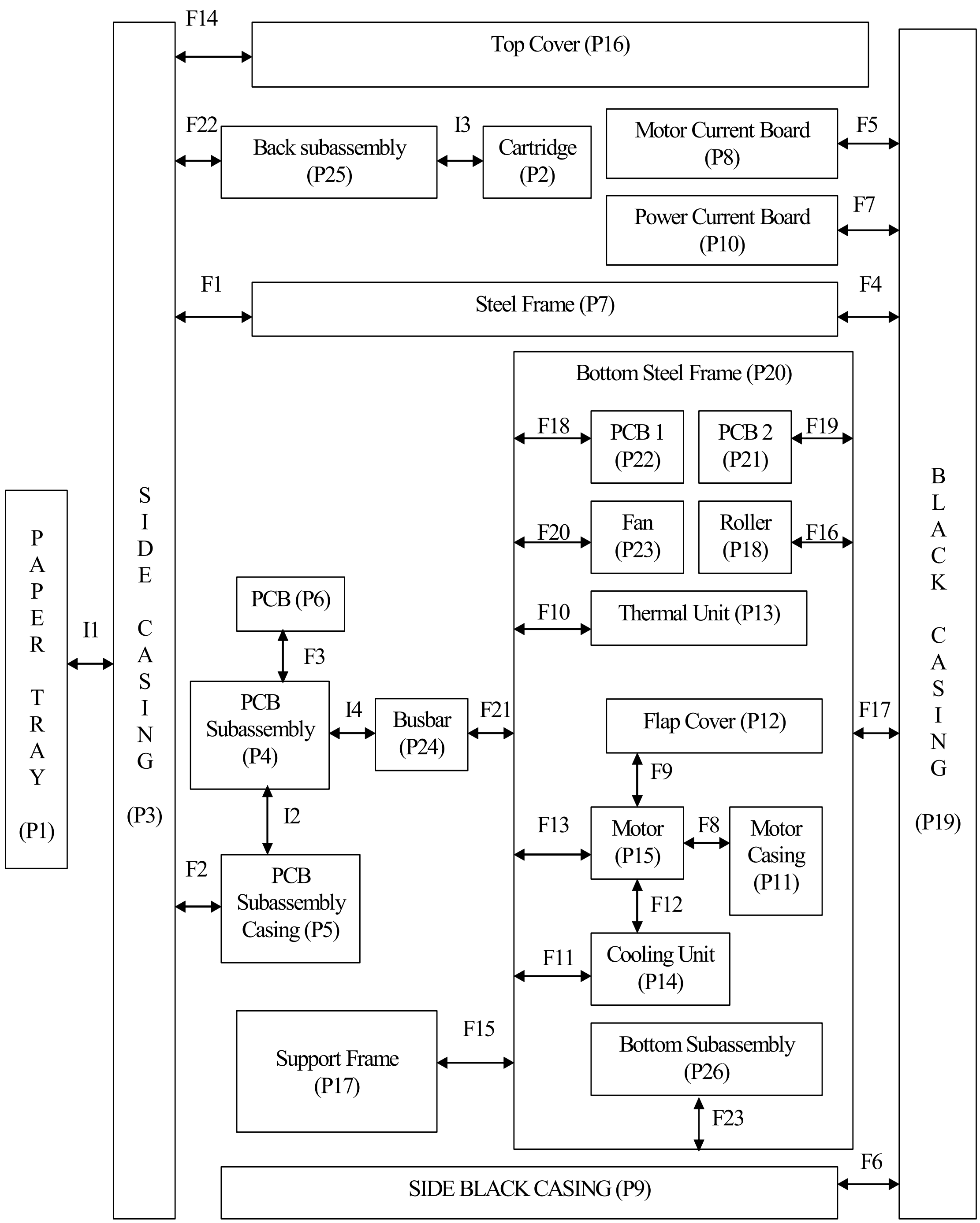

Fig. 5.1 - Mating relationships of the components (Laser Printer)

Subassembly	Parts	Joints
S1	P1, P13, P14	I5, I6
S2	P4, P5, P6, P7, P8, P9, P10, P11, P12, P15	I3, I4, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11
S3	P11, P12, P15	F7, F8
S4	P4, P5, P6, P7, P8, P11, P12, P15	I3, I4, F2, F7, F8, F10, F11

Table 5.17 – Composition of subassemblies (laser printer)

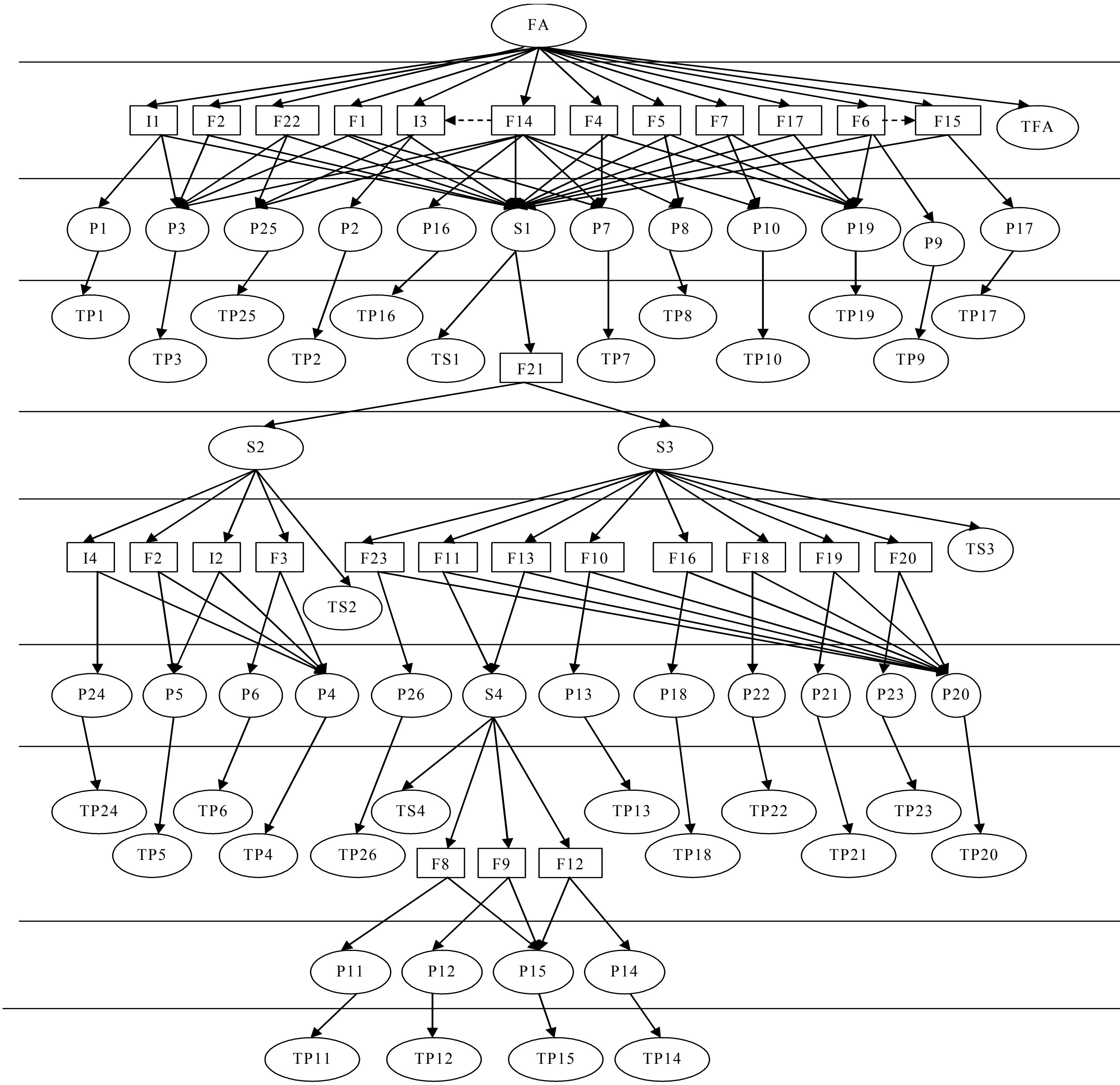

Fig. 5.2 - Network representation of the printer assembly

Part/Subassembly	Preceding Joint (s)	Succeeding Joint (s)
P1	I1	--
P2	I3, F14	--
P3	I1, F1, F2, F14, F22	I2, I4, F3, F21
P4	I2, I4, F1, F2, F3, F14, F21, F22	--
P5	I2, F1, F2, F14, F21, F22	--
P6	F1, F2, F3, F14, F21, F22	--
P7	F1, F4, F14	--
P8	F5, F14	--
P9	F6	F15
P10	F7, F14	--
P11	F4, F5, F6, F7, F8, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P12	F4, F5, F6, F7, F9, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P13	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P14	F4, F5, F6, F7, F10, F11, F12, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P15	F4, F5, F6, F7, F8, F9, F10, F11, F12, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P16	F14	I3
P17	F6, F15	--
P18	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P19	F4, F5, F6, F7, F17	F8, F9, F10, F11, F12, F13, F16, F18, F19, F20, F23
P20	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P21	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P22	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P23	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
P24	I4, F1, F2, F14, F21, F22	--
P25	I3, F14, F22	--
P26	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--
S1	I1, I3, F1, F4, F5, F6, F7, F14, F15, F17, F22	--
S2	F1, F2, F14, F21, F22	--
S3	F4, F5, F6, F7, F15, F17, F21	--
S4	F4, F5, F6, F7, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F23	--

Table 5.18 - Preceding and Succeeding Joints for Parts and Subassemblies of a Printer

In appendix D, we provide the data and parameter values for this case study. For the data set given, we found that both the three-phase solution procedure and the direct run provide an optimal solution of 48.17. However, while the three-phase procedure takes 26:18 to find the optimal solution, the time taken by direct run is 70:01. The parts/subassemblies recovered and the joint sequence are shown in the table below:

Parts recovered	P1, P2, P3, P7, P8, P9, P10, P16, P17, P19, P25, S2, S3
Joint sequence	S-I1-F6-F2-F17-F15-F14-F7-F5-I3-F1-F22-F4-F21-D

Table 5.19 – Optimal solution for laser printer disassembly

5.4 Conclusion and Directions for Future Research

In this research effort, we developed integer-programming models for disassembly optimization to minimize sequence dependent set-up cost and maximize the benefits from component recovery. We demonstrated the strength of our models through their application on case study of a computer processor. For larger problem instances, we developed an effective solution procedure that provided promising results in our experiments. At this time, we are continuing to experiment with more data sets.

A useful byproduct of our research effort is a new model for precedence constrained asymmetric traveling salesman problem (also known as sequential ordering problem or SOP). We added several constraints to tighten this representation. More research needs to be done to investigate the characteristics of this model and develop algorithms to solve this problem.

Another potential area of research is for the determination of potential benefits of end-of-life options. We have seen that the disassembly optimization procedure is highly dependent on benefits gained from recovery of parts/subassembly. We assumed in our research that an estimate of such recovery values is available; however, since the knowledge of these estimates is central to the accuracy of solutions obtained using mathematical models, it is imperative that research efforts are directed in this area.

In the long term, the focus of the research should be on the development of an integrated and preferably automated system that links the MRP and assembly planning databases to the optimization module. Such a system should also be able to make an assessment of difficulties in disassembly process and provide feedback to the product designer, so that modifications can be incorporated at the design stage itself to make the product more suitable for disassembly in the future.

REFERENCES

1) Allenby, B.R., 1994. *Greening of industrial ecosystems*, National Academy of Engineering Press, Washington DC.

2) Ascheuer, N., L.F. Escuredo, M. Grotschel, and M. Stoer, 1990. On identifying in polynomial time violated subtour elimination and precedence forcing constraints for the sequential ordering problem, *Integer Programming and Combinatorial Optimization*, Kannan, R, and W. Pulleybank (eds.), University of Waterloo, Waterloo, 1990, pp. 19-28.

3) Ascheuer, N., M. Junger, and G. Reinelt, 2000. A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints, *Computational Optimization and Applications*, vol. 17, pp. 61-84.

4) *Automotive Engineering*, 1993. Design for aluminum recycling, October issue, pp. 65-68.

5) *Automotive Engineering*, 1993. Modeling costs of plastics recycling, October issue, pp. 53-57.

6) Balas, E., 1989. The prize collecting traveling salesman problem, *Networks*, vol. 19, no. 6, pp. 621-636.

