

THE SINGLE-ITEM LOT SIZING PROBLEM IN SMT PRODUCTION LINES

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

BY

HARUN SERİN

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

FEBRUARY 2014

Approval of the thesis:

**THE SINGLE-ITEM LOT SIZING PROBLEM IN SMT PRODUCTION
LINES**

submitted by **HARUN SERİN** in particular fulfillment of the requirements for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

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

Prof. Dr. Murat Köksalan
Head of Department, **Industrial Engineering**

Prof. Dr. Haldun Süral
Supervisor, **Industrial Engineering Department, METU**

Assoc. Prof. Dr. İsmail Serdar Bakal
Co-supervisor, **Industrial Engineering Department, METU**

Examining Committee Members:

Prof. Dr. Ömer Kırca
Industrial Engineering Department, METU

Prof. Dr. Haldun Süral
Industrial Engineering Department, METU

Assoc. Prof. Dr. İsmail Serdar Bakal
Industrial Engineering Department, METU

Assoc. Prof. Dr. Zeynep Pelin Bayındır
Industrial Engineering Department, METU

Dr. Bora Kat
Expert, TÜBİTAK

Date: _____

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

Name, Last Name: Harun SERİN

Signature :

ABSTRACT

THE SINGLE-ITEM LOT SIZING PROBLEM IN SMT PRODUCTION LINES

Serin, Harun
M.Sc, Department of Industrial Engineering
Supervisor : Prof. Dr. Haldun Süral
Co-supervisor: Assoc. Prof. Dr. İsmail Serdar Bakal

February 2014, 121 pages

In this study, we examine the single-item lot sizing problem in surface mount technology (SMT) production lines which consist of a single automated pick-and-place surface mount equipment (SME). In our problem setting, processing and set up times depend on the component assignment to the component pockets on an SME. Blank body loading and component pack capacities are also taken into account. We propose an integrated mathematical model to the single-item lot sizing problem in SMT production lines where lot sizing and component assignment decisions are carried out together. In addition, we examine special cases for the component assignment problem and discuss how these cases influence the solution of the problem.

Keywords: Single-item Lot Sizing, Component Assignment, SMT Production Lines, SMT Production Planning

ÖZ

YMT ÜRETİM HATLARINDA TEK ÜRÜNLÜ KAFİLE BÜYÜKLÜĞÜ BELİRLEME PROBLEMİ

Serin, Harun
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Haldun Süral
Ortak Tez Yöneticisi: Doç. Dr. İsmail Serdar Bakal

Şubat 2014, 121 sayfa

Bu çalışmada, bir adet otomatik al-ve-yerleştir yüzeye montaj cihazından (YMC) oluşan yüzey montaj teknolojisi (YMT) üretim hatlarında, tek ürünlü kafiye büyüklüğü belirleme problemi incelenecektir. Problem kurgumuzda, işlem ve kurulum süreleri bileşenlerin YMC üzerinde bulunan bileşen hücrelerine atmasına bağlıdır. Boş gövde yükleme ve bileşen paket kapasiteleri de göz önüne alınmaktadır. YMT üretim hatlarında tek ürünlü kafiye büyüklüğü belirleme problemi için, kafiye büyüklüğü belirleme ve bileşen atama probleminin birlikte ele alındığı bütünleşik bir matematiksel model öneriyoruz. Ayrıca, bileşen atama probleminin özel durumlarını da inceleyip, bu özel durumların problemin çözümlerini nasıl etkilediğini tartışıyoruz.

Anahtar Kelimeler: Kafiye Büyüklüğü Belirleme, Bileşen Hücrelerine Atama, YMT Üretim Hatları, YMT Üretim Planlama

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to my supervisor Assoc. Prof. Dr. Haldun Süral and co-supervisor Asst. Prof. Dr. İ. Serdar Bakal for their guidance, advice, criticism, encouragements and insight throughout the research.

I would like to thank my family, especially to Zeynep Dedeoğlu, and my friends for their endless supports.

I also owe thanks to my colleagues and my managers for their supports and patients.

TABLE OF CONTENTS

ABSTRACT.....	v
ÖZ.....	vi
ACKNOWLEDGEMENTS.....	vii
TABLE OF CONTENTS.....	viii
LIST OF TABLES.....	x
LIST OF FIGURES	
CHAPTERS	
1. INTRODUCTION.....	1
2. PROBLEM ENVIRONMENT AND LITERATURE REVIEW.....	7
2.1 Surface Mount Technology (SMT).....	7
2.2 Surface Mount Equipment.....	10
2.2.1 Set Up Operations on SME.....	11
2.2.2 Assembly Operations on SME.....	13
2.2.3 Production Planning Problems for SME.....	15
2.3 Related Literature for Single SME Production Planning Problems.....	16
2.3.1 Motion Control.....	18
2.3.2 Component Pick-and-Place Sequence.....	19
2.3.3 Component Pocket Assignment.....	19
2.3.4 Component Retrieval Plan.....	21
2.3.5 Nozzle Scheduling.....	21
2.4 Related Literature for the SLP.....	22
2.4.1 The Uncapacitated Single-item Lot Sizing Problem (USLP).....	23

2.4.2 The Capacitated Single-item Lot Sizing Problem (CSLP).....	24
2.5 The SLP in SMT Production Lines.....	24
3. THE SINGLE-PERIOD COMPONENT ASSIGNMENT PROBLEM..	27
3.1 Dynamic Component Assignment (DCA) for a Lot Size.....	32
3.2 Stationary Component Assignment (SCA) for a Lot Size.....	48
3.3 Numerical Experiments.....	49
3.3.1 Test Data and Experimental Settings.....	49
3.3.2 Results.....	50
3.4 Case Study.....	57
4. THE SINGLE-ITEM LOT SIZING PROBLEM.....	63
4.1 Uncapacitated Lot Sizing with DCA.....	64
4.2 Capacitated Lot Sizing with DCA.....	69
4.3 Lot Sizing with SCA.....	69
4.3.1 SCA for $Q=1$	70
4.3.2 SCA with Incremental Q	73
4.3.3 SCA with Full Utilization of Pockets.....	74
4.4 Numerical Experiments.....	75
4.4.1 Test Data and Experimental Settings.....	75
4.4.2 Results.....	77
4.5 Case Study.....	87
5. CONCLUSIONS AND DIRECTIONS FOR FUTURE STUDIES.....	91
REFERENCES.....	95
APPENDICES	
A. EXPERIMENTAL RESULTS FOR DCA-SI AND SCA-SI.....	103
B. RESULTS FOR THE LOT SIZING PROBLEM.....	115

LIST OF TABLES

TABLES

Table 2.1 Relationship between SME and problem types.....	18
Table 3.1 Components for the example product.....	28
Table 3.2 Statistics for different policies.....	30
Table 3.3 Parameters for components of example.....	40
Table 3.4 DCA-SI results for $Q=45$	41
Table 3.5 Preliminary solutions (Times are in minutes).....	44
Table 3.6 Component assignment scheme.....	46
Table 3.7 Average results for experiments.....	52
Table 3.8 Comparative results of SCA-SI and DCA-SI.....	54
Table 3.9 Component data for (3x2) layout.....	55
Table 3.10 Average results for (3x2) layout.....	56
Table 3.11 Component data for (4x2) layout.....	57
Table 3.12 Average results for (4x2) SME layout.....	57
Table 3.13 Component information for the case study.....	59
Table 3.14 Results for the case study (Times are in minutes).....	61
Table 4.1 Characteristics and demand data for test instances.....	77
Table 4.2 Average results for lot sizing with DCA.....	78
Table 4.3 Average results for lot sizing with SCA.....	79
Table 4.4 Average results for special cases of SCA.....	80
Table 4.5 Comparative results.....	81
Table 4.6 Comparative results without excluding nonoptimal solutions.....	83
Table 4.7 Different policies for the case study.....	88
Table 4.8 Results for the case study.....	89

Table A.1 Results for option (6 x c_i) with DCA-SI (Times are in minutes except for solution time).....	103
Table A.2 Results for option (6 x c_i) with SCA-SI (Times are in minutes except for solution time).....	104
Table A.3 Results for option (8 x c_i) with DCA-SI (Times are in minutes except for solution time).....	105
Table A.4 Results for option (8 x c_i) with SCA-SI (Times are in minutes except for solution time)	106
Table A.5 Results for option (10 x $\left[\frac{c_i}{2}\right]$) with DCA-SI (Times are in minutes except for solution time)	107
Table A.6 Results for option (10 x $\left[\frac{c_i}{2}\right]$) with SCA-SI (Times are in minutes except for solution time)	108
Table A.7 Results for option (10 x c_i) with DCA-SI (Times are in minutes except for solution time)	109
Table A.8 Results for option (10 x c_i) with SCA-SI (Times are in minutes except for solution time)	110
Table A.9 Results for option (10 x $2c_i$) with DCA-SI (Times are in minutes except for solution time)	111
Table A.10 Results for option (10 x $2c_i$) with SCA-SI (Times are in minutes except for solution time)	112
Table A.11 Results for option (12 x c_i) with DCA-SI (Times are in minutes except for solution time)	113
Table A.12 Results for option (12 x c_i) with SCA-SI (Times are in minutes except for solution time)	114
Table B.1 Results for USL-DCA.....	115
Table B.2 Results for CSL-DCA.....	116
Table B.3 Results for USL-SCA.....	117
Table B.4 Results for CSL-SCA.....	118
Table B.5 Results for USL-SCA1.....	119
Table B.6 Results for USL-SCA2.....	120
Table B.7 Results for USL-SCA3.....	121

LIST OF FIGURES

FIGURES

Figure 1.1 Relation between lot size and component assignment.....	4
Figure 2.1 A typical flow of materials in SMT production line.....	9
Figure 2.2 Illustration of an SME.....	10
Figure 2.3 Relations between sub problems.....	17
Figure 3.1 Layout of SME for the example.....	29
Figure 3.2 Assembly periods.....	31
Figure 3.3 Order of events in an assembly period.....	32
Figure 3.4 SME layout for the example.....	39
Figure 3.4 Graph for unit set up time for different lot sizes.....	46
Figure 3.5 Unit total times.....	47
Figure 3.6 IGap% information for different lot sizes of case ($10 \times c_i$) for DCA-SI.....	53
Figure 3.7 (3x2) layout.....	55
Figure 3.8 (4x2) layout.....	56
Figure 3.9 SME layout for the case.....	58
Figure 3.10 Component assignment scheme for the case study.....	60
Figure 3.11 Comparative results for the case.....	61
Figure 4.1 Production periods and assembly periods.....	63
Figure 4.2 Average %Gap for solution methods for different settings.....	85
Figure 4.3 Average %Time for solution methods for different settings.....	86

CHAPTER 1

INTRODUCTION

This study is concerned with the single-item lot sizing problem (SLP) incorporating automated pick-and-place surface mount equipment (SME) operations in surface mount technology (SMT) production lines.

In electronic manufacturing environment, printed circuit board (PCB) assembly is one of the important processes (Wang & Nelson, 1999). A PCB is composed of a mechanical body, which connects electronic components through conductive parts embedded in or on a non-conductive base and several electronic components assembled to the body. There can be hundreds of component types used in a PCB and the total number of components used can be in thousands. Assembly operation of a PCB is composed of assembling electronic components to the body and most of the operations are automated by utilizing numerically programmable equipment. In the industry, there are machines that can assemble hundreds of components in a minute. Almost all electronic products include at least one PCB and examples can be found in any electronic product like computers, cell phones, remote controllers, toys, electronic warfare systems, etc.

In today's electronic market, products are becoming more complicated with increasing capabilities while they are getting smaller in size. Product life cycles are getting shorter as products become old fashioned in very short time with the rapid advancements in technology. Customers demand products with the most advanced technology in small sizes, low prices, short lead times, and high quality. Therefore, electronic manufacturing companies have to react very fast to changes in the market, decrease their costs, and utilize advanced assembly technologies to sustain their competitiveness. In the last decade, SMT is widely used in PCB production. SMT enables companies to design and produce products with high component density, which is not efficient or impossible for manual work and old assembly technologies. Assembly operations are efficiently automated in an SMT

production line by the use of surface mount equipments (SME), which is often the bottleneck in an SMT production line regardless of the arrangements of SMEs in the line (parallel or sequential) (Wang & Nelson, 1999). Therefore, effective planning of SME operations in SMT production lines is vital in electronic manufacturing environments.

Our problem originates from a real life application in the so-called ‘carrier’ production shop of an electronics R&D and production firm. Carrier is a type of SMT product, which is used in high-tech microelectronic technology. It is produced in accredited clean rooms, which have special environment with strict contamination, temperature, and pressure specifications. Carriers are composed of a body and several small (in nano metric scale) components mounted to surface of the body.

In a carrier job shop, main operation is to assemble components onto the carrier body through an automated SME. Upstream operations on the line are basically preparation of carrier bodies for assembly operation. Downstream operations are finishing operations like visual inspection and manual fixing under microscope, wire bonding, and curing. A time study that we performed for carrier production line reveals that the assembly operations on SME cover about 70% of the total processing time on the line, which shows that SME operation is the bottleneck on the line.

An SME is numerically programmable equipment that picks up components from component pockets and places them on to the blank body of a product. Pick-and-place time of a component depends on the speed of the moving grabbing unit (nozzle unit) of SME and the distance travelled. As speed is predetermined by the design and the total distance travelled by the nozzle unit depends on the pocket that a component is picked up, an efficient assignment of components to component pockets is vital for minimizing the assembly time of a product. Therefore, we face a component assignment problem for the assembly operations of an SME.

Before beginning to assemble a product on an SME, all components have to be installed to assigned pockets and blank body has to be loaded. Therefore,

there is a set up time before beginning to an assembly operation. Components are installed to pockets in packs that have component specific capacities, and a limited number of blank bodies can be loaded at a time. A pocket can hold at most a pack of a component at the same time. Component installation and body loading operations take time and they cannot be operated without stopping the assembly operation. Once an assembly operation is started, it can continue until the pack capacity of a component is reached and/or all blank bodies are assembled. To continue the assembly operation, some components have to be reinstalled and/or new blank bodies have to be loaded. In addition, stopping and restarting the assembly operation can also take time. Therefore, the total time required to assemble a given lot of a product is composed of the total component installation time, the total body loading time, the total stopping and restarting time, and the total assembly time.

A component assignment decision affects the total need of stopping the assembly operation while assembling a given lot. For instance, if a component is assigned to multiple pockets where a pack of the component can be installed to each pocket, the total stopping need of the assembly operation for that component is smaller than the case where the component is assigned to only one pocket while assembling a given lot. Therefore, to utilize an SME efficiently, one has to plan component installation and body loading operations together with the component assignment decisions for assembling a given lot of a product. In addition to component installation and body loading operations, there are also product specific set up operations like adjustments of different parts of SME and installation of product specific jigs before starting the assembly operation. However, these set up operations do not depend on the lot size and once performed, they are not needed until finishing the assembly of given lot.

As we discussed above, utilization of SME while assembling a product includes unavoidable set up operations. Once SME is set to assemble a product, it is beneficial to assemble the product in a production period as much as possible to decrease the total set up time. On the other hand, if a product is assembled more than demanded in a production period, surplus amount has to be kept in the

inventory. This will increase the inventory level and related cost. Therefore, we face a lot sizing problem where the total set up cost (time) and the total inventory holding cost have to be balanced throughout the planning horizon.

The lot sizing and the component assignment problems are interconnected and Figure 1.1 illustrates this interconnection. Component assignment decision together with the component installation and body loading plan depends on the lot size. On the other hand, finding efficient lot sizes for each production period in terms of the total cost depends on the total set up and assembly times.