7) Balas, E., M. Fischetti, and W.R. Pulleybank, 1995. The precedence- constrained asymmetric traveling salesman polytope. *Mathematical Programming*, vol. 68, pp. 241-265.

8) Boks, C.B., W.C.J. Brouwers, E. Kroll, and A.L.N. Stevels, 1996. Disassembly modeling: Two applications to a Philips 21 inch television set, *Proceedings of the*

1996 IEEE International Symposium on Electronics and the Environment, pp. 224-229.

9) Bergmann, R., 1998. Introduction to case-based reasoning, Retrieved March 2, 2001 from http://www.cbr-web.org/CBR-Web/cbrintro/sld004.html

10) Boothroyd, G., and L. Alting, 1992. Design for assembly and disassembly, *Keynote Paper, Annals of the CIRP*, vol. 41, no. 2, pp. 625-636.

11) Brooke, L., 1991. Plastics recyclability update, *Automotive industries*, April issue, pp. 40-44.

12) Chen, R.W., D. Navin-Chandra, and F.B. Prinz, 1993. Product design for recyclability: A cost-benefit analysis model and its application, *Proceedings of the 1993 IEEE International Symposium on electronics and the Environment*, pp. 178-183.

13) Corbet, K.S., 1996. Design for value maximization: Putting a business lens on environmental activities, *Proceedings of the 1996 IEEE International Symposium on Electronics and the Environment*, pp. 81-86.

14) Dantzig, G., D. Fulkerson, S. Johnson, 1954. Solution of a large scale traveling salesman problem. *Operations Research*, vol. 2, pp. 393-410.

15) Das, S.K., and S.C. Sarin, 1994. Formulation and solution of a master aggregate scheduling problem in computer integrated manufacturing systems, *International Journal of Production Economics*, Fall issue.

16) Das, S.K., P. Yedlarajiah, and R. Narendra, 2000. An approach for estimating the end-of-life disassembly effort and cost, *International Journal of Production Research*, vol. 38, no. 3, pp. 657-673.

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

18) Dewhurst, P., 1993. Product design for manufacture: Design for disassembly, *Industrial Engineering*, vol. 25, no. 9, pp. 26-28.

19) Escudero, L.F., M. Guignard, and K. Malik, 1994. A Lagrangian relax-and-cut approach for the sequential ordering problem with precedence relationships. *Annals of Operations Research*, vol.50, pp. 219-237.

20) Field, F.R., and J.P. Clarke, 1991. Recycling of USA automobile materials, ATA, vol. 44, nos. 8-9, pp. 541-545.

21) Fiksel, J., 1996. *Design For Environment: Creating Eco-Efficient Products and Processe*s, McGraw-Hill, New York.

22) Fisher, M.L., 1981. The Lagrangian relaxation method for solving integer programming problems, *Management Science*, vol. 27, no.1, pp. 1-18.

23) Fisher, M.L., 1985. An application oriented guide to Lagrangian relaxation, *Interfaces*, vol. 15, no. 2, pp. 10-21.

24) Goan, H., 1996. *An integrated approach to environmentally conscious design and manufacturing*, Doctoral Dissertation, Department of Industrial and Systems Engineering, Virginia Tech.

25) Gouveia, L., and J. Pires, 1999. The asymmetric travelling salesman problem and a reformulation of the Miller-Tucker-Zemlin constraints. *European Journal of Operational Research*, vol. 112, pp.134-146.

26) Gouveia, L., and J. Pires, 2001. The asymmetric travelling salesman problem: on generalizations of disaggregated Miller-Tucker-Zemlin constraints. *Discrete Applied Mathematics*, vol. 112, pp. 129-145.

27) Grotschel, M., and M. Padberg, 1985. "Polyhedral Theory" *in The Traveling Salesman Problem*, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys (Eds.), John Wiley & Sons, New York,

28) Gungor, A., and S.M. Gupta, 1997. An evaluation methodology for disassembly processes, *Computers and Industrial Engineering*, vol. 33, no. 2, pp. 329-332.

29) Gungor, A., and S.M. Gupta, 1998. Disassembly sequence planning for products with defective parts in product recovery, *Computers and Industrial Engineering*, vol. 35, no. 1, pp. 161-164.

30) Gungor, A. and S.M. Gupta, 1999. Issues in environmentally conscious manufacturing and product recovery: a survey, *Computers and Industrial Engineering*, vol. 36, pp. 811-853.

31) Homem de Mello, L.S., and A.C. Sanderson, 1991. AND/OR graph representation of assembly plans, *Robotics*, pp. 1113-1119.

32) Holt, D.J., 1993. Recycling and the automobile, *Automotive Engineering*, October issue, pp. 42-50.

33) Huang, H., M.H. Wang, and M.R. Johnson, 2000. Disassembly sequence generation using neural network approach, *Journal of Manufacturing Systems*, Vol. 19, no. 2, pp. 73-82.

34) Ioannou, G., and S.C. Sarin, 1997. Design for de-manufacturing and generation of disassembly plans: The current state of knowledge, *Proceedings of*

the 1997 Flexible Automation and Intelligent Manufacturing Conference, Middlesborough, UK, June 25-27, pp. 216-227.

35) Ishii, K., C.F. Eubanks, and P. Di Marco, 1994. Design for product retirement and material life-cycle, *Materials and Design*, vol. 15, no. 4, pp. 225-233.

36) Johnson, M.R., and M.H. Wang, 1995. Planning product disassembly for material recovery, *International Journal of Production Research*, vol. 33, no. 11, pp. 3119-3142.

37) Jovane, F., L. Alting, A. Armillotta, W. Eversheim, K. Feldmann, G. Seliger, and N. Roth, 1993. A key issue in product life cycle: disassembly, *Annals of the CIRP*, vol. 42, no. 2, pp. 651-658.

38) Junger, Reinelt and Rinaldi, 1995. A literature review on the traveling salesman problem, *Handbook of OR & MS*, volume on *Network Models*, M. Ball (editor), Elsevier, Amsterdam.

39) Kriwet, A., E. Zussman, and G. Seliger, 1995. Systematic integration of design-for-recycling into product design, *International Journal of Production Economics,* vol. 38, no. 1, pp. 15-32.

40) Kroll, E., B. Beardsley, and A. Parulian, 1996. A methodology to evaluate ease of disassembly for product recycling, IEEE Transactions, vol. 28, pp. 837-845.

41) Kroll, E., and T.A. Hanft, 1998. Quantitative evaluation of product disassembly for recycling, *Research in Engineering Design*, vol. 10, pp. 1-14.

42) Kuo, T.C., H.C. Zhang, and S.H. Huang, 2000. Disassembly analysis for electromechanical products: A graph-based heuristic approach, *International Journal of Production Research*, vol. 38, no. 5, pp. 903-1007.

43) Lambert, A.J.D., 1999. Linear programming in disassembly/cluster sequence generation, *Computers and Industrial Engineering*, vol. 36, pp. 723-738.

44) Laperriere, L., and H.A. ElMaraghy, 1992. Planning of products assembly and disassembly, *Annals of the CIRP*, vol. 41, no. 1, pp. 5-9.

45) Lawler, E.L., J.K. Lenstra, A. Rinnooy Kan, and D.B. Shmoys, 1985. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley and Sons, New York.

46) Low, M.K., D. Williams, and C. Dixon, 1996. Choice of end-of-life product management strategy: A case study in alternative telephone concepts, *Proceedings of the 1996 IEEE International Symposium on electronics and the Environment*, pp. 83-87.

47) Lund, R.T., 1984. Remanufacturing, *Technology Review*, February/March issue, pp. 19-29.

48) McGlothin, S., and E. Kroll, 1995. Systematic estimation of disassembly difficulties: Application to computer monitors, *Proceedings of the IEEE International Symposium on electronics and the Environment*, pp. 83-87.

49) Meacham, A., R. Uzsoy, and U. Venkatadri, 1999. Optimal disassembly configurations for single and multiple products, *Journal of Manufacturing Systems*, vol. 18, no. 5, pp. 311-322.

50) Miller, C., A. Tucker, and R. Zemlin, 1960. Integer programming formulation of traveling salesman problems. *J. ACM*, vol. 7, pp. 326-329.

51) Molloy, E., H. Yang, and J. Brown, 1991. Design for assembly within concurrent engineering, *Annals of the CIRP*, vol. 40, no. 1, pp. 107-110.

52) Moore, A.H., and K.N. Anhalt, 1995. Manufacturing for reuse, *Fortune*, February 6 issue, pp. 102-112.

53) Moore, K.E., A. Gungor, and S.M. Gupta, 1998. A petri net approach to disassembly process planning, *Computers and Industrial Engineering*, vol. 35, no. 1, pp. 165-168.

54) Moyer, L. and S.M. Gupta, 1997. Environmental concerns and recycling/disassembly efforts in the electronics industry, Journal of Electronics Manufacturing, vol. 7, no. 1, pp. 1-22.

55) Navin-Chandra, D., 1994. The recovery problem in product design, *Journal of Engineering Design*, vol. 5, no. 1, pp. 65-86.

56) Navin-Chandra, D., and R. Bansal, 1994. The recover problem: The tradeoff between designing for environmental compatibility and commercial viability, *International Journal of Environmentally Conscious Design and Manufacturing*, vol. 3, no. 2, pp. 67-71.

57) Olesen, J., and T. Keldmann, 1994. Design for environment: A framework, *Journal of Engineering Design*, vol. 5, no. 1, pp. 45-54.

58) Padberg, M., and T. Sung, 1992. An analytical comparison of different formulations of the traveling salesman problem. *Mathematical Programming*, vol. 52, 315-357.

59) Penev, K.D., and A.J. de Ron, 1996. Determination of a disassembly strategy, *International Journal of Production Research*, vol. 34, no. 2, pp. 495-506.

60) Pu, P., and L. Purvis, 1995. Assembly planning using case adaptation methods, Proceedings of the IEEE International Conference on Robotics and Automation, pp. 982-987.

61) Renich, C.R.Jr., 1991. First recylable appliance, *Appliance Manufacturer*, June issue.

62) Rose, C.M., 2000. *Design for environment: a method for formulating product end-of-life strategies*, Doctoral Dissertation, Department of Mechanical Engineering, Stanford University.

63) Scheuring, J.F., B. Bras, and K.M. Lee, 1994. Significance of design for disassembly in integrated disassembly and assembly processes, *International Journal of Environmentally Conscious Design and Manufacturing*, vol. 3, no. 2, pp. 21-33.

64) Sherali, H.D., and W.P. Adams, 1990. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, vol. 3, pp. 411-430.

65) Sherali, H.D., and W.P. Adams, 1994. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems. *Discrete Applied Mathematics*, vol. 52, pp. 83-106.

66) Sherali, H.D., and W.P. Adams, 1999. *A Reformulation-Linearization technique for solving discrete and continuous nonconvex problems*, Kluwer Academic Publishers, The Netherlands.

67) Sherali, H.D., and P.J. Driscoll, 2002. On tightening the relaxations of Miller-Tucker-Zemlin formulations for asymmetric traveling salesman problems. *Operations Res*earch, vol. 50, no. 4, pp. 656-669.

68) Sherali, H.D., S.C. Sarin, and M. Kodialam, 1990. A mathematical programming approach for a two-stage production process: Applications, models and algorithms, *International Journal of Production Planning and Control*, vol. 1, no. 1, pp. 27-39.

69) Shu, L.H., and W. Flowers, 1995. Considering remanufacture and other end-of-life options in selection of fastening and joining methods, *Proceedings of the 1995 IEEE International Symposium on Electronics and the Environment*, pp. 75-80.

70) Steinhilper, R., 1994. Design for recycling and remanufacturing of mechatronic and electronic products: Challenges, solutions and practical examples from the European viewpoint, *Design for Manufacturability*, DE-vol. 67, ASME, pp. 65-76.

71) Stuart, J.A., J.C. Ammons, L.J. Turbini, F.M. Saunders, and M. Saminathan, 1995. Evaluation approach for environmental impact and yield tradeoffs for electronics manufacturing product and process alternatives, *Proceedings of the 1995 IEEE International Symposium on Electronics and the Environment*, pp. 166-170.

72) Subramani, A.K., and P. Dewhurst, 1991. Automatic generation of product disassembly sequence, *Annals of the CIRP*, vol. 40, no. 1, pp. 115-118.