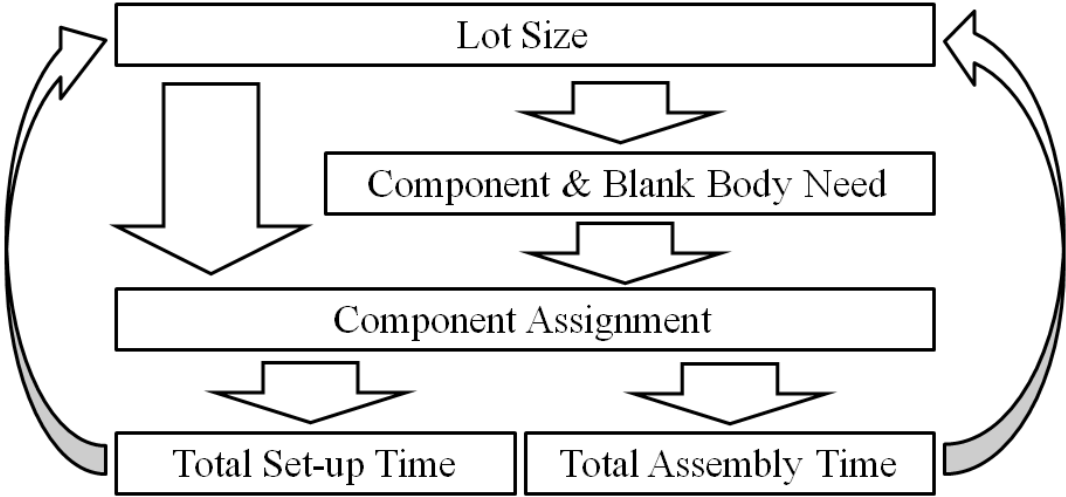


Figure 1.1 Relation between lot size and component assignment

Vast majority of the literature on the production planning problems in SMT production lines focuses mainly on the operations on surface mount equipment (SME). The subjects of these studies change from line balancing in multiple lines to planning of movements of a single SME. To the best of our knowledge, lot sizing decisions are considered as exogenous decisions. In addition, although the total number of component pack installations and body loading depends on lot sizes, component pack capacities and body loading capacities are not taken into account in the solution of the component assignment problems within an SME.

Studies make an effort with the utilization of SME for single unit of an item in much detail. However, none of the studies proposes a component assignment procedure where component installation and body loading operations are also planned throughout the assembly operation of a given lot.

There are quite a few studies in the literature about the SLP in both capacitated and uncapacitated cases. Several extensions of the standard formulations are introduced and efficient solution algorithms are developed. However, in all problem settings, production and set up costs (times) can be estimated as fixed or as a function of time. They do not include any problem setting where set up and assembly costs (times) depend on the lot size. Therefore, none of them can be applicable to our problem setting where the component assignment problem is also included in SMT lines.

The aim of the study is to examine the SLP in SMT production lines where component assignment and lot sizing decisions are integrated. After reviewing related literature, we have seen that there is a need for studying the lot sizing problem in SMT production lines, and examining connections and tradeoffs between the lot sizing and the component assignment decisions. The potential improvement that we expect to achieve from an integrated approach motivates us to conduct this study.

We first formulate the single-period component assignment problem, which refers to the component assignment problem for a given lot of a product. We propose a mathematical model for the solution of the (single-period) component assignment problem, which provides a component installation and body loading plan together with the component assignment solution while assembling a given lot. We also consider the dynamic and stationary cases of the component assignment problem; we define the dynamic component assignment case where a component can be assigned to different pockets throughout the assembly operations of a given lot, whereas in the stationary component assignment case assignments are fixed.

After formulating the single-period component assignment problem and proposing a mathematical model, we integrate our component assignment

formulation into the SLP. Among several formulations of the lot sizing problem, we pick the one based on the facility location or the transportation problem type of the formulation of the lot sizing problem because of simplicity in understanding. We propose an integrated mathematical model for the SLP in SMT production lines, and we consider both the uncapacitated and capacitated cases. We also propose heuristic approaches for the component assignment part of the lot sizing problem.

Test results show that our mathematical models can find optimal solutions for small sized problems when compared to the size of real life applications. Test results also show that our heuristic approaches not only give near optimal solutions, but also can solve real life problems in reasonable times. We also present case studies for a real life application and show that our integrated approach always gives better results in terms of the total cost.

The rest of the study is organized as follows: In Chapter 2, we first introduce SMT and problem environment. After that, our literature review is presented in two parts. The first part covers the related studies and results in the literature about single SME optimization problems. In the second part, we present the basic results about the SLP. In Chapter 3, the component assignment problem is considered for a given lot size. We present our mathematical models and the results of our numerical experiments in this chapter for the component assignment problem. In Chapter 4, we examine the single-item lot sizing problem in SMT production lines. We present our mathematical models for capacitated and uncapacitated cases. We also introduce three special cases for the component assignment problem and present the results of our numerical experiments. In Chapter 5, we conclude our findings and discuss future research opportunities.

CHAPTER 2

PROBLEM ENVIRONMENT AND LITERATURE REVIEW

PCB assembly is a fundamental part for electronic manufacturing environment. PCB products are composed of several electronic components bonded to a blank circuit board. By the help of new technologies, PCB products become smaller in size with increasing capabilities and SMT is utilized in almost all PCB manufacturing. An efficient planning procedure is required since time, flexibility, quality, and cost are crucial in electronic manufacturing environment. In this context, we consider a single SMT production line with single machine.

In an SMT production line, SME is the main bottleneck resource, which requires very high investment. Therefore, a better utilization of SME is aimed in all production planning problems in SMT production lines. We consider the utilization of SME in SMT production lines and assume that all other operations can be balanced according to this utilization. Details of such production systems are explained in following subsections.

2.1 Surface Mount Technology (SMT)

Surface mount technology is a method for producing PCB products in which the components are mounted directly onto the surface of a blank PCB. In today's electronic industry, this technology is widely used instead of the through hole technology assembly method, which is an old method where components are fitted into holes in the blank circuit boards with wire leads. Although both technologies can be used together in one product, it is impossible to mount some components like connectors or other larger components by using SMT. Examples of this kind of products can be found in several electronic devices like computers, mobile phones, remote controllers, electronic warfare systems, etc.

By using SMT, cheaper and smaller products with high capabilities can be designed and produced. This is the result of advantages of SMT:

- Products can be designed with much higher component density (in terms of the amount of component per unit area) as components used in SMT are smaller.
- Components can be mounted on the both sides of the blank circuit board.
- Assembly operation is simpler and takes less time with faster and reliable automation.

Although details can be different for different products or different manufacturing environments, Figure 2.1 illustrates a typical SMT production line used in an electronic PCB production environment. Flow of materials can be fully automated through the line or partly automated by using conveyors and other material handling equipments, or can be fully manual. Below, we briefly explain each step in an SMT production line.

In the adhesive application step, a special adhesive (e.g. solder paste) is applied on the blank body with product specific jigs. These jigs guarantee that adhesive is only applied where components will be placed with proper thickness. This step can be operated manually or automatically. Then, surface mount components (SMC) are assembled automatically by SME. This step is the main assembly step in SMT production lines and all other steps are synchronized according to the operations in this step. Further information will be provided on SMEs in the next section. Some components that cannot be assembled automatically are assembled manually after automatic assembly operations. During assembly operations, components are not firmly soldered or bonded to the body of the product. In the permanent fixing step, assembled products are taken into an oven and a special heat treatment is applied. Because of the special property of used adhesives, this operation firmly bonds the components to the product body. After permanent fixing, components other than SMCs are assembled manually or automatically.

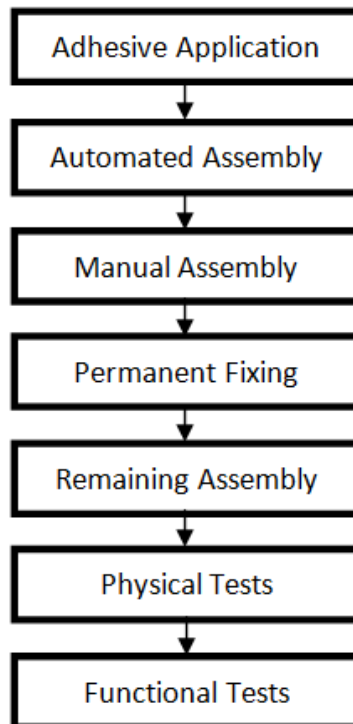


Figure 2.1 A typical flow of materials in SMT production line

During assembly operations, some components can be weakly assembled. Even if the product is electronically accurate, these weaknesses can cause problems while the product is in use and shorten the product life. To detect and eliminate these kinds of weaknesses, the assembled products are physically tested. The most common operation is to expose the assembled products to temperature stress in an oven by suddenly decreasing and increasing the environment temperature to the predefined level. This process breaks the weak bonds between components and product body, and the product will not pass the electronic tests. Another physical test operation is to apply predetermined force to break bonds between wires and product body. If wire bonds are firm enough, they must stand to the applied force. Finally, assembled products are electronically tested automatically or manually. If products include embedded firmware, it is also installed in this step. These tests are designed so that they can detect any assembly

or component defects. After this step, conforming products are ready for delivery to customers or use in upper level products.

Having covered the major steps on an SMT production line, we next focus on the surface mount equipment (SME) and corresponding critical issues.

2.2. Surface Mount Equipment

An SME is a numerically programmable equipment that picks up SMCs and places them to required places on blank bodies of products automatically. There are several types of SMEs used in the industry and readers can refer to Ayob et al. (2002) for details of different types of placement machines.

A general illustration for an SME is given in Figure 2.2. In Figure 2.2(a) a typical SME is illustrated. The place pointed by 1 is the working table, point 2 is three dimensional moving head which holds the nozzle unit and point 3 is the SMC pocket unit.

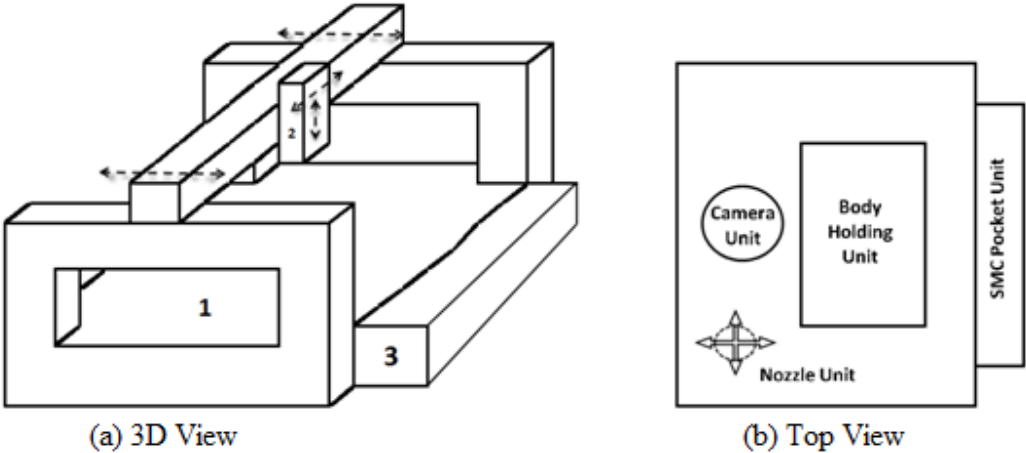


Figure 2.2 Illustration of an SME

In Figure 2.2(b), top view of the SME is illustrated. Parts are explained as follows:

- *SMC Pocket Unit:* This unit has special component pockets that can hold SMCs in packs, tapes, or rolls. These pockets can be feeders, trays or in other formats, component feeding instruments and they have feeding capability such that a nozzle unit grabs the same component from the same place each time. SMCs are picked from the assigned pockets of the SMC pocket unit by the nozzle unit. SMC pocket unit can be placed at different locations for different SME types and even in some SME types there can be multiple or moving SMC pocket unit.
- *Body Holding Unit:* This unit holds the blank bodies to be assembled. Loading operation of a blank body can be manual or automated. There can be body holding jigs specific to the product type, which is fixed to body holding unit. There are also SMEs with movable body holding unit.
- *Nozzle Unit:* This unit is the main tool for picking and placing operations. Component specific nozzles are fixed to this unit. Nozzles can grab components with the help of vacuum, up and down moves for picking and placing components, and also make rotational movements. A nozzle unit can hold single or multiple types of nozzles at the same time depending on the SME type.
- *Camera Unit:* Some components have to be inspected and adjusted before placing on to the blank body. This corrective operation is called as look-up operation and carried out at the camera unit.

2.2.1 Set Up Operations on SME

At the beginning of assembly operations, the SME has to be prepared for the corresponding product. When a production order arrives, the operator checks the set up documentation specific to the product type. This documentation includes the information about which components must be installed to which component pocket on the SMC pocket unit and required nozzles for the components. The operator also selects the product specific software from the software library in the SME controlling computer. This software programs all pick-and-place movements

of the SME and contains required information about the components, nozzles, and corresponding points on the blank body where components will be placed. The operator installs all required components to pockets according to the documentation and introduces the exact place of each component to the SME by moving the nozzle unit to reference points on the component pack, tape, or rolls. If blank body loading is not automated, a body holder jig is installed to the body holding unit, which includes prefixed blank bodies on it. The operator also introduces blank bodies by moving the nozzle unit to reference points on the body holder jig. If blank body loading is automated, this operation is not required in general. In real life applications, generally an inspection and adjustment to set up is performed after assembling the first item. After starting the assembly operation, it continues until body holder jig capacity or one of the component pack capacity is reached, or production order is completed. As a result, SME requires four types of set up operations before and during the assembly operations:

1. *Major set up:* It occurs when beginning to assemble a new type of product and includes the following set ups.
 - a. *Component installation:* Required components are installed at predetermined pockets of SMC pocket unit and they are introduced to SME by indicating reference points on component packs, tapes, or rolls to the camera of nozzle unit. By doing this, SME can store exact grabbing locations of each component. SME cannot perform assembly operations when there is a component installation operation. The total time of component installation depends on the total number of component packs, tapes, or rolls to be installed at different pockets.
 - b. *Nozzle installation:* While grabbing different components, different nozzle types can be needed depending on the dimensions and surface properties of the components. If nozzle unit has enough nozzle capacity to assemble a product type, all required nozzles are installed before beginning to assembly operation.

Otherwise, required nozzles have to be installed during the assembly operations. SME cannot perform assembly operations if there is a nozzle installation operation.

2. *Blank body loading:* If blank body loading is not automated, a body holder jig specific to product type is installed to the body holding unit which includes the prefixed blank bodies on it. The operator also introduces blank bodies by moving nozzle unit to reference points on the body holder jig. Body holder jigs have limited capacities. If the lot size of assembled product is greater than body holder jig capacity, a new body holder jig has to be installed while assembling the same product type. If a blank body loading is automated, generally, this operation is not required. SME cannot perform assembly operation when a blank body loading operation is carried out.
3. *Component reinstallation:* While assembling a product, the number of components initially installed may not be sufficient to assemble the entire lot. In such a case, assembly operations stop when all pieces of a component are used and it needs to be reinstalled to continue with assembly operations. SME cannot perform assembly operations if there is a component reinstallation operation.
4. *Nozzle change:* If the nozzle unit does not have enough capacity to hold all required nozzles for a product type, a nozzle change operation is required while continuing the assembly. These changes can be manual or automated depending on the SME type. SME cannot perform assembly operations if there is a nozzle change operation.

2.2.2 Assembly Operations on SME

Assembly operations consist of consecutive pick-and-place cycles. However, nozzle unit movements depend on look-up characteristics of components. Therefore, we have to differentiate assembly operations according to whether there is a look-up option or not:

1. *Assembly without a look-up option:* After placing a component onto the blank body, nozzle unit travels back to a pocket to pick the next component by covering the distance from the body holding unit to the corresponding pocket. Nozzle picks up the component from the pocket, and travels back to the body holding unit covering the same distance and places the component onto the body. Therefore, one cycle consists of the required time for picking and placing a component plus the required time to travel the distance in the corresponding pick-and-place cycle.
2. *Assembly with a look-up option:* Some SMCs require adjustments before assembling them onto the blank bodies and these components are predefined at the product design phase. This operation is called look-up option. Shape, dimensions, and properties of a component determine whether it has a look-up option or not. After placing a component onto the blank body, the nozzle unit travels back to a pocket to pick up the next component by covering the distance from the body holding unit to the corresponding pocket. Nozzle picks up the component from the pocket and travels to the camera unit. Adjustment of component is done at the camera unit and the nozzle travels back to the body holding unit and places the component onto the body. Therefore, the total time of one cycle consists of picking time of components, placing time of components, the time spent at the camera unit, and the time to travel the distance in the corresponding pick-and-place cycle.