73) Thurston, D.L., 1994. Internalizing environmental impacts in design, *Design for Manufacturability*, DE-vol. 67, ASME, pp. 107-113.

74) Tipnis, V.A., 1994. Challenges in product strategy, product planning, and technology development for product life-cycle design, *Annals of the CIRP*, vol. 43, no. 1, pp. 157-162.

75) Van Der Horst, T.J.J., and A. Zweers, 1994. Environmentally oriented product development – various approaches to success, *Journal of Engineering Design*, vol. 5, no. 1, 37-44.

76) Zussman, E., A. Kriwet, and G. Seliger, 1994. Disassembly-oriented assessment methodology to support design for recycling, *Annals of the CIRP*, vol. 43, no. 1, pp. 9-14.

APPENDIX A

ATSP Revisited

1. Introduction

The traveling salesman problem (**TSP**) is perhaps the most widely researched combinatorial optimization problem. The TSP can be stated as follows: given a finite set of cities $N = \{1,2,\ldots,n\}$ and the cost of travel c_{ij} between each pair of cities $i, j \in N$, find a tour that visits each city exactly once, while minimizing the total cost of travel. In this paper, we address the *asymmetric traveling salesman problem* (**ATSP**) for which c_{ij} and c_{ji} might differ for any pair $i, j \in N$.

Typical mathematical programming formulations for the ATSP involve the assignment constraints along with subtour elimination constraints (SECs), besides the binary restrictions on the decision variables (Dantzig and Johnson, 1954; Lawler *et al*., 1985; Miller *et al*., 1960; Padberg and Sung, 1992) We present a new formulation for ATSP based on a novel modeling of the subtour elimination constraints using a polynomial number of restrictions that imply an exponential subset of certain relaxed *Dantzig-Fulkerson-Johnson* (**DFJ**) (Dantzig and Johnson, 1954) subtour elimination constraints. We establish the validity of this formulation and show that it is tighter than a similar ATSP formulation recently proposed by Gouveia and Pires (1999, 2001) (In the sequel, the formulation by Gouveia and Pires is referred to as **RMTZ**). In addition, we provide an extension of the proposed formulation to model the *precedence constrained asymmetric traveling salesman problem* (**PCATSP**), also known as the *sequential ordering problem* (**SOP**) (Ascheuer *et al*., 1990; Balas *et al*., 1995; Escudero *et al*., 1994). As the name suggests, the PCATSP is similar to the ATSP except for some additional precedence constraints that require certain cities to be visited before others. In addition to comparisons with the RMTZ, we present computational results to compare the linear programming relaxations and the branch-and-bound effort required for our formulation with that of the lifted *Miller-Tucker Zemlin* (**MTZ**) (Miller *et al*., 1960) formulations of Desrochers and Laporte (**DL**) (1991) and Sherali and Driscoll (**SD**) (2002) (both of which imply the DFJ two-city SECs), as well as for these lifted MTZ formulations that additionally incorporate the DFJ three-city SECs.

2. Proposed Formulation: ATSPxy

To present the new formulation for the ATSP, let us designate city 1 as the base city, and define the familiar assignment binary variables $x_{ij}, \forall\, i, j \in N, i \neq j$, as follows:

$$x_{ij} = \begin{cases} 1, \text{ if city } i \text{ precedes city } j \text{ immediately in a tour} \\ 0, \text{ otherwise.} \end{cases}$$

In addition, we define another set of continuous variables $y_{ij}, \forall\ i, j = 2,\ldots,n, i \neq j$, which nonetheless have a binary connotation, as follows:

$$y_{ij} = \begin{cases} 1, \text{ if city } i \text{ precedes city } j \text{ (not necessarily immediately) in a tour} \\ 0, \text{ otherwise.} \end{cases}$$

Accordingly, consider the following model for the ATSP, which is justified in the sequel.

ATSPxy: Minimize $\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} c_{ij} x_{ij}$

subject to

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} x_{ij} = 1, \forall j = 1,...,n \tag{1}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij} = 1, \forall i = 1,...,n \tag{2}$$

$$y_{ij} \geq x_{ij}, \forall i, j = 2,...,n, i \neq j \tag{3}$$

$$y_{ij} + y_{ji} = 1, \forall i, j = 2,...,n, i \neq j \tag{4}$$

$$y_{ij} + y_{jk} + y_{ki} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \tag{5}$$

$$x_{ij} \text{ binary}, \forall\ i, j = 1,\ldots,n,\ i \neq j \tag{6}$$

$$y_{ij} \geq 0, \forall\ i, j = 2,\ldots,n,\ i \neq j \tag{7}$$

Constraints (1) and (2) are the usual assignment constraints. Constraints (3)-(5) prevent the occurrence of subtours and together with the other restrictions, provide a valid formulation for the ATSP as shown below in Propositions 1 and 2. Furthermore, Proposition 3 later establishes that a valid formulation for SOP can be obtained by adding the following set of constraints, which serve to impose the required precedence relationships:

$$y_{ij} = 1, \forall j = 2,...,n, \forall i \in SPC_j, \tag{8}$$

where SPC_j is the set of cities in $\{2,\ldots,n\}$ that are required to precede city j in the Hamiltonian path that commences at the base city 1. Note that in the context of precedence constrained sequential ordering problems, the base city is typically a dummy entity that represents a starting point.

Proposition A.1. *Consider any Hamiltonian tour on the cities in N that is represented by the corresponding assignment solution* $\mathbf{x}$ *to the constraints (1), (2), and (6). Then, there exists a* $\mathbf{y}$ *such that* $(\mathbf{x}, \mathbf{y})$ *is feasible to ATSPxy.*

Proof. Without loss of generality, suppose that the given tour is 1,2,…,n, and back to 1, so that $x_{12} = x_{23} = \ldots = x_{n-1,\, n} = x_{n1} = 1$, and $x_{ij} = 0$ otherwise. Define $y_{ij} = 1$, if $i < j$, and $y_{ij} = 0$ otherwise, $\forall\, i, j = 2,\ldots,n,\ i \neq j$. Then (3) holds true since $y_{23} = \ldots = y_{n-1,\, n} = 1$, and (4) holds true because for each pair $i, j \in \{2,\ldots,n\}$, we have $\{y_{ij} = 1$ and $y_{ji} = 0\}$ if $i < j$, and $\{y_{ij} = 0$ and $y_{ji} = 1\}$ if $i > j$. Furthermore, (5) is satisfied because, if not, then there exists some triplet $i, j, k \in \{2,\ldots,n\}$ such that $y_{ij} = y_{jk} = y_{ki} = 1$, which implies that $i < j < k < i$, a contradiction. This completes the proof. □

Proposition A.2. *ATSPxy is a valid formulation for the ATSP in that there exists a feasible solution* $(\mathbf{x}, \mathbf{y})$ *if and only if* $\mathbf{x}$ *represents a Hamiltonian tour on N.*

Proof. The if-part is established by Proposition 1. Conversely, suppose that $(\mathbf{x}, \mathbf{y})$ is feasible to ATSPxy. To show that the assignment solution $\mathbf{x}$ that is feasible to (1), (2) and (6) represents a Hamiltonian tour, we need to demonstrate that $\mathbf{x}$ does not admit any subtours on a set of cities contained in $\{2,\ldots n\}$, i.e.,

$$\sum_{(i,j)\in C} x_{ij} \leq p-1 \text{ for any circuit } C \equiv \{(i_1, i_2), (i_2, i_3), \ldots\ldots, (i_p, i_1)\}, \qquad (9)$$

where $\{i_1,\ldots, i_p\} \subseteq \{2,\ldots,n\}$ and where $p \in \{2,\ldots,n\text{-}1\}$.

From (3), it is sufficient to show that:

$$\sum_{(i,j)\in C} y_{ij} \leq p-1 \text{ for any circuit } C \equiv \{(i_1, i_2), (i_2, i_3), \ldots\ldots, (i_p, i_1)\}, \qquad (10)$$

In the case of such circuits involving two or three cities, this directly follows from constraints (4) and (5) respectively. Hence, by induction, suppose that the result holds for any circuit involving m cities, and consider the case of (m+1) cities, $\{i_1, i_2, \ldots, i_{m+1}\}$, where $3 \le m \le n - 2$, and let us show that

$$y_{i_1 i_2} + y_{i_2 i_3} + \ldots + y_{i_m i_{m+1}} + y_{i_{m+1} i_1} \le m \tag{11}$$

Note that by induction hypothesis, we have

$$y_{i_1 i_2} + y_{i_2 i_3} + \ldots + y_{i_m i_1} \le m - 1 \tag{12}$$

Hence, adding (12) to $y_{i_1 i_m} + y_{i_m i_{m+1}} + y_{i_{m=1} i_1} \le 2$ from (5), we get

$$y_{i_1 i_2} + y_{i_2 i_3} + \ldots\ldots\ldots + y_{i_m i_1} + y_{i_1 i_m} + y_{i_m i_{m+1}} + y_{i_{m+1} i_1} \le m + 1 \tag{13}$$

Using the fact that $y_{i_m i_1} + y_{i_1 i_m} = 1$ from (4), the inequality (13) reduces to (11), and this completes the proof. □

Proposition A.3. *Consider the model PCATSPxy given by ATSPxy augmented by the constraints (8). Then PCATSPxy is a valid formulation of the ATSP that enforces the precedence relationships, namely, there exists a feasible solution (**x, y**) to PCATSPxy if and only if **x** represents a Hamiltonian tour on N for which city i precedes city j for each* $i \in SPC_j$, $j \in \{2, \ldots, n\}$.

Proof. The if-part follows from the proof in Proposition 1, noting that (8) is satisfied for the stated ***y*** solution, given that ***x*** represents a Hamiltonian tour that satisfies the precedence relationships. Conversely, suppose that (***x, y***) is feasible to PCATSPxy. By Proposition 2, we have that ***x*** represents a Hamiltonian tour on N. Hence, all we need to show is that for any $i \in SPC_j, j \in \{2,\ldots,n\}$, we have that city i precedes city j in this tour. By contradiction, suppose not, i.e., suppose that the tour starting and ending at the base city 1 contains the path $\{j, k_1,\ldots,k_p, i\}$. Note that we must have $p \geq 1$, because otherwise, we would have $x_{ji} = 1$, which by (3) and (4) would imply that $y_{ji} = 1$ and $y_{ij} = 0$, contradicting (8). But $x_{jk_1} = \ldots = x_{k_p i} = 1$ on this path implies from (3) along with $y_{ij} = 1$ that

$$y_{jk_1} + \ldots + y_{k_p i} + y_{ij} = p + 2,$$

which contradicts (10) in Proposition (2) based on the circuit $C \equiv \{(j, k_1),\ldots,(k_p, i), (i, j)\}$ involving (p+2) edges. This completes the proof. □

Remark 1: The Dantzig- Fulkerson-Johnson formulation (Dantzig *et al*., 1954) of the ATSP models the SECs through the following set of constraints:

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1, \forall S \subseteq N \text{ with } 2 \leq |S| \leq n-2 \tag{14}$$

We call this **ATSP-DFJ** formulation. From the proof of Proposition 2, it is clear that our formulation ATSPxy implies a relaxed subset of the DFJ constraints of the type:

$$\sum_{(i,j) \in C} x_{ij} \leq |S| - 1, \forall S \subseteq N \setminus \{1\} \text{ with } 2 \leq |S| \leq n-2 \text{, and } \forall \text{ circuits C involving all the cities in S.} \tag{15}$$

Thus, a polynomial number of constraints in ATSPxy formulation capture an exponential number of, albeit, relaxed DFJ constraints. In particular, note that (15) includes the two-city DFJ constraints involving pairs of cities in $N \setminus \{1\}$. However, for $|S| = 2$, (14) also involves pairs of the type {1, j}, j = 2,…,n. Hence, in order to tighten the LP relaxation of ATSPxy, we can include the constraints

$$x_{1j} + x_{j1} \leq 1, \forall j = 2,...,n \tag{16}$$

in the model formulation. Our computations in the following section include these constraints as well.

Remark 2: Note that ATSPxy formulation uses the *y*-variables to convey a sequencing order much like the *u*-variables in the Miller-Tucker-Zemlin (Miller *et al*., 1960) formulation (and its lifted versions as discussed in Desrochers and Laporte (1991) and Sherali and Driscoll (2002). Moreover, it permits a complete polynomial length formulation (unlike ATSP-DFJ) that, as demonstrated in the following section, affords a significantly tighter LP relaxation than the MTZ model. This is useful in the modeling of more complex routing and production problems that include ATSP as only a part of their structure. Furthermore, similar to the MTZ formulation, the ATSPxy has an inherent structure that readily facilitates the formulation of precedence constraints via the restrictions (8). We used a similar structure to obtain the formulation DOM2 of the disassembly optimization problem in chapter 3.

Remark 3: Another formulation (RMTZ) that models the SECs through a reformulation of MTZ constraints has been proposed by Gouveia and Pires (1999, 2001). This formulation includes the assignment constraints (1) and (2) and the variable restrictions (6) and (7) along with the SECs in the following form:

$$x_{ij} + y_{ki} \leq y_{kj} + 1, \forall i, j, k = 2,...,n, i \neq j \neq k \tag{17}$$

$$y_{ij} \geq x_{ij}, \forall i, j = 2,...,n, i \neq j \quad (18)$$

$$x_{ij} + y_{ji} \leq 1, \forall i, j = 2,...,n, i \neq j \quad (19)$$

Note that under (4) of ATSPxy, (18) is the same as (19), and is therefore implied. Moreover, (17) then is of the form

$$y_{ki} + x_{ij} + y_{jk} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (20)$$

However, constraints (5) of ATSPxy, together with constraints (3) imply (20). Hence, ATSPxy is potentially tighter than RMTZ.

Further, Gouveia and Pires suggest the following two possible liftings of (17):

$$x_{ij} + x_{ji} + y_{ki} \leq y_{kj} + 1, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (21)$$

$$x_{ij} + x_{kj} + x_{ik} + y_{ki} \leq y_{kj} + 1, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (22)$$

In the sequel, we refer to the formulation obtained by replacing (17) with (21) as L1RMTZ and the one obtained by replacing (17) with (22) as L2RMTZ. Note that under (4) of ATSPxy, (21) is of the form

$$y_{ki} + (x_{ij} + x_{ji}) + y_{jk} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (23)$$

However, constraints of the form

$$y_{ki} + (y_{ij} + x_{ji}) + y_{jk} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (24)$$

are valid for ATSPxy and together with (3), they imply (23); hence, they are potentially tighter. The validity of (24) follows from the fact that when $x_{ji} = 0$, (24) is same as (5) which is valid, and when $x_{ji} = 1$, (3) imply that $y_{ji} = 1$ and consequently, it follows from (4) that $y_{ij} = 0$ and hence, yjk + $y_{ki} \leq 1$, since k either comes before of after the pair (j, i) in the tour. Similarly, the following two constraints are valid for ATSPxy:

$$y_{ij} + y_{jk} + y_{ki} + x_{kj} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (25)$$

$$y_{ij} + y_{jk} + y_{ki} + x_{ik} \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (26)$$

Consequently, the following constraint obtained by surrogating (24), (25) and (26) is valid for ATSPxy:

$$3*(y_{ij} + y_{jk} + y_{ki}) + x_{ji} + x_{kj} + x_{ik} \leq 6, \forall i, j, k = 2,...,n, i \neq j \neq k \quad (27)$$

Also, note that under (4), (22) is of the form

$$(y_{ki} + x_{ik}) + x_{ij} + (y_{jk} + x_{kj}) \leq 2, \forall i, j, k = 2,...,n, i \neq j \neq k, \quad (28)$$

which is a valid for ATSPxy.

In the sequel, we refer to the formulation obtained by replacing (5) with (24) as L1ATSPxy, the one obtained by replacing (5) with (27) as SL1ATSPxy and the one obtained by replacing (5) with (28) as L2ATSPxy. Additionally, we can replace (5) with one of the constraints (24), (25) or (26) depending on which of the variables x_{ji}, x_{kj} or x_{ik} has the smallest cost associated with it and hence, has the greatest tendency to be 1 in the solution. We refer to the formulation obtained by this replacement as ML1ATSPxy.

3. Computational Results

In this section, we present computational results in order to compare the LP relaxation solution quality and the effort required by the branch-and-bound methodology using CPLEX 7.0 for the proposed formulation versus some alternative formulations presented in the literature. Standard test problems from the TSP library have been used for experimentation. For all problems, we set a time limit of 100,000 seconds.

First, we compare the LP relaxation bounds of lifted ATSPxy formulations with those of the corresponding lifted RMTZ formulations. Note that although ATSPxy is a potentially tighter formulation as compared to RMTZ, same LP relaxation bounds were obtained for the two formulations (without lifting) for the selected problems we used in our experimentation. However, we obtained tighter bounds for the lifted ATSPxy formulations as compared to the corresponding lifted RMTZ formulations for most problems. In Table A.1, we compare the bounds of L1RMTZ with those obtained using L1ATSPxy, SL1ATSPxy and ML1ATSPxy. Note that L1RMTZ does not include constraints (16). In order to demonstrate that the tighter bounds of lifted ATSPxy formulation are not solely due to the presence of constraints (16), we also include the bounds obtained as a result of adding (16) to L1RMTZ. In Table A.1, we refer to this formulation as L1RMTZ + (16). In Table A.2, we compare the bounds obtained by L2RMTZ with L2ATSPxy. Again, we include the bounds obtained as a result of adding constraints (16) to L2RMTZ and refer to this formulation as L2RMTZ + (16). The relaxed problems were solved using the primal simplex method, which appeared to be faster than the dual simplex or the barrier method in this particular case.

Problem	Formulation	Z_{LP}	CPU (LP)
br17 $Z_{IP} = 39$	L1RMTZ	18.00	3.9
	L1RMTZ+ (16)	22.00	5.1
	L1ATSPxy	22.00	16.5
	MLIATSPxy	22.00	5.9
	SL1ATSPxy	22.00	5.9
ftv33 $Z_{IP} = 1286$	L1RMTZ	1226.25	3.7
	L1RMTZ+ (16)	1226.25	4.0
	L1ATSPxy	1229.08	79.5
	MLIATSPxy	1229.08	50.5
	SL1ATSPxy	1226.99	141.7
ftv35 $Z_{IP} = 1473$	L1RMTZ	1425.34	4.9
	L1RMTZ+ (16)	1426.90	4.9
	L1ATSPxy	1427.00	211.7
	MLIATSPxy	1427.00	71.6
	SL1ATSPxy	1426.78	473.8
ftv38 $Z_{IP} = 1503$	L1RMTZ	1461.08	6.0
	L1RMTZ+ (16)	1461.08	4.3
	L1ATSPxy	1463.50	376.7
	MLIATSPxy	1463.50	70.8
	SL1ATSPxy	1460.56	193.7
ftv44 $Z_{IP} = 1613$	L1RMTZ	1580.88	12.2
	L1RMTZ+ (16)	1582.00	10.0
	L1ATSPxy	1582.00	543.9
	MLIATSPxy	1582.00	353.6
	SL1ATSPxy	1581.43	554.6
ry48p $Z_{IP} = 14422$	L1RMTZ	13837.67	9.1
	L1RMTZ+ (16)	13837.67	11.0
	L1ATSPxy	13849.08	894.2
	MLIATSPxy	13842.17	1328.4
	SL1ATSPxy	13839.51	1822.6
ftv64 $Z_{IP} = 1839$	L1RMTZ	1765.79	81.1
	L1RMTZ+ (16)	1765.79	117.9
	L1ATSPxy	1769.11	43602.8
	MLIATSPxy	1766.28	7742.8
	SL1ATSPxy	1766.28	10721.4

Table A.1 - Comparison of LP Relaxation Values for L1ATSPxy with L1RMTZ

Problem	Formulation	Z_{LP}	CPU (LP)
br17 Z_{IP} = 39	L2RMTZ	28.00	7.0
	L2RMTZ+(16)	28.00	7.4
	L2ATSPxy	28.00	19.9
ftv33 Z_{IP} = 1286	L2RMTZ	1286.00	8.3
	L2RMTZ+(16)	1286.00	10.6
	L2ATSPxy	1286.00	10.6
ftv35 Z_{IP} = 1473	L2RMTZ	1453.53	6.9
	L2RMTZ+(16)	1456.06	7.7
	L2ATSPxy	1456.89	8.8
ftv38 Z_{IP} = 1503	L2RMTZ	1481.29	6.8
	L2RMTZ+(16)	1481.29	6.9
	L2ATSPxy	1484.00	9.6
ftv44 Z_{IP} = 1613	L2RMTZ	1583.38	11.4
	L2RMTZ+(16)	1584.88	10.0
	L2ATSPxy	1584.88	21.3
ry48p Z_{IP} = 14422	L2RMTZ	13859.72	14.2
	L2RMTZ+(16)	13859.72	11.1
	L2ATSPxy	13891.00	32.0
ftv64 Z_{IP} = 1839	L2RMTZ	1778.40	96.2
	L2RMTZ+(16)	1781.01	128.9
	L2ATSPxy	1785.95	192.1

Table A.2 - Comparison of LP Relaxation Values of L2ATSPxy with L2RMTZ

From Tables 1 and 2, we observe that the lifted ATSPxy formulations produce tighter bounds as compared to the corresponding lifted RMTZ formulations. Also, note that L1ATSPxy appears to perform better than SL1ATSPxy and ML1ATSPxy. Moreover, the bounds for "L2" formulations are much tighter as compared to those for corresponding "L1" formulations.

Next, we compare the results for L2ATSPxy with those of:

a. *MTZ lifted formulation of Desrochers and Laporte* (1991). We refer to this formulation as ATSP-DL.

b. *MTZ lifted formulation of Sherali and Driscoll* (2002) *based on the Reformulation-Linearization Technique (**RLT**)* (Sherali and Adams, 1990). We refer to this formulation as ATSP-SD.

c. *The aforementioned MTZ lifted formulations augmented with the DFJ three-city SECs*. We refer to these formulations as (ATSP-DL + 3DFJ) and (ATSP-SD + 3DFJ).

In the LP relaxation comparison, we have also included the bounds obtained from the standard Miller-Tucker-Zemlin (MTZ) formulation. All problems were solved using the primal simplex method, which seemed to be faster than dual simplex or barrier method in most cases. The results obtained are presented in Tables 3 and 4.

Problem	Formulation	Z_{LP}	CPU (LP)
br17 Z_{IP} = 39	L2ATSPxy	28.00	19.9
	ATSP-MTZ	2.25	0.2
	L2RMTZ	28.00	7.0
	ATSP-DL	22.00	0.6
	ATSP-SD	27.68	0.4
	ATSP-DL+3DFJ	28.00	0.3
	ATSP-SD+3DFJ	30.57	0.7
ftv33 Z_{IP} = 1286	L2ATSPxy	1286.00	10.6
	ATSP-MTZ	1187.73	0.3
	L2RMTZ	1286.00	8.3
	ATSP-DL	1217.18	0.6
	ATSP-SD	1224.50	5.6
	ATSP-DL+3DFJ	1270.72	2.0
	ATSP-SD+3DFJ	1272.33	3.5
ftv35 Z_{IP} = 1473	L2ATSPxy	1456.89	8.8
	ATSP-MTZ	1382.86	0.3
	L2RMTZ	1453.53	6.9
	ATSP-DL	1413.50	2.4
	ATSP-SD	1415.51	8.3
	ATSP-DL+3DFJ	1448.29	2.4
	ATSP-SD+3DFJ	1448.67	4.2
ftv38 Z_{IP} = 1503	L2ATSPxy	1484.00	9.6
	ATSP-MTZ	1426.74	0.4
	L2RMTZ	1481.29	6.8
	ATSP-DL	1456.34	2.3
	ATSP-SD	1458.22	10.2
	ATSP-DL+3DFJ	1478.00	2.8
	ATSP-SD+3DFJ	1478.00	4.2
ftv44 Z_{IP} = 1613	L2ATSPxy	1584.88	21.3
	ATSP-MTZ	1523.40	0.8
	L2RMTZ	1583.38	11.4
	ATSP-DL	1573.75	2.5
	ATSP-SD	1573.75	12.2
	ATSP-DL+3DFJ	1584.63	4.3
	ATSP-SD+3DFJ	1584.63	5.5
ry48p Z_{IP} = 14422	L2ATSPxy	13891.00	32.0
	ATSP-MTZ	12564.70	1.7
	L2RMTZ	13859.72	14.2
	ATSP-DL	13809.17	2.8
	ATSP-SD	13820.43	21.2
	ATSP-DL+3DFJ	13812.05	5.8
	ATSP-SD+3DFJ	13824.66	29.1
ftv64 Z_{IP} = 1839	L2ATSPxy	1785.95	192.1
	ATSP-MTZ	1722.89	5.4
	L2RMTZ	1778.40	96.2
	ATSP-DL	1761.00	8.4
	ATSP-SD	1775.73	139.1
	ATSP-DL+3DFJ	1771.66	36.6
	ATSP-SD+3DFJ	1775.73	130.4