Total assembly time for a product depends on the total number of SMC with and without look-up options, assignment of SMC types to pockets on the SMC pocket unit, and speed of the nozzle unit. Note that the nozzle unit speed depends on the capability of SME and it can be programmed to move in various speeds for different components.

2.2.3 Production Planning Problems for SME

Companies can produce PCB products for customers or for internal uses in higher level products. In both cases, demands for PCB products are known in advance for a planning horizon. Considering working hour limits and unavoidable set up times for SME in a production period, planners have to determine the quantity of a product to be assembled in each production period over the planning horizon to satisfy all demands on time. If an SME is set to assemble a type of PCB, it will be beneficial to produce as much as possible to minimize the total set up time to be spent throughout the planning horizon. On the other hand, when a PCB type is produced in a period more than the demand in that period, surplus amounts have to be kept in inventory until they are needed, and this will increase their inventory levels and related costs. It follows that the total set up times and the total inventory level and associated costs have to be balanced throughout the planning horizon while deciding about lot sizes for every production period.

To minimize the total assembly time of components for a product, it is critical to assign components to pockets so that the total distance travelled by the nozzle unit will be minimized. This observation suggests that the total assembly time of a product can be shortened by assigning most used components closer to the body holding unit. However, trying to shorten the assembly time without considering the lot size may cause inefficiencies in set up times.

As explained above, there are two decision problems in planning the utilization of an SME for a product: (1) how much to assemble in each production period throughout the planning horizon and (2) how to achieve an efficient component assignment to pockets. Although these problems seem to be hierarchical, they are interconnected. Assembly and set up times depend on the component assignment scheme as they are key parameters while deciding on the lot sizes. On the other hand, different lot sizes can influence the component assignment decisions. Therefore, these two decision problems have to be solved together.

As we mention in following subsections, to the best of our knowledge, the lot sizing problems are not structured yet for the SMT production lines in a way that the lot sizing and component assignment decisions are integrated.

2.3 Related Literature for Single SME Production Planning Problems

Crama et al. (2002) provide a hierarchical list of production planning problems in PCB production. They assume that demand mix and job shop layout are fixed, and lot sizing problem is given as a long term decision exogenously. As discussed in their study, production planning problems can be categorized in eight sub-problems as follows:

1. Assigning product types to product families and to machine groups.
2. Allocating component feeders to different machines.
3. Allocation of components to different machines indicating which components will be placed by each machine for each product type.
4. Sequence of products in a line.
5. Component pocket assignment.
6. For each machine and each product type, component placement sequence.
7. For each machine and each product type, a component retrieval plan indicating from which pocket a component will be picked up.
8. For each machine and each product type, a motion control specification indicating where movable parts of SME should be located while picking or placing a component.

Crama et al. (2002) claim that sub-problem (1) is concerned with the whole job shop and product types, sub-problem (2) to (4) include decisions within a production line, and remaining sub-problems are related with single SME. They review the related literature about sub-problems, and discuss relations between

mathematical models and environmental variables like layout of the job shop, product mix and specifications of different placement machines.

As our scope is single SME with single item, we focus our review accordingly. For the multi-item case, readers can refer to Crama et al. (2002).

Ayob and Kendall (2009) survey related literature about single SME production planning problems. They classify problems in five categories where four of them are sub-problems (5) to (8) in Crama et al. (2002), and the fifth one is the nozzle scheduling problem. They claim that these distinct problems intervene with each other as illustrated in Figure 2.3.

Past studies try either to solve a combination of these problems or focus on an individual problem by assuming others are predefined. These formulations depend on the SME type. For instance, Ayob and Kendall (2009) classify the production planning problems for single SME according to Table 2.1.

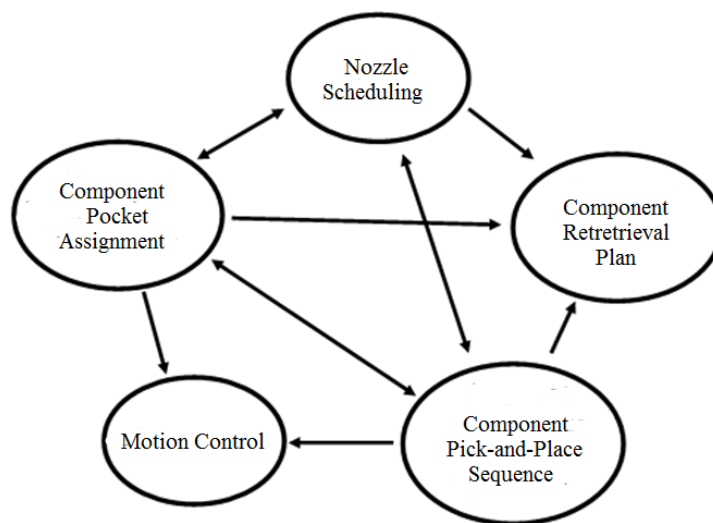


Figure 2.3 Relations between sub problems

In Table 2.1, the SME characteristic in the first row is not classified due to technology change. Ayob and Kendall (2009) state that it might be an old machine and none of the work focuses on this SME type except Leu et al. (1993).

Table 2.1 Relationship between SME and problem types

	Problem Type	SME Characteristics	SME Type
1	Travelling salesman problem (TSP)	Stationary nozzle unit, movable body holding unit, direct feeding of the components to nozzle unit.	Unclassified
2	Pick-and-place problem	Moving nozzle unit, stationary body holding unit, stationary SMC pocket unit	Sequential pick-and-place
3	Moving body with time delay problem	Moving body holding unit, moving SMC pocket unit, supply of components with a multi-head turret or a moving nozzle unit between two fixed location	Turret-type

2.3.1 Motion Control

Motion control problem occurs in turret-type SME (third row in Table 2.1) where all nozzle, body holding, and SMC pocket units are movable. It is the problem of finding efficient pick-and-place points for components by planning the robotic moves of the SME. Su et al. (1995) propose a dynamic pick-and-place points approach and compare it with the fixed pick-and-place point approach, and shows that dynamic pick-and-place approach is always superior to fixed pick-and-place approach on an SME.

Bonert et al. (2000) formulate the motion control problem as a travelling salesperson problem (TSP) where both the salesperson (e.g. nozzle unit) and the cities (e.g. body holding unit) are moving. They utilize genetic algorithm (GA) as a search tool for finding solutions. Ayob and Kendall (2005) propose a mathematical model for motion control problem with a weighted multi-objective function that minimizes robot assembly time, movements of body holding unit and SMC pocket unit at the same time. They claim that results give good robot assembly times and less movement of SMC pocket unit and body holding unit.

2.3.2 Component Pick-and-Place Sequence

Pick-and-place sequence refers to finding a shortest route (or time) to pick-and-place components to blank body while assembling a product. Chan and Mercier (1989) claim that by defining PCB points as cities and the time between component placement as distances, component pick-and-place sequencing problem can be formulated as a TSP. Bard et al. (1994) propose a weighted nearest neighbor TSP to generate a component pick-and-place sequence. Francis et al. (1994) employ a special structured TSP formulation for solving pick-and-place sequencing problem. Khoo and Ong (1998) provide a GA for component pick-and-place sequencing problem. Jeevan et al. (2002) also formulate the problem as TSP and use GA for solving it. They also include the nozzle changing decision to their problem. Kumar and Luo (2003) formulate the problem for FUJI FCP-IV and obtain near optimal solutions by relaxing the problem to an instance of TSP. Duman and Or (2004) also include precedence relations of components into the TSP formulation.

Ball and Magazine (1988) formulate the problem as a rural postman problem and propose an algorithm which yields the optimal solution under certain conditions. Altinkemer et al. (2000) formulate the sequencing and component assignment problems as a vehicle routing problem and propose an integrated approach. Grunow et al. (2004) also formulate the problem as a vehicle routing problem, but they don't integrate the problem of component assignment.

2.3.3 Component Pocket Assignment

Component pocket assignment problem refers to decision of assigning different components to different pockets. Ball and Magazine (1988) state that cost of a particular assignment depends on the component pick-and-place sequence. Sun et al. (2005) also claim that evaluation of the quality of a solution is not straightforward in assignment problem and other problems should be solved while searching for an improved component assignment.

Leipälä and Nevalainen (1989) formulate the component assignment problem as a quadratic assignment problem after finding a pick-and-place sequence in a turret-type SME setting. Grotzinger (1992) studies the problem for dual delivery cases where SMC pocket unit is also movable, and proposes an MIP model. He claims that this model is illustrative about the difficulties and advantages of modeling highly complex concurrently moving systems.

Sohn and Park (1996) solve the component assignment and component pick-and-place sequence problem sequentially for a turret-type SME. They first assign components to pockets according to the frequency of use, and then solve the sequencing problem by using this assignment. Moyer and Gupta (1996) solve the component assignment problem for a turret-type SME by assuming component pick-and-place sequence is predefined.

Yeo et al. (1996) propose a rule-based approach for solving the component assignment and component pick-and-place sequence problems simultaneously for a turret-type SME. They claim that, to optimize SME, next pick up operation should be at most one pocket distance away. Based on the component pick-and-place sequence, they try to maximize the number of adjacent pockets in consecutive pickups.

Klomp et al. (2000) solve the component assignment problem by using a heuristic for a turret type SME. They conduct numerical experiments using real life instances and claim that the heuristic is superior to approaches currently used in practice. Sun et al. (2005) use a genetic algorithm (GA) for assigning components to pockets for a turret type SME. They try to improve efficiency of SME by maximizing simultaneous pickups of components.

All these studies focus on the turret-type SME and feeder type of component pockets. Ayob and Kendall (2005) claim that tray type of pockets are an unexplored field and researchers ignore tray type of pocket problems by assuming there is no tray type of pockets on SME.

2.3.4 Component Retrieval Plan

This problem refers to the decision of which component will be picked up from which pocket if a component is assigned to more than one pocket. Crama et al. (1996) view the problem as a longest path minimization problem and propose a two phase polynomial time DP algorithm. Ho an Ji (2006) propose a hybrid genetic algorithm to solve the component pick-and-place sequencing, component assignment and component retrieval plan problems simultaneously for a turret-type SME.

2.3.5 Nozzle Scheduling

This problem refers to finding an efficient nozzle assignment, sequencing and switching by minimizing the total number of nozzle change during an assembly operation on an SME. Magyar et al. (1999) state that optimizing pick-and-place operations without considering nozzle changes can yield solutions where many unnecessary nozzle changes decreasing the throughput of an SME.

Chang and Terwilliger (1987) propose a rule based approach to process planning in printed wiring assembly operations and one of the rules aims minimization of nozzle change. Shih et al. (1996) also propose a rule based approach to nozzle scheduling problem.

Magyar et al. (1999) propose a hierarchical problem solving approach where component pick-and-place sequencing and scheduling the assignment of nozzles are carried out sequentially. Tirpak et al. (2000) assign nozzles to nozzle units by considering the best distribution of nozzles to nozzle units. They improve initial nozzle set up by randomly selecting two nozzles and changing their positions. Jeevan et al. (2002) include nozzle change decisions while solving component pick-and-place sequencing problem in their GA solution approach.

2.4 Related Literature for the SLP

The single-item lot sizing problem (SLP) with deterministic dynamic demands have been the subject of many studies in the fields of Operations Management and Operations Research for decades. We refer to Brahim et al. (2006) for an informative review of the standard SLP and its extensions. The problem is to determine whether a production will take place or not and what will be the quantity if it takes place in a production period. Starting a production in a production period incurs a set up cost. If a production takes place in a production period, producing as much as possible will be beneficial in terms of the total set up cost throughout the planning horizon. On the other hand, an inventory holding cost is incurred for excess quantities produced in a production period. Therefore, there is a tradeoff between the total setup cost and the total inventory holding cost.

Lot sizing problems can be classified in terms of number of machines, number of product levels, length of production periods, capacity constraints, etc. Considering number of product levels, lot sizing problems are classified as single level and multi level lot sizing problems. Single level lot sizing problems only deal with one level of product while multi level lot sizing problems include the lower level products.

According to the length of production period, lot sizing problems are classified as small time bucket and big time bucket lot sizing problems. In small time bucket problems, a planning horizon is broken into small time intervals in which at most one item can be produced. A basic small time bucket problem is the Discrete Time Lot Sizing and Scheduling Problem (DLSP). One of the main drawback of the DLSP is that capacity is either fully used or nothing is produced in a period. This restriction is removed in the Continuous Setup Lot Sizing Problem (CSLP), but still only a single-item can be produced within a time period (Karmarkar & Schrage, 1985).

In big bucket lot sizing problems, production periods are relatively long and more than one unit item can be produced. Brahim et al. (2006) state that the first two works about big bucket lot sizing problems are Wagner and Whitin

(1958) and Manne (1958). Unlike continuous models like Economic Order Quantity (EOQ), big time bucket lot sizing models assume that the planning horizon is finite and divided into discrete production periods where demands are given and may be different for different production periods. For this reason they are sometimes called Dynamic Lot Sizing Problems.

2.4.1 The Uncapacitated Single-item Lot Sizing Problem (USLP)

The standard form of the USLP minimizes the total cost throughout the planning horizon. The first exact algorithm for solving this model is an $O(T^2)$ forward dynamic programming (DP) algorithm presented by Wagner and Whitin (1958). Wagelmanst et al. (1992), Aggarwal and Park (1993), Federgruen and Tzur (1991), and Van Hoesel et al. (1994) improve that DP algorithm to $O(T \log T)$.

Evans (1985) proposes a shortest path (SP) formulation for the USLP. The formulation is based on a graph where each node represents a production period, including an additional dummy production period at the end of the planning horizon. An arc between two nodes represents the option of a production in the production period where the arc begins with a quantity that is sufficient for the total demands of production periods from the period where arc begins until the period just before where arc ends. Importance of this formulation comes from the fact that the linear relaxation of the formulation always gives integer values at optimality.

As in SP formulation, Krarup and Bilde (1977) prove that the linear relaxation of facility location based formulation (FL) of the USLP also gives integer values at optimality. They also suggest an $O(T^2)$ algorithm to solve the problem. Based on FL formulation, Wagelmanst et al. (1992), Aggarwal and Park (1993), and Federgruen and Tzur (1991) also obtain different algorithms with $O(T \log T)$ complexity.

Zangwill (1969) formulates the problem with an extension allowing backlogging where a stock out occurs and there is a penalty cost for each unit stock out. Sambasivan and Schmidt (2000) formulate the problem for multiple

facilities. They also propose a SP reformulation and a branch and bound procedure to solve the problem.

Friedman and Hoch (1978), Rajagopalan (1992), and Nahmias (1982) study the USLP for perishable goods where inventory holding cost depends on how long a product is in inventory. Hsu (2000) suggests a $O(T^4)$ DP algorithm for solving the USLP for perishable inventory.

2.4.2 The Capacitated Single-item Lot Sizing Problem (CSLP)

In real life applications, resources are capacitated. This leads the characterization of the CSLP by limiting production quantity by a given capacity. The complexity of the problem becomes an issue after including the capacity constraint. The CSLP is generally NP-hard even for special cases (Florian, Lenstra, & Rinnooy Kan, 1980), (Bitran & Yanasse, 1982). However, it is not NP-hard in the strong sense and several polynomially solvable cases are described by Bitran and Yanasse (1982).