Table A.3 - Comparison of LP Relaxation Values for the Various ATSP Formulations

Problem	Formulation	Best Integer Solution	CPU (IP)	B&B Nodes
br17 $Z_{IP} = 39$	L2ATSPxy	39	22424.3	66251
	L2RMTZ	39	95670.3	623434
	ATSP-DL	39	4573.6	1122848
	ATSP-SD	39	624.1	6965
	ATSP-DL+3DFJ	39	2246.8	327452
	ATSP-SD+3DFJ	39	55.2	682
ftv33 $Z_{IP} = 1286$	L2ATSPxy	1286	18.8	0
	L2RMTZ	1286	20.7	0
	ATSP-DL	1286	16.1	235
	ATSP-SD	1286	928.9	2704
	ATSP-DL+3DFJ	1286	37.0	115
	ATSP-SD+3DFJ	1286	126.6	37
ftv35 $Z_{IP} = 1473$	L2ATSPxy	1473	4457.6	89
	L2RMTZ	1473	1324.8	76
	ATSP-DL	1473	43.0	1424
	ATSP-SD	1473	2068.8	3594
	ATSP-DL+3DFJ	1473	54.3	312
	ATSP-SD+3DFJ	1473	870.7	1205
ftv38 $Z_{IP} = 1503$	L2ATSPxy	1503	8905.1	36
	L2RMTZ	1503	746.2	49
	ATSP-DL	1503	101.3	3081
	ATSP-SD	1503	10841.8	7590
	ATSP-DL+3DFJ	1503	439.2	2980
	ATSP-SD+3DFJ	1503	3770.6	3684
ftv44 $Z_{IP} = 1613$	L2ATSPxy	1613	51962.1	82
	L2RMTZ	1613	7801.6	159
	ATSP-DL	1613	295.4	6554
	ATSP-SD	1613	2381.2	2111
	ATSP-DL+3DFJ	1613	421.1	1393
	ATSP-SD+3DFJ	1613	3105.3	1194

Table A.4 – Comparison of CPLEX 7.0 Branch & Bound Effort for Various ATSP Formulations

It can be observed from Table A.3 that in all cases (except br17), ATSPxy gave better LP relaxation bounds as compared to all other formulations. Also, from Table A.4, we observe that although L2ATSPxy takes a longer time to find optimal integer solutions in some cases, it requires a significantly lower number of branch-and-bound nodes to be explored to reach optimality. The longer computational time can be attributed to the greater effort required for solving the LP relaxation at each node. We expect that with ongoing improvements in solving linear programming problems, this will not be a major limitation of the proposed formulation.

Finally, we provide the results for precedence constrained ATSP (PCATSP) problems in Tables 5 and 6. Again, we used selected sequential ordering problems from the TSP library for experimentation. Note that all of the lifted ATSPxy formulations (namely, L1ATSPxy, SL1ATSPxy, ML1ATSPxy and L2ATSPxy) augmented with constraints (8) result in valid PCATSP formulations. Tighter representations are obtained by incorporating the following valid constraints in the precedence constrained formulations:

$$x_{ik} = 0, \forall k = 2,...,n, j \in SPC_k, i \in SPC_j \qquad (29)$$

Constraints (29) are valid because if city *j* must be visited before city *k*, and city *i* must be visited before city *j*, then city *k* cannot be visited immediately after city *i*; city *j* must be visited in the sequence between *i* and *k*.

We call the precedence constrained ATSPxy formulation as PCATSPxy. The names of the lifted formulations are changed similarly and are self-explanatory. We compared the LP relaxation values and the branch-and-bound effort required for these formulations with the precedence constrained formulations of Sherali and Driscoll (2002) and Desrochers and Laporte (1991). We refer to these formulations as PCATSP-SD and PCATSP-DL, respectively. The dual simplex method was found to be faster in this case; hence it was used to obtain the LP relaxation bounds.

Problem	Formulation	Z_{LP}	CPU (LP)
esc07 Z_{IP} = 2125	L1PCATSPxy	2125.00	0.1
	L2PCATSPxy	2125.00	0.2
	SL1PCATSPxy	2028.13	0.1
	ML1PCATSPxy	2012.50	0.2
	PCATSP-SD	2087.5	0.1
	PCATSP-DL	2005.83	0.1
esc11 Z_{IP} = 2075	L1PCATSPxy	2058.83	0.3
	L2PCATSPxy	2050.75	0.3
	SL1PCATSPxy	2050.75	0.3
	ML1PCATSPxy	2054.75	0.2
	PCATSP-SD	2036.90	0.3
	PCATSP-DL	2032.53	0.1
esc12 Z_{IP} = 1675	L1PCATSPxy	1554.00	0.4
	L2PCATSPxy	1528.50	0.4
	SL1PCATSPxy	1515.40	0.3
	ML1PCATSPxy	1507.50	0.3
	PCATSP-SD	1520.35	0.4
	PCATSP-DL	1508.52	0.2
br17.10 Z_{IP} = 55	L1PCATSPxy	27.29	2.3
	L2PCATSPxy	28.00	1.1
	SL1PCATSPxy	18.00	0.7
	ML1PCATSPxy	18.00	0.8
	PCATSP-SD	22.68	0.9
	PCATSP-DL	22.00	0.2
esc25 Z_{IP} = 1681	L1PCATSPxy	1610.96	17.3
	L2PCATSPxy	1591.00	7.5
	SL1PCATSPxy	1585.14	6.9
	ML1PCATSPxy	1601.00	6.5
	PCATSP-SD	1540.32	2.9
	PCATSP-DL	1529.87	0.3
prob42 Z_{IP} = 243	L1PCATSPxy	232.50	718.1
	L2PCATSPxy	223.62	248.6
	SL1PCATSPxy	223.83	160.4
	ML1PCATSPxy	223.00	202.9
	PCATSP-SD	225.49	36.2
	PCATSP-DL	225.20	1.0

Table A.5 - Comparison of LP Relaxation Values for the Various PCATSP Formulations

From Table A.5, we observe that L1PCATSPxy produced tighter bounds than the other lifted PCATSPxy formulations (except br17.10) as well as PCATSP-SD and PCATSP-DL; hence it was used for comparing the branch-and-bound effort with PCATSP-SD and PCATSP-DL.

Problem	Formulation	Best Integer Solution	CPU (IP)	B&B Nodes
esc07 Z_{IP} = 2125	L1PCATSPxy	2125	0.1	0
	PCATSP-SD	2125	0.2	0
	PCATSP-DL	2125	0.1	1
esc11 Z_{IP} = 2075	L1PCATSPxy	2075	0.3	0
	PCATSP-SD	2075	0.3	0
	PCATSP-DL	2075	0.1	0
esc12 Z_{IP} = 1675	L1PCATSPxy	1675	6.5	83
	PCATSP-SD	1675	17.1	520
	PCATSP-DL	1675	1.1	133
br17.10 Z_{IP} = 55	L1PCATSPxy	55	580.4	1842
	PCATSP-SD	55	8536.9	175284
	PCATSP-DL	55	25354.1	5877686
esc25 Z_{IP} = 1681	L1PCATSPxy	1681	487.4	69
	PCATSP-SD	1681	281.7	1107
	PCATSP-DL	1681	72.1	5074

Table A.6 – Comparison of CPLEX 7.0 Branch-and-Bound Effort for the Various PCATSP Formulations

It is evident from the results in Table A.6 that in case of L1PCATSPxy, significantly lower number of branch-and-bound nodes are explored before reaching the optimal integer solution. Also, we observe that in case of br17.10, where the CPU time required to obtain the optimal solution is relatively large, L1ATSPxy converges to optimum much faster as compared to PCATSP-SD and PCATSP-DL. Note that for br17.10, the precedence graph was randomly generated (as against "esc" problems, which are real life instances). This highlights the usefulness of L1PCATSPxy for complex precedence constrainted ATSP problems.

4. Conclusion

We have presented new valid formulations for ATSP and precedence constrained ATSP using polynomial number of restrictions that imply an exponential subset of certain relaxed Dantzig-Fulkerson-Johnson subtour elimination constraints. We have demonstrated that the proposed formulations, when appropriately lifted, are tighter than the other well-known formulations available in the literature. It would be of interest to study the efficiency of the proposed formulations further for solving more complex routing and production scheduling problems, which have ATSP as an embedded substructure.

APPENDIX B

DATA SETS AND PARAMETER VALUES FOR THE CASE STUDY ON DISASSEMBLY OF A COMPUTER PROCESSOR

In the first part of this appendix, we specify the elements of various sets (such as sets of preceding and succeeding joints of a part/subassembly) used in our model. This information is directly taken from our AMPL model and is in AMPL format; however, the contents are self-explanatory and easy to understand.

We used randomly generated data to be used in our case study on the disassembly of a computer processor. In order to establish the validity of our optimization model, we need to verify two things:

1) the precedence relationships are satisfied in the optimal solution, and
2) the model actually gives the true optimum solution.

The first condition can be verified directly by looking at the optimal solution. In order to verify the second condition, we arbitrarily assign relatively high recovery values to a particular part/subassembly. In that case, we would expect this parts/subassembly to be one of the terminal nodes in our optimal solution. The optimum solution for our model is then checked to verify whether this part/subassembly is actually recovered.

In order to accomplish the aforesaid tasks, we adopt the following testing procedure. Firstly, we randomly generate 2 data sets. Then, for each of the data sets thus generated, we have 3 subcategories. In the first subcategory, we solve the problem with our original data. In the second and third subcategories, we modify the data to assign typically large recovery value to a certain part/subassembly. Besides this modification, the other data is the same as in original data set. All the programs are run on a Windows operating system with ILOG AMPL version 10.6.16 – Win 32 and CPLEX 7.0.0. We report the computational time in CPU seconds for each of the data sets.

Description of Sets

```
set I := FA P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13 P14 P15 S1 S2 S3 S4 S5;
set K := I1 I2 I3 I4 I5 I6 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F D;
set SOS:=FA S1 S2 S3 S4 S5; # set of subassemblies
set START := F; # dummy starting joint
set DUMMY := D; # dummy last joint

set SOJ[FA]  := I1 I2 I3 I4 I5 I6 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set SOJ[P1]  := ;
set SOJ[P2]  := ;
set SOJ[P3]  := ;
set SOJ[P4]  := ;
set SOJ[P5]  := ;
set SOJ[P6]  := ;
set SOJ[P7]  := ;
set SOJ[P8]  := ;
set SOJ[P9]  := ;
set SOJ[P10] := ;
set SOJ[P11] := ;
set SOJ[P12] := ;
set SOJ[P13] := ;
set SOJ[P14] := ;
set SOJ[P15] := ;
set SOJ[S1]  := I5 I6;
```

```
set SOJ[S2]  := I3 I4 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set SOJ[S3]  := F7 F8;
set SOJ[S4]  := I3 I4 F2 F3 F4 F7 F8 F10 F11;
set SOJ[S5]  := F5;


set KP[FA]  := F;
set KP[P1]  := F I1 I5 I6;
set KP[P2]  := F I2;
set KP[P3]  := F F1;
set KP[P4]  := F I1 I2 I3 I4 F1 F2 F3 F4 F6 F9 F10 F11;
set KP[P5]  := F I1 I2 I3 F1 F2 F3 F4 F6;
set KP[P6]  := F I1 I2 I4 F1 F2 F3 F4 F6;
set KP[P7]  := F I1 I2 F1 F2 F3 F4 F6 F10;
set KP[P8]  := F I1 I2 F1 F2 F3 F4 F6 F11;
set KP[P9]  := F I1 I2 F1 F2 F3 F4 F5 F6 F9 F11;
set KP[P10] := F I1 I2 F1 F2 F3 F4 F5 F6 F9 F11;
set KP[P11] := F I1 I2 F1 F2 F3 F4 F6 F7;
set KP[P12] := F I1 I2 F1 F2 F3 F4 F6 F8;
set KP[P13] := F I1 I2 I5 F1;
set KP[P14] := F I1 I2 I6 F1;
set KP[P15] := F I1 I2 F1 F2 F3 F4 F6 F7 F8;
set KP[S1]  := F I1;
set KP[S2]  := F I1 I2 F1;
set KP[S3]  := F I1 I2 F1 F2 F3 F4 F6;
set KP[S4]  := F I1 I2 F1 F6;
set KP[S5]  := F I1 I2 F1 F2 F3 F4 F6 F9;

set KS[FA]  := ;
set KS[P1]  := ;
set KS[P2]  :=  I3 I4 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
```

```
set KS[P3]  :=  I3 I4 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set KS[P4]  :=  ;
set KS[P5]  :=  ;
set KS[P6]  :=  ;
set KS[P7]  :=  ;
set KS[P8]  :=  ;
set KS[P9]  :=  ;
set KS[P10] := ;
set KS[P11] := ;
set KS[P12] := ;
set KS[P13] := ;
set KS[P14] := ;
set KS[P15] :=;
set KS[S1]  :=;
set KS[S2]  :=  ;
set KS[S3]  := I3 I4 F5 F9 F10 F11;
set KS[S4]  := ;
set KS[S5]  := ;

set P[FA]  := ;
set P[P1]  := P13 P14;
set P[P2]  := ;
set P[P3]  := ;
set P[P4]  := P5 P6 P7 P8 S5;
set P[P5]  := ;
set P[P6]  := ;
set P[P7]  := ;
set P[P8]  := ;
set P[P9]  := ;
set P[P10] := ;
set P[P11] := ;
```

```
set P[P12] := ;
set P[P13] := ;
set P[P14] := ;
set P[P15] := P7 P8 P11 P12 S5;
set P[S1]  := P2 P3 S2;
set P[S2]  := ;
set P[S3]  := ;
set P[S4]  := ;
set P[S5]  := ;

set KPM[D] :=F;
set KPM[I1]:=F;
set KPM[I2]:=F;
set KPM[I3]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[I4]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[I5]:=I1 F;
set KPM[I6]:=I1 F;
set KPM[F1]:=F;
set KPM[F2]:=I1 I2 F1 F;
set KPM[F3]:=I1 I2 F1 F;
set KPM[F4]:=I1 I2 F1 F;
set KPM[F5]:=I1 I2 F1 F2 F3 F4 F6 F9 F;
set KPM[F6]:=I1 I2 F1 F;
set KPM[F7]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[F8]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[F9]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[F10]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[F11]:=I1 I2 F1 F2 F3 F4 F6 F;
set KPM[F]:=;

set KSM[F]:=I1 I2 I3 I4 I5 I6 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
```

```
set KSM[I1]:=I3 I4 I5 I6 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set KSM[I2]:=I3 I4 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set KSM[I3]:=;
set KSM[I4]:=;
set KSM[I5]:=;
set KSM[I6]:=;
set KSM[F1]:=I3 I4 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11;
set KSM[F2]:=I3 I4 F5 F7 F8 F9 F10 F11;
set KSM[F3]:=I3 I4 F5 F7 F8 F9 F10 F11;
set KSM[F4]:=I3 I4 F5 F7 F8 F9 F10 F11;
set KSM[F5]:=;
set KSM[F6]:=I3 I4 F5 F7 F8 F9 F10 F11;
set KSM[F7]:=;
set KSM[F8]:=;
set KSM[F9]:=F5;
set KSM[F10]:=;
set KSM[F11]:=;
set KSM[D]:=;
```

Data Set 1

Subcategory1 (original data):

In order to keep the data realistic, we adopt the following procedure (all data are obtained from the uniform distribution between specified intervals and truncated to second decimal place). The values of parameter b_i are obtained in the interval 15-25 for individual parts and in interval 35-60 for subassemblies. The values of c_i are obtained in interval 5-10 for individual parts and from interval 10-15 for subassemblies. The f_k values are in the range 2-7, and the set-up cost values are between 1-20.

Part/Subassembly i	b_i	c_i
P1	20.98	7.23
P2	19.26	8.79
P3	20.51	5.79
P4	17.00	6.80
P5	16.31	9.01
P6	18.48	5.43
P7	17.40	8.16
P8	22.10	8.10
P9	15.25	9.14
P10	24.16	5.14
P11	17.07	7.12
P12	20.34	6.71
P13	17.00	7.51
P14	15.07	5.41
P15	19.12	6.25
FA	54.91	10.01
S1	59.57	10.76
S2	47.46	10.04
S3	41.88	10.42
S4	55.75	13.77
S5	46.11	10.16

Table B.1 Values of b_i and c_i – data set 1

Joint k	f_k
I1	3.47
I2	5.90
I3	5.11
I4	5.36
I5	6.56
I6	2.96
F1	3.55
F2	5.49
F3	6.82
F4	3.98
F5	3.80
F6	3.62
F7	4.32
F8	5.38
F9	4.31
F10	3.86
F11	5.64

Table B.2 Values of f_k – data set 1

l → / k ↓	I1	I2	I3	I4	I5	I6	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
I1	-	13.88	6.31	19.94	19.64	1.13	7.34	12.75	1.77	5.38	18.84	11.25	8.12	1.27	19.73	18.59	9.95
I2	9.45	-	17.73	16.11	6.73	16.98	3.25	17.89	8.52	1.66	6.62	10.72	15.60	2.51	1.34	15.04	1.37
I3	4.74	2.15	-	11.50	16.08	16.91	18.41	1.42	5.90	13.15	10.12	15.28	16.54	14.01	5.88	8.94	12.78
I4	10.56	5.62	15.67	-	14.16	3.82	14.03	18.29	11.87	11.50	6.71	14.70	15.66	6.15	11.60	12.88	6.31
I5	7.27	18.16	5.27	19.77	-	9.12	18.16	6.01	10.60	5.67	16.72	11.84	14.04	8.28	8.89	14.05	2.28
I6	18.97	18.77	11.19	4.57	17.59	-	4.29	5.55	1.73	7.02	18.98	14.11	19.73	15.07	8.18	15.82	2.55
F1	16.32	15.91	15.76	7.91	8.02	4.35	-	17.58	3.62	9.75	8.19	14.98	11.02	10.18	7.80	16.18	18.16
F2	18.87	8.29	2.48	7.64	14.12	1.86	16.41	-	10.46	8.22	8.61	10.48	15.01	17.70	1.35	19.95	2.82
F3	10.76	3.97	18.74	16.17	4.44	18.33	3.95	8.39	-	6.25	16.26	12.44	5.51	5.09	4.64	7.70	13.21
F4	16.74	9.86	8.09	10.04	7.70	14.50	11.21	5.85	6.82	-	8.50	15.54	11.43	11.28	19.00	17.00	11.19
F5	13.41	4.89	16.99	11.51	11.75	11.34	13.89	5.03	17.56	8.78	-	19.70	13.97	8.16	8.68	2.12	4.53
F6	3.37	2.27	6.78	17.77	7.33	15.50	12.66	13.82	1.84	12.54	8.92	-	10.91	9.87	13.15	7.99	3.46
F7	9.42	17.90	7.43	6.20	4.42	3.99	3.68	18.60	8.77	16.21	11.27	15.68	-	3.06	15.02	10.63	8.35
F8	5.04	15.35	6.86	17.75	13.14	18.33	1.67	12.51	6.13	2.04	3.96	12.82	1.18	-	19.25	2.30	18.51
F9	14.71	2.49	14.87	3.49	16.26	16.79	2.94	6.07	4.97	9.68	17.05	14.76	3.53	14.98	-	11.11	5.16
F10	1.85	3.26	3.00	9.16	8.12	19.86	13.61	18.37	7.56	2.45	12.33	1.82	15.13	16.82	11.55	-	12.73
F11	12.45	11.63	16.75	9.69	4.53	17.93	8.86	10.53	2.10	2.11	4.11	14.03	13.14	12.80	9.46	6.27	-

Table B.3 Values of set-up cost (s_{kl}) – data set 1

Subcategory 2:

We now assign a value of b_i = 400 for the final assembly. With such high recovery value for the final assembly, we should expect the optimal solution to be no disassembly.

Subcategory 3:

Next, we assign a value of b_i = 400 for subassembly S5. Rest of the original data remains the same. We would then expect S5 to be one of the subassemblies recovered.

Data Set 2

Subcategory 1 (original data)

For this data set, the values of b_i are obtained in the range 30-40 for the individual parts and in the range 60-70 for the subassemblies. The values of c_i are obtained in the range 10-15 for the parts and 15-25 for the subassemblies. The values of f_k are in the range 10-15 and set-up costs are in the interval 20-40. All the values are again randomly generated from uniform distribution.

Part/Subassembly i	b_i	c_i
P1	33.82	11.75
P2	31.01	10.28
P3	35.96	13.00
P4	38.99	12.21
P5	38.85	12.98
P6	39.58	11.50
P7	30.14	13.01
P8	34.07	11.11
P9	38.63	14.89
P10	31.39	13.99
P11	32.45	13.75
P12	30.45	10.52
P13	30.32	12.74
P14	31.64	13.79
P15	32.20	14.22
FA	65.98	23.96
S1	64.26	19.34
S2	65.51	20.00
S3	62.00	23.85
S4	61.31	21.19
S5	63.48	18.36

Table B.4 Values of b_i and c_i – data set 2

Joint k	f_k
I1	14.54
I2	13.39
I3	14.71
I4	12.41
I5	10.88
I6	13.67
F1	14.56
F2	10.03
F3	14.01
F4	14.08
F5	10.88
F6	12.79
F7	12.48
F8	12.49
F9	11.81
F10	13.52
F11	13.76

Table B.5 Values of f_k – data set 2

l → / k ↓	I1	I2	I3	I4	I5	I6	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
I1	-	38.60	37.69	29.12	24.03	35.81	28.70	22.11	25.14	37.70	30.77	22.24	37.02	31.40	23.28	39.13	37.57
I2	32.89	-	21.16	29.81	38.53	22.93	24.29	27.97	29.49	29.80	36.53	36.44	37.05	22.29	31.26	22.54	27.10
I3	29.24	32.37	-	31.86	23.25	24.48	37.83	26.85	22.04	31.39	23.55	21.48	29.41	35.71	30.75	21.14	21.35
I4	22.97	32.28	29.04	-	23.24	21.22	25.94	32.15	33.46	36.52	35.79	34.48	33.50	27.21	35.54	26.97	39.16
I5	25.94	37.12	34.24	25.22	-	39.82	34.05	21.63	36.85	21.10	37.27	36.03	25.89	24.02	38.09	25.62	38.49
I6	29.44	30.03	31.65	35.52	30.99	-	39.48	31.87	21.07	29.90	37.52	32.21	25.96	29.70	38.01	21.10	38.12
F1	27.14	32.90	27.89	23.83	30.17	27.85	-	34.42	39.38	27.56	25.53	38.91	22.30	35.44	34.51	26.56	38.48
F2	20.91	21.70	37.17	21.01	37.30	39.35	27.38	-	23.60	21.66	31.06	24.39	33.03	31.52	21.97	25.57	21.73
F3	26.92	36.72	21.89	20.61	38.04	26.23	26.35	23.43	-	33.79	35.00	34.36	39.51	27.84	21.05	28.02	33.71
F4	27.63	30.50	25.28	28.91	29.80	34.79	37.68	22.79	34.13	-	23.98	22.94	36.45	31.22	31.54	34.92	26.95
F5	28.82	26.04	25.86	21.95	26.96	22.36	24.12	27.29	25.34	39.93	-	21.36	38.51	35.65	28.62	36.15	25.56
F6	37.37	23.65	24.08	21.78	31.31	25.95	29.11	38.14	26.23	34.23	29.93	-	27.79	20.54	33.60	38.70	25.95
F7	20.32	27.63	28.07	28.72	23.78	28.64	30.58	25.37	38.85	35.81	29.78	32.04	-	27.11	37.98	23.18	39.78
F8	26.56	27.01	21.65	24.62	20.55	31.72	30.80	37.40	31.43	21.78	21.55	35.52	26.39	-	37.72	34.21	29.91
F9	34.69	32.49	29.37	37.72	36.12	39.46	37.19	20.51	31.90	31.10	31.06	35.56	27.70	27.93	-	34.22	24.55
F10	31.51	35.42	23.64	26.35	24.29	37.57	30.57	20.74	29.53	34.77	25.65	38.90	34.40	33.43	36.10	-	29.29
F11	39.06	25.01	35.49	26.66	32.50	25.41	20.02	32.33	28.38	27.13	22.77	32.45	26.71	24.67	26.60	29.30	-

Table B.6 Values of set-up cost (s_{kl}) – data set 2

Subcategory 2:

We now set the value b_i for part P4 (motherboard) equal to 400. We would expect P4 to be one of the recovered parts.