Chen et al. (1994) propose a DP algorithm for solving the most general CSLP with more than 24 production periods. Shaw and Wagelmans (1998) propose a DP algorithm for the CSLP with a piecewise linear production cost function, general holding cost function, and backlogging. Baker et al. (1978) and Lotfi and Yoon (1994) use branch and bound algorithm to solve the CSLP. Chung et al. (1994) use a combination of DP and branch and bound methods for solving the CSLP.

2.5 The SLP in SMT Production Lines

As we discussed above, there are innumerable studies on the SME production planning problems and the SLP in the literature. In the context of the component assignment problem, we have reviewed the single-item, single-SME production planning literature. None of these studies takes the component installation and blank body loading operations into account. They only try to find

an efficient utilization of an SME for single unit of a product by minimizing the assembly time. In terms of set up operations, the only problem defined so far for single type of product is the nozzle scheduling problem. However, this problem is also concerned with the assembly of a single unit. All of these studies implicitly assume that minimizing assembly time of single unit of a product and repeating the solution for each unit while assembling a given lot will result in an efficient utilization of an SME. This assumption is reasonable in terms of the total assembly time. However, if we consider assembling a given lot of a product, studies in the literature do not provide any component installation and body loading plan which also contribute to the total time to assemble a given lot. In addition, more efficient component assignment decisions may be available when a lot size is considered instead of single unit. For instance, the decision of assigning a component to multiple pockets can only be examined when one considers a lot size instead of single unit where a pack of the component is always sufficient for single unit and may not be sufficient for the lot size.

Component installation is only considered as a major set up operation in studies dealing with multiple item cases. However, reinstallation of new packs of components is never considered in those studies as they assume that pack capacities are always sufficient for a given lot of a product. This assumption is applicable for reel type of component packs in some technologically advanced SMEs as a new reel can be attached to running out reel without stopping the assembly operation. On the other hand, this assumption is not valid for stick and tray type of component packs.

Body loading is never taken into account as a set up operation in the production planning problems for SMEs. Studies assume that blank bodies are loaded one by one and do not affect any decision, or body loading time can be included in the unit assembly time of a product. However, this assumption is not valid for cases where blank bodies are loaded in multiple units at a time.

Because of all reasons we discussed above, literature about the production planning problems for SMEs does not consider set up operations of a given lot. Our main contribution in this field is to formulate the component assignment

problem in such a way that, solution of the problem will provide a component installation and body loading plan together with the component assignment decisions while assembling a given lot.

In the SLP literature, the lot sizing problem is studied with fixed or time depended set up and unit production cost (time) figures. When we consider the assembly operation on an SME, the total set up and the unit assembly times depend on the lot size and the component assignment decisions. Without knowing the lot size of a product for each production period in advance, it is very hard to calculate or estimate the total set up and the unit assembly times for each production period. Therefore, there is a need for extending the classical lot sizing formulations in the context of assembly operations on SMEs. With this motivation, we reformulate the lot sizing problem by integrating our component assignment formulation such that the solution of the problem provides not only the lot size but also a component installation and body loading plan together with the component assignment decisions for each production period in the planning horizon. We also propose heuristic approaches to solve the problem.

CHAPTER 3

THE SINGLE-PERIOD COMPONENT ASSIGNMENT PROBLEM

In this chapter, we first make a detailed explanation of the component assignment problem for a single product with given lot size and discuss the different options while assigning components to component pockets. After that, we present our problem formulation verbally with our basic assumptions. We also propose a mathematical model for solving the component assignment problem.

The objective of the component assignment problem for a given lot size is to minimize the total time required to assemble the whole lot. This requires minimizing the total assembly time and the total set up time simultaneously, which is composed of the total component installation time, the total body loading time, and the total restarting time.

An SMT product can be composed of several types of SMCs with different quantities assembled onto the product body. While assembling a product, SME picks up components from the corresponding pockets and places them on the corresponding places on the body. Therefore, component packs have to be installed to pockets at the beginning of the assembly operation. Each pocket can hold at most one type of component at the same time and they have different distances from the body holding unit of SME. So, the nozzle unit travels different distances for assembling components from different pockets. Some of the components used in a product can also have look-up option in which components are visually inspected and adjusted at the camera unit before placing them on to the body. Therefore, the total distance travelled by the nozzle unit also depends on whether the component has look-up option or not while assembling a component from a pocket. These properties of SME show that the total assembly time of a product directly depends on the assignment decisions of components to pockets.

Before beginning the assembly operations, all required components have to be installed to the component pockets and blank bodies have to be loaded to the

body holding unit. However, component packs have component specific capacities and a limited number of bodies can be loaded to the body holding unit at the same time. Therefore, the assembly operation has to stop for loading new bodies or installing new packs of components if they are not sufficient to assemble the whole lot. Body loading, component installation and restarting SME after stopping take some time. An illustrative example is provided below.

Assume that we have a product containing only two components given in Table 3.1 and look-up is not required for any of them. Also, assume that the body loading capacity of this product is 4 units, and 8 units of it will be assembled using an SME which has a layout like in Figure 3.1. As can be seen from Figure 3.1, pocket 1 is the closest one to the body holding unit and pocket 3 is the farthest one.

Table 3.1 Components for the example product

Component ID	1	2
Number of component used for one unit	3	1
Component pack capacity	10	20

To minimize the total assembly time, it is important to assign components that are used more frequently to closer pockets. For assembling one unit of the product, it is clear to see that assigning component 1 to pocket 1 and assigning component 2 to pocket 2 will give the best assembly time. Note that, one pack of component 1 is only sufficient for 3 units of the product. Therefore, installing one pack of component 1 to pocket 1 and installing one pack of component 2 to pocket 2 at the beginning of assembly operation requires a stop after assembling 3 units of the product to install second pack of component 1. As body loading capacity is 4 units, assembly operation has to stop after assembling 4 units of the product too. Another option for the component assignment can be installing a second pack of component 1 to pocket 2 and installing one pack of component 2 to pocket 3 at the

beginning. This will eliminate stopping and restarting of the assembly operation after assembling 3 units of the product. However, this will increase the total assembly time.

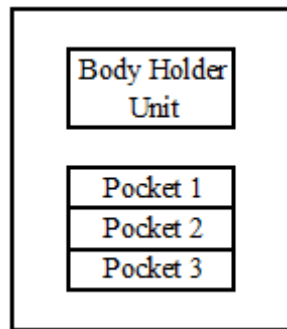


Figure 3.1 Layout of SME for the example

Consider the following three component assignment policies and related statistics given in Table 3.2 for assembling 8 units of the product:

1. Install one pack of component 1 to pocket 1 and one pack of component 2 to pocket 2. Assemble 3 units. Reinstall a second pack of component 1 to pocket 1. Assemble 1 unit. Load new bodies. Assemble 2 units. Reinstall a third pack of component 1 to pocket 1. Assemble 2 units.
2. Install one pack of component 1 to pocket 1 and one pack of component 2 to pocket 2. Assemble 3 units. Reinstall a second pack of component 1 to pocket 1 and load new bodies. Assemble 3 units. Reinstall a third pack of component 1 to pocket 1 and load new bodies. Assemble 2 units.
3. Install one pack of component 1 to pocket 1, a second pack of component 1 to pocket 3 and one pack of component 2 to pocket 2. Assemble 4 units. Reinstall a third pack of component 1 to pocket 1 and load new bodies. Assemble 4 units.

Table 3.2 Statistics for different policies

Related Statistics	Policies		
	1	2	3
Number of component pack installation	4	4	4
Number of body loading	2	3	2
Number of stopping the assembly operation	3	2	1
Number of component used from pocket 1	24	24	20
Number of component used from pocket 2	8	8	8
Number of component used from pocket 3	0	0	4

As can be seen from Table 3.2, different policies give different results in terms of the number of different set up operations or the number of component usages from different pockets. Policy 1 and 2 are equivalent in terms of assembly time as the same amount of components are used from the same pockets in both policies. In addition, policy 1 and 2 give shorter assembly times than policy 3 where 4 units of component 1 is used from pocket 3. However, policy 3 gives the best total set up time as the number of stopping assembly operations is smaller and the number of other set up operations are smaller than or equal to those in policy 1 and 2. The best policy to assemble 8 units of the product depends on the time each set up operation takes, the time of starting assembly operations after each stop, and the assembly time of components from each pocket.

The example shows that even for a small example, there are several component assignment policies to assemble a given lot of a product. To select the best alternative, an efficient component assignment procedure is needed.

Assembling a given lot without stopping the assembly operation is not possible if pack capacities of components or body loading capacity is not sufficient to assemble the whole lot. Therefore, assembling a given lot can be composed of several small assembly periods as shown in Figure 3.2. We define an assembly

period as a time period that is spent for assembly operation between two consecutive stops of an SME.

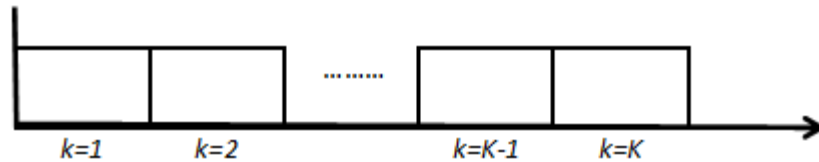


Figure 3.2 Assembly periods

In Figure 3.2, assembly periods are indexed with k . An assembly period starts with required set up operations. At the beginning of an assembly period, component installation and/or body loading operations are performed. After that, assembly operations start and continue until one of the capacity limit of component packs or blank body is reached, or the whole lot is finished. The lot is completed in K assembly periods. We illustrate the order of events within an assembly period in Figure 3.3 for our problem setting. Note that at least one of the operations shown with 1 and 2 is performed at the beginning of an assembly period. For example, only a blank body loading can be performed if there are enough components in pockets. Note that if there are some remaining components in a pocket and a component pack is to be installed to that pocket, remaining amounts are uninstalled from the pocket before the installation of new pack, and we assume that uninstallation does not take time. In addition, we assume that body loading and component installation cannot be performed simultaneously.

As can be seen from Figure 3.3, we also introduce a starting time at the beginning of each assembly period. This time refers to the time of initiating the assembly operation. In industry applications, some adjustments before beginning to assembly operation may be required or the first unit of the product is assembled slowly for adjusting the SME. We assume that this time can be estimated and can be included into the problem formulation a fixed parameter.

1	2	3	4
Installation of Component Packs to Pockets	Blank Body Loading	Starting Time	Assembly of Components

Figure 3.3 Order of events in an assembly period

Component assignment scheme can be different in different assembly periods while assembling a given lot of a product. This means that a component can be assigned to different pockets in different assembly periods. We call this kind of component assignment scheme as “dynamic component assignment (DCA)” and we formulate the DCA problem as a mixed integer problem (MIP) in the following subsection.

3.1 Dynamic Component Assignment (DCA) for a Lot Size

In this section, we introduce our formulation together with its assumptions, and we present our mathematical model for the DCA problem. Below are the assumptions that we utilize in formulating the DCA problem:

- The only movable part of SME is the nozzle unit and it can only assemble single unit of a component in one pick-and-place cycle.
- Nozzle unit is set up for a product and can hold all required nozzles at the same time for the product. This means there is no need to perform a nozzle installation while assembling the whole lot.

- Installation of different component packs to different component pockets takes the same amount of time for all components and for all pockets, and all components can be installed to all pockets.
- Pick-and-place time of components from different pockets are sequence independent. This indicates that pick-and-place time of a component only depends on the pocket to be picked. Sun et al. (2005) argue that due to technological advancements, the component density on a PCB has gradually increased and the distance between the PCB points has decreased. This argument supports our assumption as we assumed that pick-and-place times are independent of the points on the blank body.
- The total number of pockets available in an SMC pocket unit is always greater than or equal to the total number of component types used in a product. In real world applications, the total number of component types assigned to an SME is not greater than the total pocket on that SME if there are multiple SMEs in the line. Even if a line has only one SME, such products are very rare and the SMC pocket unit capacity problem can be handled by assembling these products in multiple assembly runs on SME. If a product requires multiple runs on an SME, we can treat each run as a different product in multiple item case.

Based on these assumptions, our component assignment model differs from the past studies with following properties:

- We try to minimize the total time required to assemble a given lot size while past researches deal with only a single unit of item.
- Our formulation allows using different component assignment schemes in different assembly periods while past studies try to find a unique repetitive component assignment scheme for a product.
- We include component pack capacities in our formulation where reinstallation of components also takes time. This is a critical issue

especially when tray type pockets are used. Ayob and Kendall (2009) state that reinstallation of a tray takes relatively much time, and optimization of tray type of pockets is an unexplored research area. Our formulation is applicable for tray type pockets.

- Our formulation has flexibility of including body loading times and body loading capacities, which is also not included in formulations in the current studies.

The objective of the DCA problem is to minimize the total time of assembling a given lot of a product on SME.

Note that we can handle the body loading process by assuming that the blank body is also a component that has a pack capacity equal to the body loading capacity. In addition, we can force mathematical model to assign bodies to a dummy pocket having a zero pick-and-place time. Therefore, there is no need to add a separate mechanism to track the body loading operation in the mathematical formulation of the problem.

The corresponding indices, parameters and decision variables are given below:

Indices:

i, s : Components, $1, 2, \dots, I$. Note that 1 stands for blank bodies.

j : Pockets, $1, 2, \dots, J$. Note that 1 stands for the dummy pocket.

k : Assembly periods, $1, 2, \dots, K$.

Parameters:

Q : Lot size.

K : Upper bound on the total number of assembly periods.

n_i : Number of component i required for producing one unit of a product.

c_i : Component pack capacity of component i .

a_{ij} : Pick-and-place time of one unit of component i from pocket j .

β : Installation time of one pack of a component to a pocket.

- γ : Starting time of the assembly operation after a set up operation.
 θ : Body loading time.

Decision variables:

$$y_k: \begin{cases} 1, & \text{if an assembly starts in assembly period } k \\ 0, & \text{otherwise} \end{cases}$$

$$f_{kij}: \begin{cases} 1, & \text{if an installation of component } i \text{ to pocket } j \text{ is done in assembly} \\ & \text{period } k \\ 0, & \text{otherwise} \end{cases}$$

u_{kij} : Number of component i uninstalled from pocket j in assembly period k .

p_{kij} : Number of component i at pocket j at the beginning of assembly period k after component installation. ($p_{0ij} = 0$)

g_{kij} : Number of component i used from pocket j in assembly period k ($g_{0ij} = 0$).

q_k : Number of products assembled in assembly period k .

Below is our mathematical model for the dynamic component assignment problem for single-item and we will refer to this model as DCA-SI.

[DCA-SI]

$$\text{Min. } \gamma \sum_{k=1}^K y_k + \theta \sum_{k=1}^K f_{k11} + \beta \sum_{k=1}^K \sum_{i=2}^I \sum_{j=2}^J f_{kij} + \sum_{k=1}^K \sum_{i=2}^I \sum_{j=2}^J g_{kij} a_{ij} \quad (3.1)$$

Subject to:

$$\sum_{j=2}^J f_{k1j} = 0 \quad \forall k \quad (3.2)$$

$$\sum_{i=1}^I f_{kij} \leq y_k \quad \forall k, j \quad (3.3)$$

$$\sum_{k=1}^K q_k = Q \quad (3.4)$$

$$\sum_{j=1}^J g_{kij} = n_i q_k \quad \forall k, i \quad (3.5)$$

$$g_{kij} \leq p_{kij} \quad \forall k, i, j \quad (3.6)$$

$$p_{kij} = p_{(k-1)ij} - g_{(k-1)ij} - u_{kij} + c_i f_{kij} \quad \forall k, i, j \quad (3.7)$$

$$p_{kij} \leq c_i \left(1 - \sum_{s=1 \text{ and } s \neq i}^I f_{ksj} \right) \quad \forall k, i, j \quad (3.8)$$

$$c_i f_{kij} \leq p_{kij} \quad \forall k, i, j \quad (3.9)$$

$$u_{kij} \leq c_i \sum_{s=1}^I f_{ksj} \quad \forall k, i, j \quad (3.10)$$

$$y_k \geq y_{k+1} \quad \forall k: 1, \dots, K-1 \quad (3.11)$$

$$q_k \in \{0, 1, 2, \dots\} \quad \forall k \quad (3.12)$$

$$y_k \text{ and } f_{kij} \in \{0, 1\} \quad \forall k, i, j \quad (3.13)$$

$$u_{kij}, p_{kij} \text{ and } g_{kij} \geq 0 \quad \forall k, i, j \quad (3.14)$$

In DCA-S, the objective function (3.1) minimizes the sum of the total start time, the total body loading time, the total component installation time, and the total assembly time.

Constraint (3.2) ensures that component 1 not installed to any pockets other than pocket 1 throughout all assembly periods. This guarantees that if a body loading has to be done before an assembly period, blank bodies must be installed to dummy pocket. Constraint (3.3) states that at most one type of component can be installed to a pocket at an assembly period. It also guarantees that if there is not a starting operation in an assembly period, there cannot be any set up operation. Together with (3.2), it also ensures that a dummy pocket is always occupied by only blank bodies.

Constraint (3.4) assures that the total number of assembled products is equal to lot size. Constraint (3.5) expresses the component usage from different

pockets at each assembly period and it ensures that the total number of a component used from different pockets is equal to the need of that component in that assembly period. According to constraint (3.6), the total component usage in an assembly period from a component pocket cannot be greater than the available quantity of component at that pocket at the beginning of that assembly period.

Constraint (3.7) gives the balance equation for the quantities of components in pockets between two consecutive assembly periods. Three different cases can be observed for a pocket between two consecutive assembly periods:

1. There is no component installation to a pocket. In this case, $p_{kij} = p_{(k-1)ij} - g_{(k-1)ij}$ must be valid. If there is no component installation to the pocket, we are sure that $c_i f_{kij} = 0$. In addition, $u_{kij} = 0$ must hold and is assured by constraint (3.10).
2. A different component is installed to a pocket. In this case, previously remaining component must be uninstalled and $u_{kij} = p_{(k-1)ij} - g_{(k-1)ij}$ must be valid. Therefore $p_{kij} = 0$ must hold and is assured by constraint (3.8).
3. The same component is reinstalled to a pocket. In this case, $p_{kij} = c_i$ must be valid as components are installed to pockets in term of packs and previously remaining components have to be removed even if the same component is installed. Constraints (3.8) and (3.9) assures that $p_{kij} = c_i$. From the constraint (4.7), $u_{kij} = p_{(k-1)ij} - g_{(k-1)ij}$ also holds in this case and previously remaining amounts are uninstalled.

As K is an upper bound for the total number of assembly periods, there can be some idle assembly periods. An idle assembly period does not include any assembly operation and time of an idle assembly period is zero. These idle periods can produce many alternative solutions if they are not forced to appear closer to K . For example, if there is only one idle assembly period, it should be K^{th} assembly period and if there are two idle assembly periods, they should be $K-1^{th}$ and K^{th}

assembly periods. To force our mathematical formulation to give solutions only in this manner, we introduce constraints (3.11).

Constraints (3.12), (3.13) and (3.14) ensure the properties of decision variables. As n_i , c_i and q_k are integer numbers, the model always gives integer results for variables in constraint (3.14).

The computational requirements of solving DCA-SI mostly depends on K . Therefore, finding minimum possible K value is critical for finding an optimal solution to the problem faster. Note that Q is an upper bound for K . This can be found by assuming that at least one product should be assembled at an assembly period. However, upper bounds that are more efficient than that trivial fact can be found by considering the stopping need of SME. Note that, assembly operation must stop when the blank body capacity is reached and/or a component is finished. Remember that we considered blank body as a component in our formulation. Therefore, for a given lot size Q , the total number of a component (say component i) pack needed to assemble the whole lot can be calculated as $\left\lceil \frac{n_i Q}{c_i} \right\rceil$ where $\lceil x \rceil$ gives the smallest integer value larger than or equal to x . Then the total number of component packs needed to assemble the lot is $\sum_{i=1}^I \left\lceil \frac{n_i Q}{c_i} \right\rceil$. At the beginning of the first assembly period, all I types of components have to be installed simultaneously, and at the worst case, each component reinstallation can require stopping assembly operations separately. So, $1 + \sum_{i=1}^I \left\lceil \frac{n_i Q}{c_i} \right\rceil - I$ is also an upper bound on K , where subtracting I is needed for including the simultaneous installations at the beginning of the first assembly period into calculation and $+1$ is for representing the installations at the beginning of the first assembly period.

These discussions suggest that $K = \text{Min} \left\{ \left(1 + \sum_{i=1}^I \left\lceil \frac{n_i Q}{c_i} \right\rceil - I \right), Q \right\}$ is an upper bound on the total number of assembly periods for assembling Q units of a product.

We illustrate the DCA-SI model and its solution with the following example.

Assume that we have an SME layout given in Figure 3.4 where, the (inner) rectangle represents the body holding unit and the circle represents the camera unit. The SME has seven different pockets. Distance of each pocket to the body holding unit is proportional to how we really see in the figure. We assume that the distance between a pocket and the body holding unit (and the camera unit) equals to the distance from center of the pocket to the center of the body holding unit (and the camera unit). For example, pocket 2 and pocket 3 have the same distance, and are closest pockets to the body holding unit, and pocket 7 is the farthest pocket.

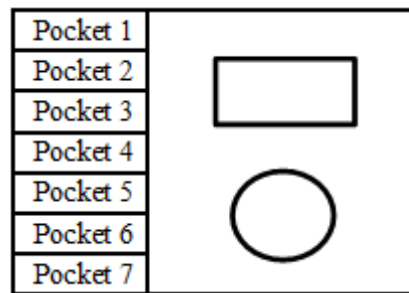


Figure 3.4 SME layout for the example

We use the component data in Table 3.3 for the example product. Note that $i=1$ represents the body of the example product and $j=1$ represents the dummy pocket. Therefore, $j=2$ refers to pocket 1 in Figure 3.4. As can be seen from the table, pick-and-place time for a pocket is the same for all components except $i=5$ as it requires a look-up operation before placing.

This data is collected from an industrial application where a special SME is used for carrier production. Carrier is a type of chip and wire (CW) technology product, which is used in high-tech microelectronic technology products. It is produced in accredited clean rooms with strict contamination, temperature, and pressure specifications. Carriers are composed of a body and several small (in nano metric scale) components mounted to surface of the body. The main characteristic of carriers is the size of the products. Components and products are

very small compared to a PCB product. However, the nature of the production planning problem is the same with any SMT product. For the sake of simplicity, we chose a relatively simple product and restricted the size of SME.

Table 3.3 Parameters for components of example

<i>i</i>	<i>n_i</i>	<i>c_i</i>	<i>a_{ij}</i> (in seconds)							
			<i>j=1</i>	<i>j=2</i>	<i>j=3</i>	<i>j=4</i>	<i>j=5</i>	<i>j=6</i>	<i>j=7</i>	<i>j=8</i>
1	1	10	0	7	5	5	7	9	11	14
2	7	225	0	7	5	5	7	9	11	14
3	17	400	0	7	5	5	7	9	11	14
4	6	256	0	7	5	5	7	9	11	14
5	2	400	0	15	12	11	11	11	12	15

In Table 3.4, an optimal component assignment solution of DCA-SI for the given example for a lot size of 45 is illustrated. The solution is found for the parameters $\beta = 3$ minutes, $\gamma = 10$ minutes, and $\theta = 5$ minutes. In the optimal solution, 45 units of the product are assembled in five assembly periods. These assembly periods are described below:

k=1: As pockets are assumed to be empty at the beginning, all components and blank bodies are installed. As can be seen from the table, only one pack of each component is installed. Component 1 (blank body) is assigned to pocket 1 (dummy pocket), component 2 is assigned to pocket 5, component 3 is assigned to pocket 3, component 4 is assigned to pocket 4, and component 5 is assigned to pocket 6. As blank body capacity is reached, the assembly operation stops after assembling 10 units of the product. If component usages from the pockets are examined, it can be seen that the component assignment is not optimal if we consider only this assembly

Table 3.4 DCA-SI results for $Q=45$

k	i/j	f_{ij}								p_{ij}								g_{ij}							
		1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
1	1	1							10								10								
	2				1								225								70				
	3			1								400								170					
	4				1								256							60					
	5					1									400							20			
2	1	1							10								10								
	2												155								70				
	3											230								170					
	4												196							60					
	5														380							20			
3	1	1							10								5								
	2												85								35				
	3				1							60	400							60	25				
	4			1								256							30						
	5															360							10		
4	1	1							10								10								
	2			1								225		50						70					
	3												375								170				
	4											226							60						
	5															350							20		
5	1	1							10								10								
	2											155		50						70					
	3												205								170				
	4											166							60						
	5															330							20		

period. For example, interchanging the places of component 2 and 4 will give a better assembly time for this period because pocket 5 is farther than pocket 4, and component usage from pocket 4 is 60 and component usage from pocket 5 is 70. However, together with the other assembly periods, DCA-SI gives this solution at optimality.

$k=2$: Only blank body loading is performed and there is no component installations at the beginning of the assembly period 2 because remaining components are sufficient. Again, 10 units of the product are assembled and the assembly operation stops after reaching the blank body capacity.

$k=3$: At the beginning of this assembly period, blank bodies are loaded, a second pack of component 3 is installed to pocket 4 and a second pack of component 4 is installed to pocket 2. Note that remaining amounts are uninstalled from the pocket for component 4 and it is not uninstalled for component 3. Therefore, component 3 occupies two pockets in this assembly period. This installation is mandatory because pack capacity of component 3 is only sufficient for 23 units of the product. After assembling 5 units of the product, the assembly operation stops.

$k=4$: At the beginning of this assembly period, blank bodies are loaded and a second pack of component 2 is installed to pocket 3. Note that all of component 3 is used from pocket 3 in assembly period 3 and pocket 3 is empty before this assembly period. In addition, 50 units of component 2 is remaining at pocket 5. 10 units of the product is assembled in this assembly period and the assembly operation stops after reaching the blank body capacity.

$k=5$: At the beginning of this assembly period, only blank bodies are loaded and 10 units of the product are assembled in this period. Assembly of lot is completed after this assembly period.

We also present the result of preliminary solution runs for 12 different lot sizes in Table 3.5. To find these solutions, Cplex version 12.5.1.0 is used on a computer with 32 bit Windows Operating System, Intel Core 2 Duo 2.13-2.13 GHz CPU and a usable memory of 2.96 GB. Note that 32 bit Windows Operating System limits the maximum amount of memory that a running application can use at 1700 MB. Also maximum CPU time is defined as 3600 seconds for these solutions.

Below, we list the definitions of parameters presented and mention some findings about the solutions given in Table 3.5:

- *Optimality*: In Table 3.5, optimality refers to whether an optimal solution is found or a resource limit is reached while solving DCA-SI for given lot size. As can be seen from the table, optimal solutions are found for all lot sizes except for $Q=100$ where 1700 MB memory limit is reached after finding an integer solution with a relative gap of 2.2% to the best bound found at that time.
- *Number of Idle k* : This row refers to the number of idle assembly periods in given upper bound K . For all lot sizes with $Q \geq 10$ in the table, 10 units of the product are assembled in each non-idle assembly period, which is actually the body loading capacity ($c_I=10$), and all component installation operations are performed when assembly operation stops for body loading needs. Therefore, the total number of non-idle assembly periods for each lot size is equal to $\left\lceil \frac{Q}{10} \right\rceil$.
- *Total Start Time*: This row refers to the total time that spent on the starting of assembly operation for given lot size. The total number of start is also equal to $\left\lceil \frac{Q}{10} \right\rceil$ for our particular problem.
- *Total Body Loading Time*: This row refers to the total time that spent on the body loading operation for assembling the whole lot. The total number of body loading is also equal to $\left\lceil \frac{Q}{10} \right\rceil$ as the assembly operation stops for only

body loading operations in this particular problem. Note that this may not be the case when lot sizes are not expressed as an integer multiple of the body loading capacity or for different problems.

Table 3.5 Preliminary solutions (Times are in minutes)

Time Parameters	$\beta=3 \ \gamma=10 \ \theta=5$											
	1	10	20	30	40	45	50	60	70	80	90	100
Q	1	10	20	30	40	45	50	60	70	80	90	100
K	1	1	2	4	6	8	9	10	12	14	16	19
Optimality	√	√	√	√	√	√	√	√	√	√	√	0.022
Number of Idle k	0	0	0	1	2	3	4	4	5	6	7	9
Total Start Time	10	10	20	30	40	50	50	60	70	80	90	100
Total Body Loading Time	5	5	10	15	20	25	25	30	35	40	45	50
Total Comp. Installation Time	12	12	12	15	18	21	24	24	30	30	36	39
Total Set Up Time	27	27	42	60	78	96	99	114	135	150	171	189
Total Assembly Time	3.07	30.67	61.33	92	122.67	139.83	153.33	184	214.67	245.33	276	306.67
Total Time	30.07	57.67	103.33	152	200.67	235.83	252.33	298	349.67	395.33	447	495.67
Unit Set Up Time	27	2.7	2.1	2	1.95	2.13	1.98	1.9	1.93	1.88	1.9	1.89
Unit Assembly Time	3.07	3.07	3.07	3.07	3.07	3.11	3.07	3.07	3.07	3.07	3.07	3.07
Unit Total Time	30.07	5.77	5.17	5.07	5.02	5.24	5.05	4.97	5	4.94	4.97	4.96
Unit Total Time without Init. Comp. Inst.	18.07	4.57	4.57	4.67	4.72	4.97	4.81	4.77	4.82	4.79	4.83	4.84

- *Total Comp. Installation Time*: This row presents the total time spent on component installation for assembling the given lot. We assumed that all pockets are empty at the beginning of the first assembly period. One pack of each component is installed before the first assembly period in all solutions. If the remaining amount of a component from the previous assembly period is not sufficient for an assembly period, remaining amounts are uninstalled and a new (complete) pack is installed to the pocket at the beginning of the assembly period. Consider the increase in the

total component installation when the lot size increases from 20 to 30. It is the result of installing a second pack of component 3, which has a pack capacity sufficient for 23 products, at the beginning of third assembly period.

- *Total Set Up Time*: This row gives the summation of the total starting time, the total body loading time, and the total component installation time. As can be observed from the table, total set up time increases with increasing lot size.
- *Total Assembly Time*: This row gives the total time spent on picking and placing components for corresponding lot size. Note that the assembly time depends on component assignment schemes used in each assembly period.
- *Total Time*: This row shows the objective value function for each lot size. It is calculated by summing up the total set up time and the total assembly time.
- *Unit Set Up Time*: These values are calculated by dividing the total times with the lot sizes. As can be seen from Table 3.5 and Figure 3.4, unit set up time decreases for increasing lot sizes even for some lot size, unit set up time is increased. This result occurs because all the components have to be installed at the beginning of first assembly period. If the body loading and component pack capacities are sufficient, the decrease is natural in unit set up time. However, body loading and/or component installation operations are needed for increasing lot sizes. Therefore, unit set up time is not monotone decreasing with increasing lot sizes.
- *Unit Assembly Time*: Values in this row is calculated by dividing the total assembly time by corresponding lot size. It can be seen from the table that unit assembly times are equal to each other for all lot sizes. This result shows that component assignment schemes are the same or alternative to each other for all lot sizes. According to results, the assignment scheme given in Table 3.6 is always optimal for lot sizes in Table 3.5. Remember that $i=1$ refers to the blank body and $j=1$ refers to the dummy pocket and DCA-SI forces the blank body to be assigned always to the dummy pocket.

Note that this result is specific to this example and lot sizes. As can be seen from Table 3.5, more used components are assigned to closer pockets.