Subcategory 3:

Next we set b_i = 400 for part P9 (hard disk drive). Again, we would expect P9 to be one of the recovered parts in the optimal solution.

APPENDIX C

AMPL CODE FOR THE IMPLEMENTATION OF THREE - PHASE SOLUTION PROCEDURE

```
model dismod.mod;
data printer.dat;

# Lower cutoff specified below is same as Z*.
option cplex_options 'mipdisplay=5' 'mipinterval=3' 'timing=1' 'lowercutoff=15;
option omit_zero_rows 1;
option omit_zero_cols 1;

#Relaxation of constraints (3.4l) in DOM2
drop 3.4l;

#variable definition
var t>=0; #step size
var iter default 0; # maximum number of iterations
var lambda default 2;
var valcnt default 0;
var zstar default 44.9; # best known feasible solution

set ALLTOUR{1..10000} default {};
set SUBTOUR{1..10000} ordered default {} ;
set PREC{1..10000} ordered default{};
set M default {};

var ls{i in K} default 0;
var lt{i in K} default 0;
var tv{i in K,j in K} default 0;
var scount default 1;
```

```
var tcount default 0;
var iter1 default -1;
var cnt default 0;

drop DKminus1;
drop DKminus2;
drop DKminus3;

drop c00;

let iter:=0;
let lambda:=2;

repeat {
	solve;
```

#Detemination of g[i,j]

```
	for {i in K}
		for{j in K}
			if p[i,j]=1 then
				let g[i,j]:=g[i,j]+1;

	display benefit;
```

#Phase 1: Lagrangian Relaxation

```
	if benefit < temp then {
		let temp:=benefit;
		for{i in K}
			for{j in K}
				let v[i,j]:=u[i,j];
		let valcnt:=0;}
```

```
else let valcnt:=valcnt+1;

display valcnt;

display lambda;

let iter:=iter + 1;
display iter;
```

#Determination of step size (t)

```
let t:=lambda*(temp - zstar)/(sum{k in K,l in K}((p[k,l]-q[k,l])^2));
display t;
```

#Determination of penalty (u[k,l])

```
for {k in K}
  for {l in K}
        let u[k,l]:=max(0, u[k,l] + t* (p[k,l]-q[k,l]));


if valcnt=3 then{
        let lambda:=lambda/2;
        for{i in K}
                for{j in K}
                        let u[i,j]:=v[i,j];
        let valcnt:=0;}

if t< 0.1 then
        break;
```

```
        } until (iter>=10);

display temp;
for{i in K}
        for{j in K}
                let u[i,j]:=v[i,j];
let iter1:=0;

restore c00;
restore DKminus1;
restore DKminus2;
restore DKminus3;
```

#Phase 2: Addition of Valid Inequalities

```
repeat{

 repeat {

        solve;
```

#Determination of g[i,j]

```
        for {i in K}
                for{j in K}
                        if p[i,j]=1 then
                                let g[i,j]:=g[i,j]+1;

        let temp:=benefit;
        display temp;

        let scount:=1;
        for{i in K}{
```

```
            let ls[i]:=0;
            let lt[i]:=0;}

let tcount:=0;
let cnt:=0;

let iter1:=iter1+1;
display iter1;

if iter1=10 then
  break;

let temp:=benefit;
            let ls["S"]:=1;

for{i in K}
            if p["S",i]=1 then
                        let ls[i]:=ls["S"]+1;

let scount:=scount+1;

repeat{
            for {i in K, j in K:y[i]=1&&y[j]=1}{
                        if (ls[i]=scount && p[i,j]=1) then
                                    let ls[j]:=ls[i]+1;
                        if (ls[j]=scount+1) then
                                    let scount:=scount+1;}}until(ls["D"]>=1);

display ls;

if (ls["D"]=sum{k in K}y[k]) then
```

```
        break;
    else let indexs:=indexs+1;
```

#Detection of presence of a subtour and identifying the joints involved in subtour

```
    for {i in K}
        if (ls[i]=0 && y[i]=1) then
            let ALLTOUR[indexs]:=ALLTOUR[indexs] union {i};



    display ALLTOUR[indexs];
```

#Determination of actual subtour

```
    for{i in ALLTOUR[indexs]}{
        let lt[i]:=1;
        let SUBTOUR[indexs]:=SUBTOUR[indexs] union {i};
        break;}

    let tcount:=tcount+1;

    repeat{
        for {i in ALLTOUR[indexs], j in ALLTOUR[indexs]:i<>j}{
            if (lt[i]=tcount && p[i,j]=1 && lt[j]=0) then {
                let SUBTOUR[indexs]:=SUBTOUR[indexs] union {j};
                let lt[j]:=lt[i]+1;}
            if (lt[j]=tcount+1) then
                let tcount:=tcount+1;}
            let cnt:=cnt+1;}until(cnt>=card{ALLTOUR[indexs]});
```

```
	display SUBTOUR[indexs];
```

 # Adding SEC to the model

```
	let TOUR[indexs]:=SUBTOUR[indexs];

	for {i in TOUR[indexs]}
	let PTOUR[indexs]:=PTOUR[indexs] union KPM[i];

	for{h in 1..indexs}{

			for{k in TOUR[h]}
					let KTOUR[h]:={i in TOUR[h]:i in KPM[k]};

			let rhss1[h]:=max(0,sum{i in SUBTOUR[h]}y[i]-1); }

	}until (card{SUBTOUR[indexs]}=0);

	if iter1=10 then
		break;
```

#Detemination of precedence violations

```
	if card{PREC[indexp]}>0 then
		let indexp:=indexp+1;

	let M:={};
	for {i in K,j in KPM[i]:y[i]=1 && y[j]=1}
	{
		if ls[j]>ls[i] then{
```

```
                        let PREC[indexp]:=PREC[indexp] union {i};
                        let M:=M union {i};

                        repeat{
                                for {m in M}
                                        for {k in K: y[k]=1 && k<>j}
                                                if p[m,k]=1 then{
                                                        let M:=M diff {m} union {k};
                                                        let PREC[indexp]:=PREC[indexp]
union {k};}}until (card{PREC[indexp]}>=ls[j]-ls[i]);
                                let PREC[indexp]:=PREC[indexp] union {j};;}

        if ls[j]>ls[i] then
                break;
        }

        if card{PREC[indexp]}>0 then{
                let PFC[indexp]:=PREC[indexp];

                for {i in PFC[indexp]}
                        let PPFC[indexp]:=PPFC[indexp] union KPM[i];

                for{h in 1..indexp}{

                let rhsp1[h]:=sum{k in PFC[h]:k<>first(PFC[h]) &&
k<>last(PFC[h])}y[k];}}
        else
                break;

        display PREC[indexp];}until (iter1>=10)
        display benefit;
```

 # Phase 3: Variable fixing step

```
for {i in K}
        for{j in K}
                if g[i,j]=0 then
                        let p[i,j]:=0;
```

 # Restoring relaxed constraints 4l of DOM2

```
restore 4l;
```

 # Phase 3: Branch and bound implementation

```
option cplex_options 'mipdisplay=5' 'mipinterval=3' 'timing=1' 'lowercutoff =15
solve;

for {i in K}
        for{j in K}
                let u[i,j]:=0;
display benefit;
shell ‘time’;
```

APPENDIX D
DATA SETS AND PARAMETER VALUES FOR THE CASE STUDY ON DISASSEMBLY OF A LASER PRINTER

In the first part of this appendix, we specify the elements of various sets (such as sets of preceding and succeeding joints of a part/subassembly) used in our model. This information is directly taken from our AMPL model and is in AMPL format; however, the contents are self-explanatory and easy to understand.

In the next part, we provide the values of parameters in the model. The set-up costs are randomly generated; however, in this case study, we assigned other values in a realistic manner based on the type of joint and the type of part/subassembly.

Description of sets

set I := FA P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13 P14 P15 P16 P17 P18 P19 P20 P21 P22 P23 P24 P25 P26 S1 S2 S3 S4;
set K := I1 I2 I3 I4 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14 F15 F16 F17 F18 F19 F20 F21 F22 F23 F D ;
set SOS:=FA S1 S2 S3 S4; # joints within a subassembly
set DUMMY:=D;
set START:=F;

set SOJ[FA] := I1 I2 I3 I4 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14 F15 F16 F17 F18 F19 F20 F21 F22 F23;
set SOJ[P1] := ;
set SOJ[P2] := ;
set SOJ[P3] := ;

```
set SOJ[P4]  := ;
set SOJ[P5]  := ;
set SOJ[P6]  := ;
set SOJ[P7]  := ;
set SOJ[P8]  := ;
set SOJ[P9]  := ;
set SOJ[P10] := ;
set SOJ[P11] := ;
set SOJ[P12] := ;
set SOJ[P13] := ;
set SOJ[P14] := ;
set SOJ[P15] := ;
set SOJ[P16] := ;
set SOJ[P17] := ;
set SOJ[P18] := ;
set SOJ[P19] := ;
set SOJ[P20] := ;
set SOJ[P21] := ;
set SOJ[P22] := ;
set SOJ[P23] := ;
set SOJ[P24] := ;
set SOJ[P25] := ;
set SOJ[P26] := ;
set SOJ[S1]  := I2 I4 F3 F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F21 F23;
set SOJ[S2]  := I2 I4 F3;
set SOJ[S3]  := F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23;
set SOJ[S4]  := F8 F9 F12;


set KP[FA]  := F;
set KP[P1]  := F I1;
```

```
set KP[P2]  := F I3 F14;
set KP[P3]  := F I1 F1 F2 F14 F22;
set KP[P4]  := F I2 I4 F1 F2 F3 F14 F21 F22;
set KP[P5]  := F I2 F1 F2 F14 F21 F22;
set KP[P6]  := F F1 F2 F3 F14 F21 F22;
set KP[P7]  := F F1 F4 F14;
set KP[P8]  := F F5 F14;
set KP[P9]  := F F6;
set KP[P10] := F F7 F14;
set KP[P11] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P12] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P13] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P14] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P15] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P16] := F F14;
set KP[P17] := F F6 F15;
set KP[P18] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P19] := F F4 F5 F6 F7 F17;
set KP[P20] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P21] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P22] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P23] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[P24] := F I4 F1 F2 F14 F21 F22;
set KP[P25] := F I3 F14 F22;
set KP[P26] := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;
set KP[S1]  := F I1 I3 F1 F4 F5 F6 F7 F14 F15 F17 F22;
set KP[S2]  := F F1 F2 F14 F21 F22;
set KP[S3]  := F F4 F5 F6 F7 F15 F17 F21;
set KP[S4]  := F F4 F5 F6 F7 F10 F11 F13 F15 F16 F17 F18 F19 F20 F21 F23;

set KS[FA]  := ;
```

```
set KS[P1]  := ;
set KS[P2]  := ;
set KS[P3]  := ;
set KS[P4]  :=  ;
set KS[P5]  :=  ;
set KS[P6]  :=  ;
set KS[P7]  :=  ;
set KS[P8]  :=  ;
set KS[P9]  := F15  ;
set KS[P10] := ;
set KS[P11] := ;
set KS[P12] :=  ;
set KS[P13] := ;
set KS[P14] :=  ;
set KS[P15] := ;
set KS[P16] := I3 ;
set KS[P17] := ;
set KS[P18] := ;
set KS[P19] := F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23;
set KS[P20] := ;
set KS[P21] := ;
set KS[P22] := ;
set KS[P23] := ;
set KS[P24] := ;
set KS[P25] := ;
set KS[P26] := ;
set KS[S1]  := ;
set KS[S2]  := ;
set KS[S3]  := ;
set KS[S4]  := ;
```

```
set P[FA]  := ;
set P[P1]  := ;
set P[P2]  := P16;
set P[P3]  := P1;
set P[P4]  := P5;
set P[P5]  := ;
set P[P6]  := ;
set P[P7]  := P16;
set P[P8]  := P16;
set P[P9]  := ;
set P[P10] := P16;
set P[P11] := P20;
set P[P12] := P20;
set P[P13] := P20;
set P[P14] := P20;
set P[P15] := P11 P20;
set P[P16] := ;
set P[P17] := P9;
set P[P18] := P20;
set P[P19] := ;
set P[P20] := ;
set P[P21] := P20;
set P[P22] := P20;
set P[P23] := P20;
set P[P24] := ;
set P[P25] := P2 P16;
set P[P26] := P20;
set P[S1]  := P2 P3 P9 P16 P19;
set P[S2]  := P3;
set P[S3]  := P19;
set P[S4]  := P20;
```

```
set KPM[D]:=;
set KPM[I1]:=F;
Fet KPM[I2]:=I1 F1 F2 F14 F22 F;
Fet KPM[I3]:=F14 F;
Fet KPM[I4]:=I1 F1 F2 F14 F22 F;
Fet KPM[F1]:=F;
Fet KPM[F2]:=F;
Fet KPM[F3]:=I1 F1 F2 F14 F22 F;
Fet KPM[F4]:=F;
Fet KPM[F5]:=F;
Fet KPM[F6]:=F;
Fet KPM[F7]:=F;
Fet KPM[F8]:=F4 F5 F6 F7 F17 F;
Fet KPM[F9]:=F4 F5 F6 F7 F17 F;
Fet KPM[F10]:=F4 F5 F6 F7 F17 F;
Fet KPM[F11]:=F4 F5 F6 F7 F17 F;
Fet KPM[F12]:=F4 F5 F6 F7 F17 F;
Fet KPM[F13]:=F4 F5 F6 F7 F17 F;
Fet KPM[F14]:=F;
Fet KPM[F15]:=F;
Fet KPM[F16]:=F4 F5 F6 F7 F17 F;
Fet KPM[F17]:=F;
Fet KPM[F18]:=F4 F5 F6 F7 F17 F;
Fet KPM[F19]:=F4 F5 F6 F7 F17 F;
Fet KPM[F20]:=F4 F5 F6 F7 F17 F;
Fet KPM[F21]:=I1 F1 F2 F14 F22 F;
Fet KPM[F22]:=F;
Fet KPM[F23]:=F4 F5 F6 F7 F17 F;
Fet KPM[F]:=;
```

```
Fet KFM[F]:=I1 I2 I3 I4 F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14 F15 F16
F17 F18 F19 F20 F21 F22 F23 ;
set KSM[I1]:=I2 I4 F3 F21  ;
set KSM[I2]:=;
set KSM[I3]:=;
set KSM[I4]:=;
set KSM[F1]:=I2 I4 F3 F21 ;
set KSM[F2]:=I2 I4 F3 F21 ;
set KSM[F3]:=;# I3 F5 F7 F8 F9 F10 F11;
set KSM[F4]:=F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23 ;
set KSM[F5]:=F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23 ;
set KSM[F6]:=F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23 ;
set KSM[F7]:=F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23 ;
set KSM[F8]:=;
set KSM[F9]:=;
set KSM[F10]:=;
set KSM[F11]:=;
set KSM[F12]:=;
set KSM[F13]:=;
set KSM[F14]:=I2 I3 I4 F3 F21 ;
set KSM[F15]:=;
set KSM[F16]:=;
set KSM[F17]:=F8 F9 F10 F11 F12 F13 F16 F18 F19 F20 F23 ;
set KSM[F18]:=;
set KSM[F19]:=;
set KSM[F20]:=;
set KSM[F21]:=;
set KSM[F22]:=I2 I4 F3 F21 ;
set KSM[F23]:=;
set KSM[D]:=;
```