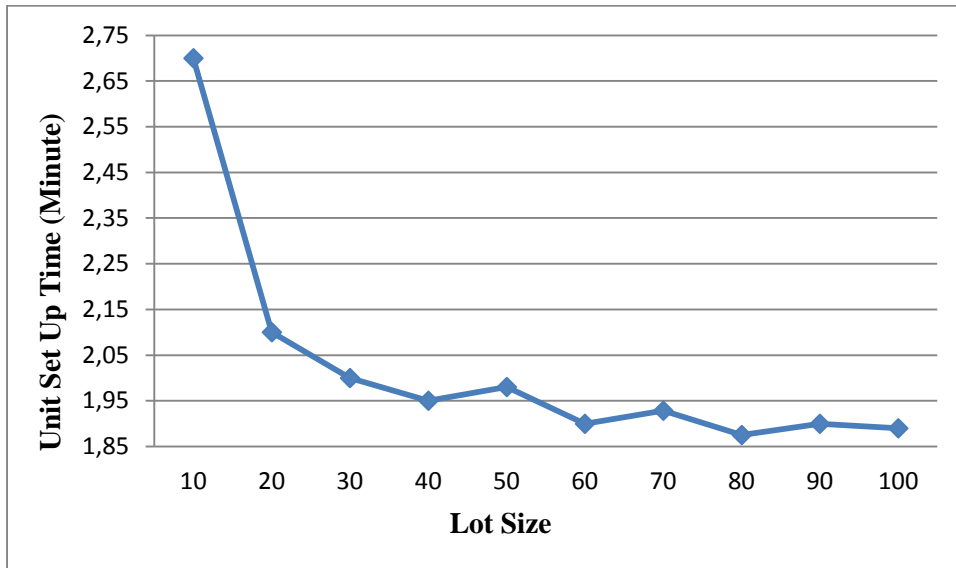


Figure 3.4 Graph for unit set up time for different lot sizes

Table 3.6 Component assignment scheme

$i \setminus j$	1	2	3	4	5
1	1				
2				1	
3			1		
4		1			
5					1

- *Unit Total Time*: Values in this row are calculated by dividing the total times by corresponding lot sizes.
- *Unit Total Time without Init. Comp. Inst.*: This row denotes the unit total time without initial component installation. In this particular problem, we

assumed that pockets are empty at the beginning of the time frame. However, there can be some components at some pockets initially and this can eliminate some component installation needs at the beginning of the first assembly period. In extreme cases, there can be full packs of right components at right pockets initially. In this case, there is no need to perform a component installation at the beginning of the first period. In Table 3.5, such a case represented at the last row. It is calculated by excluding the initial component installation time (12 minutes) from the total time of each lot size. In Figure 3.5, the unit total times and the unit total times without initial component installations are illustrated. As the unit assembly times are equal for all lot sizes, the unit set up times determines the shape of graphs in Figure 3.5. Observe that, the unit total times with and without initial component installations become closer to each other as lot size increases.

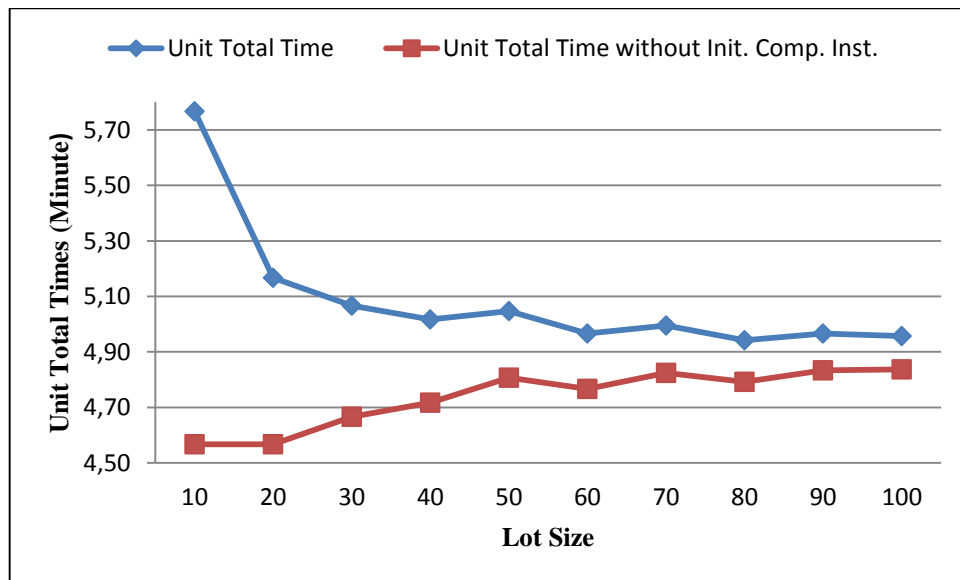


Figure 3.5 Unit total times

These examples show the key properties about the solution of DCA-SI. However, findings are mostly depended on the chosen parameters. For example, DCA-SI finds an optimal solution with the same parameters for a lot size of 45 in which the unit assembly time is greater than what we have found for other lot sizes in Table 3.5. This result occurs because of the optimal component assignment scheme for lot size 45. In the solution for lot size 45, component 3 is assigned to more than one pocket in some assembly periods to decrease the total number of component installations.

3.2 Stationary Component Assignment (SCA) for a Lot Size

In this section, we present a special case for the component assignment problem where a unique component assignment scheme is used in all assembly periods for a given lot size of a product. Note that, the unique component assignment scheme in this special case depends on the lot size; in other words, there can be different schemes for different lot sizes.

In an optimal solution of DCA-SI, there can be several different component assignment schemes in different assembly periods and the same component can be used from multiple pockets in an assembly period. This will require an extra detailed programming of SME in real life applications for different lot sizes, which may not be manageable in practice.

As discussed in chapter 2, an SME requires product specific pick-and-place software which programs SME for the corresponding product. This programming tells SME where to pick components from and where to place them. SME reads codes line by line and performs the action that a code line tells. Going first line to last line generally completes the assembly of one unit of a product, and for each unit, SME goes through the same lines and repeats the actions. Therefore, finding solutions with different component assignment schemes in different assembly periods decreases repetitiveness of the actions, and it can even require coding the assembly of each unit of a product for a given lot size. This fact necessitates finding stationary repetitive component assignment schemes special to lot sizes.

We can find a unique component assignment scheme for a lot size by modifying DCA-SI formulation with the inclusion of following constraint:

$$f_{kij} \leq f_{1ij} \quad \forall k: 2, \dots, K, i, j \quad (3.15)$$

This constraint forces DCA-SI to always install components to pockets that is installed at the beginning of the first assembly period. Therefore, DCA-SI will always give the same component assignment scheme for all assembly periods. We will refer to formulation after including this constraint to DCA-SI as “Stationary Component Assignment for Single-item (SCA-SI)” throughout the text.

3.3 Numerical Experiments

After developing a workable mathematical model for the component assignment problem for both dynamic and stationary cases for a given lot size, it is critical to test performance of these models in terms of solution effort and solution quality. As observed while reviewing literature, there is no related study dealing with the component assignment problem for a given lot size. Therefore, there is no related test data and experimental settings, which we can adapt to test our mathematical models.

3.3.1 Test Data and Experimental Settings

To generate test data, we first define a base product and SME setting, and create test instances by altering key parameters. We use the product that is illustrated in Table 3.3 and SME setting in Figure 3.4 as a base product and SME setting. We create our options by changing the following parameters. Note that we use time parameters as in Table 3.5.

- *Body loading capacity (c_1):* Body loading capacity is one of the factors that stops the assembly operation on an SME. There can be different body

loading capacities for different products. Generally, large products have a body loading capacity of one unit each time. On the other hand, small products can have body loading capacities more than one unit. Also, body loading operation can be automated or manual. If it is automated, it takes less time to load blank bodies on SME. Instead of creating options by changing body loading time, we create four different options by changing the body loading capacity. We test formulations for body loading capacity of 6, 8, 10, and 12.

- *Lot size (Q):* We chose 21 different lot sizes for each body loading capacity where n^{th} lot size is given by the formulation $Q^n = (n - 1) \left(\frac{c_1}{2}\right)$ and $Q^1 = 1$. This lot size scheme is chosen because it gives not only lot sizes that are integer multiple of the body loading capacity but also lot sizes that in between the integer multiple of body loading capacity.
- *Component pack capacities (c_i for $i > 1$):* Component pack capacity is another factor that stops the assembly operation. We use the data in Table 4.3 as base component packs capacities. Then we create options for three cases where component pack capacities are set as: $\left\lceil \frac{c_i}{2} \right\rceil$, c_i , and $2c_i$.

3.3.2 Results

In this section we present the results of our test runs for our component assignment approaches. For each option, the result of the mixed integer (MIP) solutions are provided.

We mainly use following statistics for analysis in our results:

- *CPU:* This statistic denotes the solution time of MIP formulation for the corresponding test instances. Note that termination criteria is set to 7200 CPU seconds in all solutions.
- *IGap%:* This statistic refers to the integrality gap, which is percentage gap between the optimal solution found by the MIP model and lower bound

found by solving the linear relaxation. This statistic is calculated by utilizing the following equation. Note that for instances where MIP model cannot find an optimal solution because of time or memory limit, we use the lower bound found by the solver at last cut.

$$IGap\% = \frac{\text{MIP model solution} - \text{Solution with linear relaxation}}{\text{Solution with linear relaxation}} \times 100$$

- *#opt*: This statistic refers to the number of optimal solution found for an option.

We coded our formulations on GAMS version 24.1.2 and used Cplex version 12.5.1.0 solver with default options on a computer with 32 bit Windows 7 Premium Operating System, Intel Core 2 Duo 2.13-2.13 GHz CPU and a usable memory of 2.96 GB. Note that 32 bit Windows Operating System limits the maximum amount of memory that a running application can use at 1700 MB. Also maximum CPU time is set to 7200 seconds for these solutions.

We present average results in Table 3.7. The first column refers to the test option where the first character represents the body holding capacity option, and the second character represents the component pack capacity option. There are three numbers in the column *#opt*. First number represents the number of optimal solution found in the time limit, second number in rectangular parentheses represents the number of instances where time limit is reached with an integer solution, and the third number in parentheses represents the number of instances where memory limit is reached with an integer solution.

We generate six options with four different body loading capacities and component pack capacities. We only solve different component pack capacity for the body loading capacity option 10. For each option, we solve DCA-SI and SCA-SI for 21 different lot sizes. Detailed results for each option is presented in Appendix A.

Table 3.7 Average results for experiments

	DCA-SI			SCA-SI		
	CPU	IGap%	#opt	CPU	IGap%	#opt
6 x c_i	309.48	15	210	3.47	18	210
8 x c_i	786.76	15	20[1](0)	6.87	15	210
10 x $\lfloor \frac{c_i}{2} \rfloor$	1,539.57	14	153	32.58	14	210
10 x c_i	1,054.33	15	191	10.99	15	210
10 x 2 c_i	278.17	15	210	5.62	15	210
12 x c_i	857.70	15	16[1](4)	17.21	14	210
OVERALL	804.34	14.8	112[6](8)	12.79	15.1	1260

We solved different body loading capacities with different lot sizes as we described in our experimental setting part. For only c_i cases in Table 3.7, solution times are increasing as body loading capacities increase. This is the result of increasing lot sizes by increasing body loading capacity. Note that the average solution time of $(12 \times c_i)$ is smaller than that of $(10 \times c_i)$, which is an opposite case to this observation. This is the result of terminating solutions in $(12 \times c_i)$ for non-optimal solutions because of memory limit. As can be seen from the table, there are 4 non-optimal solutions found at memory limit. This property can be clearly observed for SCA-SI as an optimal solution for each instance is found.

Solution time of SCA-SI is very small compared to DCA-SI. This result is expected because there is only one component assignment scheme decision in SCA-SI and this scheme is used throughout all assembly periods. On the other hand, DCA-SI searches for a component assignment scheme at each assembly period.

For the case where the body loading capacity is 10, while component pack capacities are increasing, the solution time decreases and the number of optimal solutions increases. Remember that we set $K = \text{Min} \left\{ \left(1 + \sum_{i=1}^l \left\lfloor \frac{n_i Q}{c_i} \right\rfloor - I \right), Q \right\}$ as an upper bound on the total number of assembly periods for a given lot size Q . This upper bound decreases with increasing component pack capacities. Therefore, the total number of decisions decreases and the probability of finding an optimal

solution with given time and memory limitation increases. Note that the upper bound on the total number of assembly period increases with increasing lot sizes and finding an optimal solution requires more effort.

As can be seen from Table 3.7, overall average IGap% is greater in SCA-SI than DCA-SI. For different lot sizes of a case, we can observe that IGap% is decreasing with increasing lot sizes, even if it is not monotone decreasing. This property can be seen in Figure 3.6 where detailed IGap% information is illustrated for different lot sizes of case ($10 \times c_i$) for DCA-SI.

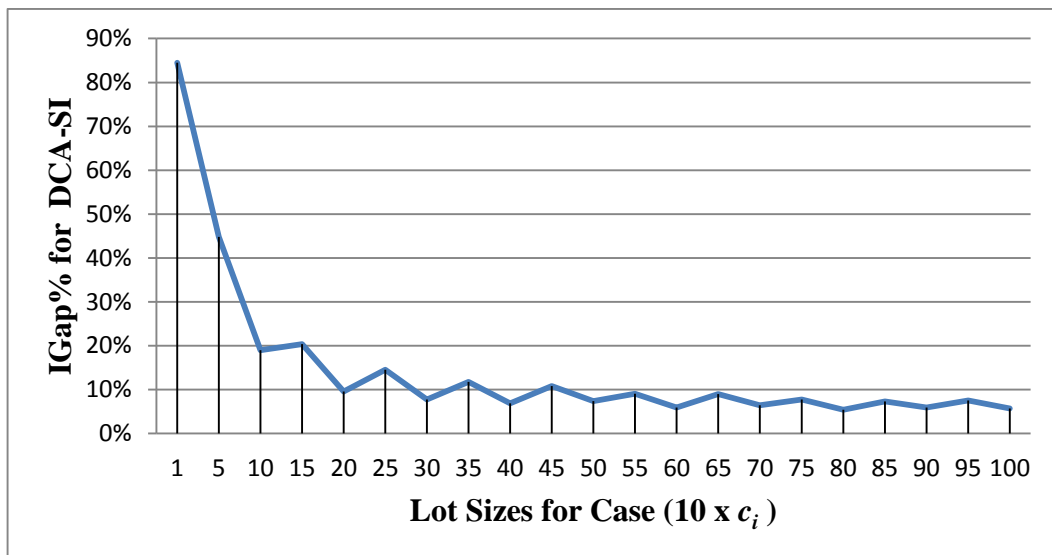


Figure 3.6 IGap% information for different lot sizes of case ($10 \times c_i$) for DCA-SI

If the lot size is small compared to pack capacities, the linear relaxation can assign fractional component packs only enough for assembling given lot size. On the other hand, DCA-SI has to assign components in integer multiples of packs, as the variable is binary. This creates big IGap% for small lot sizes. For big lot sizes, both MIP and linear relaxation have to install enough components for assembling the lot size where more than one pack of some components are needed. Therefore, IGap% decreases as lot size increases.

We present a comparison of the objective function values of DCA-SI and SCA-SI and their associated CPU times in Table 3.8. We define two statistics as following. Note that nonoptimal instances are not taken into account while calculating these statistics.

- *%Dev*: Percentage deviation of the solution of SCA-SI from DCA-SI. This statistic is calculated by utilizing the following equation.

$$\%Dev = \frac{(\text{SCA} - \text{SI result}) - (\text{DCA} - \text{SI result})}{(\text{DCA} - \text{SI result})} \times 100$$

- *%Time*: Percentage of CPU time for SCA-SI compared to DCA-SI. This statistic is calculated by utilizing the following equation.

$$\%Time = \frac{\text{CPU time of SCA} - \text{SI}}{\text{CPU time of DCA} - \text{SI}} \times 100$$

Table 3.8 Comparative results of SCA-SI and DCA-SI

Option	%Dev			%Time		
	Min	Average	Max	Min	Average	Max
6 x c_i	0.00	0.30	4.64	0.45	46.67	148.94
8 x c_i	0.00	0.01	0.20	0.61	38.30	145.00
10 x $\left\lceil \frac{c_i}{2} \right\rceil$	0.00	0.04	0.25	0.99	44.89	201.61
10 x c_i	0.00	0.04	0.49	0.10	37.70	231.91
10 x 2 c_i	0.00	0.00	0.00	1.17	54.64	175.81
12 x c_i	0.00	0.03	0.42	1.09	88.08	244.04
OVERALL	0.00	0.07	4.64	0.10	51.71	244.04

Note that these %Dev come from the fact that DCA-SI can use different component assignment schemes in different assembly periods if it is better in terms of the total time. As can be seen from Table 3.8, SCA-SI performs very

close to DCA-SI while finding an optimum solution takes half of the CPU time on the average compared to DCA-SI. For very small lot sizes, CPU time of both DCA-SI and SCA-SI is a fraction of a second in most of the instances. For these instances, CPU time of SCA-SI is greater than CPU time of DCA-SI. That is why maximum values of %Time is great for all options.

Up to this point, we only consider one type of SME layout, which is illustrated in Figure 3.4. However, solution efforts and solution qualities also depend on the SME layout. We also consider two different new layouts for SME in our experiment. In Figure 3.7, an SME layout is illustrated and we call it (3x2) layout as it has six component pockets composed of two side by side rows. The inner rectangle represents body holder unit and the circle represents the camera unit. Note that we do not include Pocket 1 in Figure 3.7 as we refer to it as a dummy pocket. Table 3.9 shows component related data for (3x2) layout.

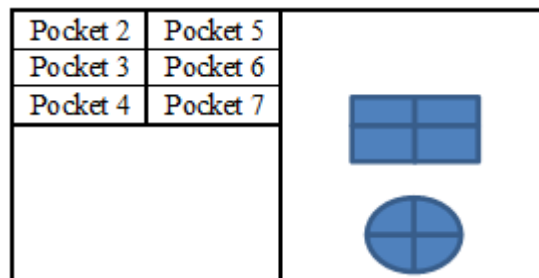


Figure 3.7 (3x2) layout

Table 3.9 Component data for (3x2) layout

			<i>a_{ij}</i> (in seconds)						
<i>i</i>	<i>n_i</i>	<i>c_i</i>	<i>j=1</i>	<i>j=2</i>	<i>j=3</i>	<i>j=4</i>	<i>j=5</i>	<i>j=6</i>	<i>j=7</i>
1	1	10	0	6	6	6	5	4	4
2	7	225	0	6	6	6	5	4	4
3	17	400	0	6	6	6	5	4	4
4	6	256	0	6	6	6	5	4	4
5	2	400	0	12	10	10	10	9	8

Table 3.10 shows the average results for (3x2) layout. As it can be observed, the results are similar with the results shown in Table 3.7. However IGap% values are greater than for (3x2) layout.

Table 3.10 Average results for (3x2) layout

	DCA-SI			SCA-SI		
	CPU	IGap%	#opt	CPU	IGap%	#opt
6 x c_i	304.62	74	210	5.24	74	210
8 x c_i	873.96	85	20[1](0)	7.92	85	210
10 x $\left[\frac{c_i}{2}\right]$	3,252.21	56	14[5](2)	40.81	56	210
10 x c_i	1,039.55	95	18[1](2)	14.25	95	210
10 x 2 c_i	491.50	120	20[1](0)	4.73	120	210
12 x c_i	1,314.24	104	16[2](3)	42.53	104	210
OVERALL	1,212.68	89	109[10](7)	19.25	89	1260

In Figure 3.8, another SME layout is illustrated and we call it (4x2) layout as it has 8 component pockets composed of two side by side rows. Table 3.11 shows component related data for (4x2) layout.

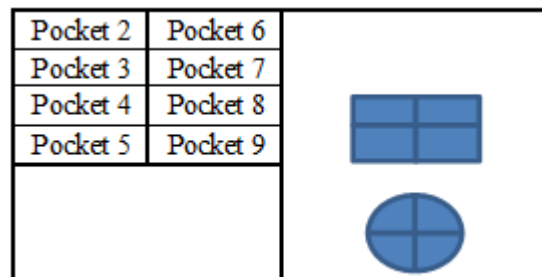


Figure 3.8 (4x2) layout

Table 3.11 Component data for (4x2) layout

			<i>a_{ij} (in seconds)</i>								
<i>i</i>	<i>n_i</i>	<i>c_i</i>	<i>j=1</i>	<i>j=2</i>	<i>j=3</i>	<i>j=4</i>	<i>j=5</i>	<i>j=6</i>	<i>j=7</i>	<i>j=8</i>	<i>j=9</i>
1	1	10	0	6	6	6	6	5	4	4	5
2	7	225	0	6	6	6	6	5	4	4	5
3	17	400	0	6	6	6	6	5	4	4	5
4	6	256	0	6	6	6	6	5	4	4	5
5	2	400	0	12	10	10	9	10	9	8	8

Table 3.12 shows the average results for (4x2) layout. When compared to (3x2) layout, IGap% is slightly small and CPU times are slightly big.

Table 3.12 Average results for (4x2) SME layout

	DCA-SI			SCA-SI		
	CPU	IGap%	#opt	CPU	IGap%	#opt
6 x <i>c_i</i>	281.29	72	210	6.54	72	210
8 x <i>c_i</i>	794.61	84	20[1](0)	7.42	84	210
10 x $\left\lceil \frac{c_i}{2} \right\rceil$	2,216.75	55	14[5](2)	46.29	55	210
10 x <i>c_i</i>	1,615.97	93	18[1](2)	13.52	93	210
10 x 2 <i>c_i</i>	224.93	118	20[1](0)	4.70	118	210
12 x <i>c_i</i>	2,157.40	102	16[2](3)	40.82	102	210
OVERALL	1,215.16	87	109[10](7)	19.88	87	1260

3.4 Case Study

In this section, we consider a particular product from a real life application and solve the component assignment problem with different lot sizes. We compare the results with the given component assignment scheme in real life application and our aim is to demonstrate the significance of the component assignment problem.

SME layout for the case is illustrated in Figure 3.9. The inner rectangle represents the body holder unit and the circle represents the camera unit. SME has two pocket units which are actually trays with a total of 24 pockets. We give a number for each pocket starting from 2 and assume that pocket 1 refers to the dummy pocket.

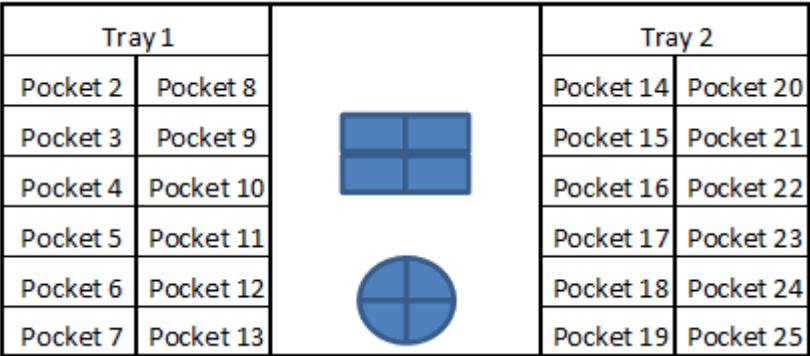


Figure 3.9 SME layout for the case

In Table 3.13, information about the components is given. There are 13 types of components used for the product. In the table, we also include the blank body ($i=1$) and dummy pocket ($j=1$). Components 2, 3 and 13 have look-up option.

Figure 3.10 illustrates the component assignment scheme currently used while assembling the product. Numbers in the pockets refer to components (index i in Table 3.13). With this assignment scheme, unit assembly time for the product is 2 minutes.

Table 3.13 Component information for the case study

$j i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14
n_i	1	1	1	1	1	5	2	3	2	2	1	1	2	1
c_i	10	49	100	400	400	121	144	144	400	400	100	256	81	100
a_{ij} (in seconds)	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	6	12	12	6	6	6	6	6	6	6	6	6	12
	3	6	10	10	6	6	6	6	6	6	6	6	6	10
	4	6	10	10	6	6	6	6	6	6	6	6	6	10
	5	6	9	9	6	6	6	6	6	6	6	6	6	9
	6	8	10	10	8	8	8	8	8	8	8	8	8	10
	7	9	10	10	9	9	9	9	9	9	9	9	9	10
	8	5	10	10	5	5	5	5	5	5	5	5	5	10
	9	4	9	9	4	4	4	4	4	4	4	4	4	9
	10	4	8	8	4	4	4	4	4	4	4	4	4	8
	11	5	8	8	5	5	5	5	5	5	5	5	5	8
	12	6	8	8	6	6	6	6	6	6	6	6	6	8
	13	8	9	9	8	8	8	8	8	8	8	8	8	9
	14	5	10	10	5	5	5	5	5	5	5	5	5	10
	15	4	9	9	4	4	4	4	4	4	4	4	4	9
	16	4	8	8	4	4	4	4	4	4	4	4	4	8
	17	5	8	8	5	5	5	5	5	5	5	5	5	8
	18	6	8	8	6	6	6	6	6	6	6	6	6	8
	19	8	9	9	8	8	8	8	8	8	8	8	8	9
	20	6	12	12	6	6	6	6	6	6	6	6	6	12
	21	6	10	10	6	6	6	6	6	6	6	6	6	10
	22	6	10	10	6	6	6	6	6	6	6	6	6	10
	23	6	9	9	6	6	6	6	6	6	6	6	6	9
	24	8	10	10	8	8	8	8	8	8	8	8	8	10
	25	9	10	10	9	9	9	9	9	9	9	9	9	10



Tray 1			Tray 2	
	13			2
4	14			
9	11			
12	10			
5	8			
7	6			

Figure 3.10 Component assignment scheme for the case study

Table 3.14 presents the results that we obtain with dynamic and component assignment models, stationary component assignment with $Q=1$, and the results with current assignment. In the table, SCA with $Q=1$ refers to the results by using optimal component assignment scheme that is found for the lot size of 1 unit of item throughout all assembly periods. For DCA-SI and SCA-SI, we only include cases where an optimal solution is found.

Figure 3.11 shows the %Dev between the result with current assignment and the solution of SCA with $Q=1$.

As can be seen from Table 3.14 and Figure 3.11, Given SCA is not efficient compared to other solutions. DCA-SI, SCA-SI and SCA with $Q=1$ give the same objective function values at optimality and it is always better than the results with Given SCA. The chosen product is a relatively small sized product in terms of the number of component types used compared to other products in the industry. However, we can only find solutions for some cases where the lot size is relatively small by using DCA-SI and SCA-SI. On the other hand, we can find solutions with SCA with $Q=1$ for all lot sizes and the solution is very close to DCA-SI. Figure 3.11 shows that %Dev increases with increasing lot size. This indicates that using an inefficient component assignment scheme will cost more with increasing lot size.

Table 3.14 Results for the case study (Times are in minutes)

Lot Size	DCA_SI	SCA_SI	SCA with $Q=1$	Given SCA
1	56	56	56	57
5	64	64	64	68
10	74	74	74	82
15	99	99	99	111
20	108	108	108	125
25	-	136	136	157
30	146	146	146	171
35	-	-	171	200
40	181	181	181	214
45	-	-	212	249
50	-	-	227	269
55	-	-	252	298
60	-	-	262	312
65	-	-	290	344
70	-	-	300	358
75	-	-	328	390
80	-	-	337	404
85	-	-	368	439
90	-	-	384	459
95	-	-	403	482
100	-	-	419	502

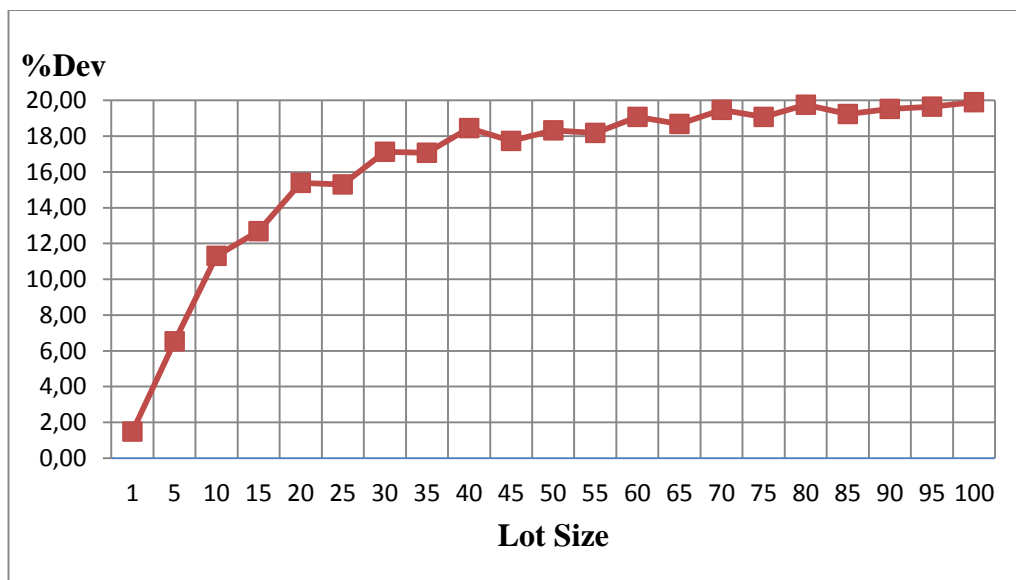


Figure 3.11 Comparative results for the case

CHAPTER 4

THE SINGLE-ITEM LOT SIZING PROBLEM

In this chapter, we present the single-item lot sizing problem (SLP) in single SME production lines. Although it is not common to dedicate an SME to only one type of product in an electronic manufacturing environment, we believe that studying the problem for a single-item is beneficial for understanding the interconnections between the component assignment and lot sizing problems, and that it will provide valuable insights about the multi-item case.

In this chapter, we combine the lot sizing problem with the component assignment problem and present our model for the single-item case. While formulating, we modify the facility location based formulation of the SLP, which is a strong formulation for the SLP. We present both uncapacitated and capacitated forms of the model with dynamic and stationary component assignment policies.

Figure 4.1 illustrates the planning horizon in terms of production periods and assembly periods. In this figure, production periods are represented by t and assembly periods are represented by k . A production period is composed of several assembly periods and demands occur at the end of each production period.

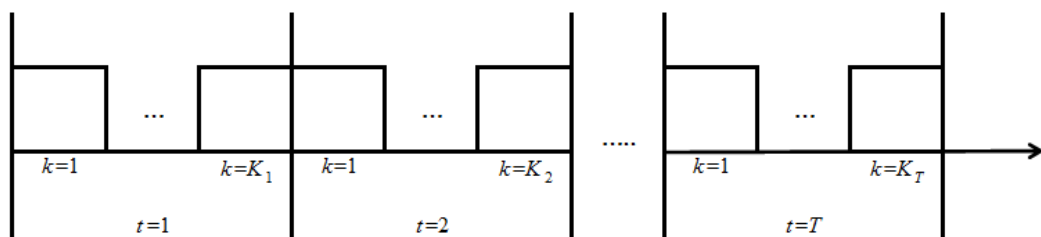


Figure 4.1 Production periods and assembly periods

At the beginning of each production period, a major set up time is required if there is an assembly operation throughout the corresponding production period. Demand is deterministic and nonstationary throughout the planning horizon. Therefore, we face a single-item lot sizing problem with dynamic demands.