Parameter Values

Parts/Subassemblies	bi	ci
FA	50	35
P1	15	3
P2	12	2
P3	12	3
P4	20	6
P5	10	2
P6	10	2
P7	25	4
P8	15	2
P9	18	5
P10	15	2
P11	2	0.25
P12	12	0.25
P13	3	0.25
P14	3	0.25
P15	15	3
P16	12	2
P17	14	4
P18	20	6
P19	11	1
P20	6	0.25
P21	12	4
P22	10	2
P23	9	1
P24	13	2
P25	11	2
P26	18	3
S1	20	5
S2	25	10
S3	30	6
S4	20	4

Table D.1 – Values of b_i and c_i (laser printer)

Joint	fk
S	0
I1	4
I2	4
I3	4
I4	4
F1	5.28
F2	7.2
F3	45
F4	4
F5	4
F6	4
F7	3.2
F8	3.2
F9	2.8
F10	2.8
F11	2.8
F12	3.5
F13	4.2
F14	7.6
F15	2.4
F16	3.2
F17	1.25
F18	2.3
F19	1.2
F20	3.6
F21	5.28
F22	4.2
F23	3.2
D	0

Table D.2 – Values of f_k (laser printer)

l →	I1	I2	I3	I4	F1	F2	F3	F4	F5	F6	F7	F8	F9
k ↓													
I1	.	5.93	30.23	45.06	44.35	47.96	1.71	20.96	43.3	7.79	13.01	3.23	2.59
I2	40.53	.	13.56	47.63	3.62	35.55	41.01	48.65	23.85	15.71	37.76	18.22	39.01
I3	39.01	34.3	.	36.49	5.17	7.48	38.05	31.7	9.51	20.84	28.06	35.86	28.2
I4	46.35	45.29	27.7	.	34.07	25	8.14	2.86	40.02	33.91	36.85	29.64	8.46
F1	26.32	13.57	15.27	40.31	.	34.12	38.01	47.48	31.35	36.38	48.43	19.06	42.67
F2	20.96	49.89	44.84	40.73	45.52	.	35.6	20.67	6.44	44.97	19.93	5.69	39.1
F3	30.12	28.33	48.44	24.67	13.53	41.08	.	42.68	33.74	46.42	23.14	9.24	4.04
F4	35.29	21.3	18.91	22.32	17.17	11.34	37.28	.	44.94	30.57	26.62	29.9	29.66
F5	44.27	10.31	1.78	10.04	29.42	33.64	8.95	10.56	.	27.87	15.43	27.5	9.49
F6	41.22	3.05	45.02	21.62	7.28	2.48	11.03	34.42	41.21	.	24.45	29.66	18.76
F7	1.7	34.98	35.33	44.51	9.28	44.41	15.07	12.35	3.07	45.85	.	47.01	6.96
F8	4.99	22.7	13.97	13.99	40.7	10.85	23.26	21	46.84	5.6	9.56	.	8.07
F9	11.46	12.15	36.34	45.12	4.7	41.86	47.25	13.37	27.12	10.98	38.08	30.13	.
F10	27.55	11.93	36.21	34.26	38.58	9.39	36.51	47.37	30.04	18.15	30.31	48.31	1.4
F11	25.1	24.47	3.26	33.89	29.25	37.38	22.21	39.98	45.43	48.6	5.66	36.89	21.32
F12	10.5	32.6	35.41	41.93	21.11	30.02	17.54	35.06	47.19	22.41	8.5	9.74	38.59
F13	17.91	4.68	7.67	36.66	27.87	2.97	48.98	30.15	46.24	24.97	20.88	26.07	2.57
F14	40.59	20.03	32.1	39.58	30.16	3.53	26.03	37.37	22.34	3.43	3.7	11.2	30.52
F15	28.26	24.06	20.84	46.53	40.15	5.48	16.57	20.47	9.57	21.64	28.42	9.28	40.21
F16	36.34	20.94	35.36	2.33	7.54	46.64	29.52	40.9	6.42	6.02	20.7	21.86	42.41
F17	46.59	28.69	44.94	42.87	32.55	11.96	17.15	45.58	34.09	38.19	9.47	1.87	24.48
F18	7.7	44.96	17.28	44.82	3.71	10.04	49.94	17.4	17.6	37.68	19.51	3.01	26.47
F19	47.73	41.8	29.3	43.9	42.42	23.42	11.7	8.77	7.08	1.29	6.66	37.65	39.66
F20	22.09	33.19	15.43	39.94	23.59	12.97	10.78	48.13	33.19	26.82	42.72	9.52	26.54
F21	28.84	7.56	25.09	46.09	31.77	27.03	11.75	2.34	15.08	3.43	42.94	25.32	44.12
F22	6.15	33.14	34.27	7.89	49.72	36.44	8.39	4.54	3.42	17.1	3.68	5.99	2.8
F23	32.11	13.03	21.51	38.22	2.57	10.74	32.12	18.27	6.67	30.34	38.95	3.14	17.25

Table D.3 – Values of s_{kl} (laser printer)

l →	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
k ↓														
I1	9.04	11.8	1.84	15	17.8	28.1	18.5	19.2	18.4	45.6	23.8	21.9	15.9	48.8
I2	4.64	10.7	4.14	18.6	24.9	26.1	19.3	49.3	2.99	12.3	1.24	46.4	5.92	13.6
I3	9.88	48.5	34.7	26.9	40	40.5	13.9	9.72	43.5	6.63	3.92	38.3	37.2	49.3
I4	44.7	19.5	10.8	11.1	17.4	16.9	15.7	40.3	35.1	14.3	45.3	2.92	35.7	23.2
F1	28.3	43.8	22.6	11.7	43.1	14.7	35.5	35.7	19.4	17.2	5.21	48.9	15	27.2
F2	39.4	33.6	33.2	13.7	38.5	35.3	43.1	1.14	34.3	46.5	3.08	26.4	45.7	47.8
F3	1.25	27.5	31.3	25.2	29.4	30.5	46.6	27.2	7.47	5.03	29.2	41.6	4.22	14.3
F4	25.4	6.41	30.1	28.4	38.9	12.3	36.8	29.8	27.7	40.6	48.3	5.67	6.32	35.9
F5	10.1	42.8	47.5	13.3	22.2	27.7	48.4	36.5	47.6	28.9	47.1	13.7	9.18	44.3
F6	41.3	16.2	49.5	38.8	36.7	9	40.6	46.4	12.4	19.7	5.42	45.7	42.8	29.1
F7	17.7	5.68	47.3	26.1	31.8	41.9	48.7	24.3	8.43	14.9	40.3	40.6	35.1	4.36
F8	4.71	1.75	36.3	19	33.3	1.99	44.1	2.26	15.8	9.06	23.5	28.1	48	16
F9	8.43	19.7	38.5	25.3	42.3	8.61	39	44.8	6.93	33.1	2.83	27.1	42.3	42.6
F10	.	23.1	42.1	44.7	47.5	2.69	39.7	32.5	38.8	34.6	22.9	47.6	34.1	24.9
F11	12.2	.	49.5	45.7	29	16.6	20.9	7.67	27	20.5	37.2	5.96	35.2	41.9
F12	47.3	13.9	.	42.5	29.8	34	49.8	14.8	11.6	44.5	19.6	4.84	43.6	5.84
F13	13	45.9	28.8	.	14.4	7.17	44.8	49.7	41.2	23.6	10.6	23.9	20.6	18
F14	14.2	49	21.3	40.5	.	16.8	2.59	10	27.4	30.7	40.9	1.93	13.1	28.5
F15	22.1	38.2	12	31.5	6.09	.	40	7.39	46.7	20.2	4.8	17.3	30.8	12.7
F16	42.4	19.9	34.9	41.6	11.8	12.3	.	35	3.09	44.4	48.7	25.8	5.83	25.9
F17	42.2	41	16.2	13.3	31	8	35.5	.	46.4	47.4	13.5	45.1	38.2	41.3
F18	41	40.9	20.6	34	48.3	34	27.1	15.4	.	25.1	47.3	49.9	2.64	28.2
F19	44.4	17.2	6.09	2.5	16.3	18.4	41.9	13.8	42.6	.	40.9	43.8	36.9	30.9
F20	21.9	35.1	48.3	17.6	46.7	36.6	47.3	48.9	41.1	48.9	.	25.8	46.9	13.3
F21	43.6	40.9	36.7	39.4	31.1	24	19	13.5	23.2	48.5	25.5	.	12.5	48
F22	12.8	19.7	43	38.7	22.6	22.5	9.98	45.4	42.4	15.1	47.8	9.19	.	46.4
F23	11.9	33.2	21.4	16	28.9	9.19	9.56	17.9	48.7	2.63	22.8	32.6	30.7	.

Table D.3 (Contd.) – Values of s_{kl} (laser printer)

VITA

Ajay Bhootra was born in Indore, India and completed his schooling from St. Mary's Convent School, Dewas. He earned the Bachelor of Engineering in Industrial and Production Engineering from SGSITS, Indore and Master of Science in Industrial and Systems Engineering (Operations Research) from Virginia Tech. He is currently enrolled in the Master of Science program in Finance at Virgnia Tech.