The lot sizing problem that we consider is to find the lot sizes for each production period to satisfy the demand without backordering while minimizing the total system cost throughout the planning horizon. Related costs are assembly operation, set up, major set up and inventory holding costs. Below are basic assumptions that we make in the formulation. Note that the assumptions related to the component assignment problem, which are discussed in Chapter 3, are still valid.

- All pockets of the SME are empty at the beginning of each production period throughout the planning horizon.
- Inventory is not allowed both at the beginning and at the end of the planning horizon.
- If an assembly operation takes place in a production period, there is a major set up at the beginning of that production period. These major set up operations can be nozzle set up or can be adjusting parts of SME for the corresponding product.
- Costs are known and stationary.
- Overtime is not possible.

4.1 Uncapacitated Lot Sizing with DCA

In this section, we present our mathematical model for the uncapacitated single-item lot sizing problem with dynamic component assignment (USL-DCA). The objective of the USL-DCA is to minimize the total cost for satisfying demand throughout the planning horizon without backlogging. Below are the indices, parameters and decision variables we introduce for the USL-DCA:

Indices:

- t, z : Production periods, $1, 2, \dots, T$
 i, s : Components, $1, 2, \dots, I$. Note that 1 stands for blank bodies.
 j : Pockets, $1, 2, \dots, J$. Note that 1 stands for the dummy pocket.
 k : Assembly periods, $1, 2, \dots, K$.

Parameters:

- d_t : Demand in production period t .
 K_t : Upper bound on the total number of assembly periods in production period t .
 τ : Major set up time of SME at the beginning of a production period.
 ε : Utilization cost of SME for one unit of time. This cost can be estimated by taking into account the wage of operators, opportunity cost of non-productive usage of time, and overhead costs related to SME.
 h : Unit holding cost per production period. This cost can be estimated by multiplying the total cost of finished product with the interest rate related with the length of production period.
 n_i : Number of component i required for producing one unit of a product.
 c_i : Component pack capacity of component i .
 a_{ij} : Pick-and-place time of one unit of component i from pocket j .
 β : Installation time of one pack of a component to a pocket.
 γ : Starting time of the assembly operation after a set up operation.
 θ : Body loading time.

Decision variables:

- $y_{kt} : \begin{cases} 1, & \text{if an assembly operation starts in assembly period } k \text{ of production} \\ & \text{period } t \\ 0, & \text{otherwise} \end{cases}$
 $f_{ktij} : \begin{cases} 1, & \text{if an installation of component } i \text{ to pocket } j \text{ is done in assembly} \\ & \text{period } k \text{ of production period } t \\ 0, & \text{otherwise} \end{cases}$

u_{ktij} : Number of component i uninstalled from pocket j in assembly period k of production period t .

p_{ktij} : Number of component i available at pocket j at the beginning of assembly period k of production period t after component installation operations. ($p_{0tij} = 0$)

g_{ktij} : Number of component i used from pocket j in assembly period k of production period t . ($g_{0tij} = 0$).

q_{kzt} : Number of product assembled in assembly period k of production period z to satisfy demand at production period t for $z \leq t$.

[USL-DCA]

$$\begin{aligned} \text{Minimize } & \sum_{t=1}^T \left[\varepsilon \left(\tau y_{1t} + \gamma \sum_{k=1}^{K_t} y_{kt} + \theta \sum_{k=1}^{K_t} f_{kt11} + \beta \sum_{k=1}^{K_t} \sum_{i=2}^I \sum_{j=2}^J f_{ktij} \right. \right. \\ & \left. \left. + \sum_{k=1}^{K_t} \sum_{i=2}^I \sum_{j=2}^J g_{ktij} a_{ij} \right) + h \sum_{z=1}^t \sum_{k=1}^{K_z} q_{kzt} (t - z) \right] \end{aligned} \quad (4.1)$$

Subject to:

$$\sum_{j=2}^J f_{kt1j} = 0 \quad \forall k: 1, 2, \dots, K_t, t \quad (4.2)$$

$$\sum_{i=1}^I f_{ktij} \leq y_{kt} \quad \forall k: 1, 2, \dots, K_t, j, t \quad (4.3)$$

$$\sum_{z=1}^t \sum_{k=1}^{K_z} q_{kzt} = d_t \quad \forall t \quad (4.4)$$

$$\sum_{j=1}^J g_{kzij} = n_i \sum_{t=z}^T q_{kzt} \quad \forall k: 1, 2, \dots, K_z, i, z \quad (4.5)$$

$$g_{ktij} \leq p_{ktij} \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.6)$$

$$p_{ktij} = p_{(k-1)tij} - g_{(k-1)tij} - u_{ktij} + c_i f_{ktij} \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.7)$$

$$p_{ktij} \leq c_i \left(1 - \sum_{s=1 \text{ and } s \neq i}^I f_{ktsj} \right) \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.8)$$

$$c_i f_{ktij} \leq p_{ktij} \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.9)$$

$$u_{ktij} \leq c_i \sum_{s=1}^I f_{ktsj} \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.10)$$

$$y_{kt} \geq y_{(k+1)t} \quad \forall k: 1, 2, \dots, K_t - 1, t \quad (4.11)$$

$$q_{kzt} \in \{0, 1, 2, \dots\} \quad \forall k: 1, 2, \dots, K_z, t, z, \text{ where } z \leq t \quad (4.12)$$

$$y_{kt} \text{ and } f_{ktij} \in \{0, 1\} \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.13)$$

$$u_{ktij}, p_{ktij}, \text{ and } g_{ktij} \geq 0 \quad \forall k: 1, 2, \dots, K_t, i, j, t \quad (4.14)$$

In USL-DCA, the objective function (4.1) minimizes the total cost. Note that the total time that the SME is utilized is converted into monetary terms by multiplying it with the cost of utilizing SME for one unit of time. Note that we do not define a binary variable to track for whether a major set up occurs at the beginning of each production period. If there is an assembly operation in a production period, y_{1t} must be equal to 1 (observe how constraints (4.3) and (4.11) guarantee this if there is an assembly operation), and at optimality, y_{1t} must be equal to 0 if there is no assembly operation in a production period.

Constraint (4.2) ensures that component 1 is not installed to any pockets other than pocket 1. This guarantees that if a body has to be loaded before an assembly period starts, blank bodies must be installed to the dummy pocket ($j=1$). Constraint (4.3) states that at most one type of component can be installed to a pocket at the same time and it guarantees that if there is not a starting operation before an assembly period starts, there cannot be done any set up operation. Together with (4.2), it also ensures that the dummy pocket is always occupied by only blank bodies.

Constraint (4.4) assures that the total number of products assembled in production periods 1 to t for production period t , equals to the demand in production period t .

Constraint (4.5) expresses the component usage from different pockets in each assembly period. It ensures that the total number of a component used from different pockets is equal to the need of that component in that assembly period. According to constraint (4.6), the total component usage in an assembly period from a component pocket cannot be greater than the available quantity of component at that pocket at the beginning of that assembly period.

Constraint (4.7) gives the balance equation for the quantities of the components in pockets between two consecutive assembly periods. There can be three cases for a pocket between two consecutive assembly periods:

1. There is no component installation to a pocket. In this case, $p_{ktij} = p_{(k-1)kij} - g_{(k-1)tij}$ must be valid. If there is no component installation to that pocket we are sure that $c_i f_{ktij} = 0$. In addition, $u_{ktij} = 0$ must hold and constraint (4.10) assures this.
2. A different component is installed to a pocket. In this case, previously remaining component must be uninstalled and $u_{ktij} = p_{(k-1)tij} - g_{(k-1)tij}$ must be valid. Therefore $p_{ktij} = 0$ must hold and constraint (4.8) assures this.
3. The same component is reinstalled to a pocket. In this case, $p_{ktij} = c_i$ must be valid as components are installed to pockets in term of packs and previously remaining components have to be removed even if the same component is installed. Constraints (4.8) and (4.9) assures that $p_{ktij} = c_i$. From constraint (4.7), $u_{ktij} = p_{(k-1)tij} - g_{(k-1)tij}$ also holds in this case and previously remaining components are uninstalled.

As K_t is an upper bound on the total number of assembly periods in a production period, there can be some idle assembly periods at this production period. Since an idle assembly period does not include any assembly operations, time used in an idle assembly period will be zero. These idle periods can produce many alternative solutions if they are not forced to appear closer to K_t . For

example, if there is only one idle assembly period, it should be K_t^{th} assembly period and if there are two idle assembly periods, they should be $(K_t - 1)^{th}$ and K_t^{th} assembly periods. To force our mathematical formulation to give solutions only in this manner, we introduce constraints (4.11).

Constraints (4.12), (4.13) and (4.14) state the properties of decision variables. As n_i , c_i and q_{kzt} are integer numbers, the model always gives integer results for variables in constraint (4.14).

The maximum amount that can be assembled in a production period (say production period z) equal to $\sum_{t=z}^T d_t$. As discussed in Section 3.1, $K_z = \text{Min} \left\{ \left(1 + \sum_{i=1}^I \left\lfloor \frac{n_i \sum_{t=z}^T d_t}{c_i} \right\rfloor - I \right), \sum_{t=z}^T d_t \right\}$ is an upper bound on the total number of assembly periods for production period z , where $[b]$ is the smallest integer greater than or equal to b .

4.2 Capacitated Lot Sizing with DCA

In this section, we also consider the capacity constraint on the total number of product that can be assembled in a production period. Capacitated form of the USL-DCA can be obtained by including the following constraint in the model.

$$\sum_{t=z}^T \sum_{k=1}^{K_z} q_{kzt} \leq CA \quad \forall z \quad (4.15)$$

In (4.15), CA refers to the maximum number of product that can be assembled in a production period. We will refer to this model as CSL-DCA.

4.3 Lot Sizing with SCA

As discussed in Chapter 3, stationary component assignment policy implies keeping component assignment scheme fixed throughout all assembly periods while assembling a given lot size. As we have multiple production periods in the

lot sizing problem, there can be different lot sizes to be assembled for different production periods. Therefore, in the context of lot sizing, stationary component assignment policy refers to keeping component assignment scheme fixed throughout all assembly periods in a production period. This means, there can be different component assignment schemes in different production periods; but, in a production period, component assignment scheme is unique.

By including inequality (4.16) in the model, a unique component assignment scheme for each production period can be obtained. We will refer to this model as “Uncapacitated Single-Item Lot Sizing with Stationary Component Assignment (USL-SCA)”, and the model after including this constraint to CSL-DCA as “Capacitated Single-Item Lot Sizing with Stationary Component Assignment (CSL-SCA)”.

$$f_{ktij} \leq f_{1tij} \quad \forall k: 2, \dots, K_t, i, j, t \quad (4.16)$$

Note that constraint (4.16) forces to install components to the pockets at all assembly periods as the same way in which they are installed at the beginning of the first assembly period of a production period.

Although CSL-SCA gives solutions with unique component assignment schemes for each production period, one can consider finding a unique component assignment scheme for a product independent of demands and lot sizes. In following subsections, we present three special cases of SCA where component assignment scheme is predefined and unique for a product throughout the entire planning horizon.

4.3.1 SCA for $Q=1$

In industrial applications, finding a component assignment scheme for single unit of a product and using it while assembling the product is a common practice. Main advantage of this approach is its simplicity in both finding solutions to the component assignment problem and carrying over the solutions to the

manufacturing environment. However, we are not able to conclude that all industrial applications use the component assignment scheme based on assembling a single unit. Nevertheless, we consider such a scheme as a “base” scheme in our assessments of different policies.

If a predefined component assignment scheme is to be introduced in a formulation which minimizes the total time of assembling a given lot, it must minimize the assembly time. Therefore, introducing the scheme that minimizes unit assembly time of a single-item is a rational decision. Note that solving DCA-SI for a single unit always gives the minimum assembly time for that item. Below is the mathematical model for the lot sizing problem with this special approach for the SCA, which will be referred as USL-SCA1. Note that we only introduce the modified parts of the model USL-DCA for the case of SCA for single unit of an item.

Parameters:

a_i : Unit assembly time of component i . Note that it is known since assignment is predefined.

Decision variables:

f_{kti} : $\begin{cases} 1, & \text{if an installation of component } i \text{ is done in assembly period } k \text{ of} \\ & \text{production period } t \\ 0, & \text{otherwise} \end{cases}$

u_{kti} : Number of component i uninstalled in assembly period k of production period t .

p_{kti} : Number of component i available at the beginning of assembly period k of production period t after component installation.

g_{kti} : Number of component i used at assembly period k of production period t .

[USL-SCA1]

$$\text{Minimize } \sum_{t=1}^T \left[\varepsilon \left(\tau y_{1t} + \gamma \sum_{k=1}^{K_t} y_{kt} + \theta \sum_{k=1}^{K_t} f_{kt1} + \beta \sum_{k=1}^{K_t} \sum_{i=2}^I f_{kti} \right. \right. \\ \left. \left. + \sum_{k=1}^{K_t} \sum_{i=2}^I g_{kti} a_i \right) + h \sum_{z=1}^t \sum_{k=1}^{K_t} q_{kzt} (t-z) \right] \quad (4.17)$$

Subject to:

$$f_{kti} \leq y_{kt} \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.18)$$

$$\sum_{z=1}^t \sum_{k=1}^{K_z} q_{kzt} = d_t \quad \forall t \quad (4.19)$$

$$g_{kzi} = n_i \sum_{t=z}^T q_{kzt} \quad \forall k: 1, 2 \dots K_z, i, z \quad (4.20)$$

$$g_{kti} \leq p_{kti} \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.21)$$

$$p_{kti} = p_{(k-1)ti} - g_{(k-1)ti} - u_{kti} + c_i f_{kti} \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.22)$$

$$p_{kti} \leq c_i \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.23)$$

$$u_{kti} \leq c_i f_{kti} \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.24)$$

$$y_{kt} \geq y_{(k+1)t} \quad \forall k: 1, \dots, K_t - 1, t \quad (4.25)$$

$$q_{kzt} \in \{0, 1, 2, \dots\} \quad \forall k: 1, 2 \dots K_z, t, z, \text{ where } z \leq t \quad (4.26)$$

$$y_{kt} \text{ and } f_{kti} \in \{0, 1\} \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.27)$$

$$u_{kti}, p_{kti} \text{ and } g_{kti} \geq 0 \quad \forall k: 1, 2 \dots K_t, i, t \quad (4.28)$$

In USL-SCA1, the objective function (4.17) minimizes the total cost throughout the planning horizon. Constraint (4.18) states that a component installation cannot be performed without stopping the assembly operation. Constraint (4.19) assures that demand is met. Constraint (4.20) assures the component usage as they are needed. Constraint (4.21) guarantees that the total number of a component used in an assembly period cannot be greater than the number of available components in that assembly period. Constraint (4.22) gives the balance equation for the quantities of the components in SME and constraint

(4.23) guarantees that at most one pack of a component can be available in an assembly period. Constraint (4.24) assures that, if there is not an installation for a component, that component cannot be uninstalled. All remaining constraints are the same as the ones in USL-DCA and K_t is estimated similarly.

In this special case, indices related to the pockets are completely removed from the formulation. The only remaining decision related to the component installation is when a component will be installed. Note that in this special case, a component is assigned to only one pocket and multiple pack installation for a component at the same time is not possible.

4.3.2 SCA with Incremental Q

In this case, initially, exactly one pack of each component is assigned to pockets optimally. Note that, this assignment is the optimal component assignment scheme with $Q=1$. After that, Q is increased and if a component is not sufficient for Q , a new pack of that component is assigned to an empty pocket, which gives the best pick-and-place time for that component, without changing the places of initially assigned components. This process is carried out until all pockets are occupied by a component. Let e_{ij} be the component assignment scheme after utilizing all pockets such that if component i is assigned to pocket j , $e_{ij} = 1$, otherwise it is zero. This procedure gives a component assignment scheme where $\sum_{j=1}^J e_{ij} \geq 1$ is valid for all i and $\sum_{j=1}^J e_{ij} > 1$ may hold for some components. Once e_{ij} values are found in this manner, one can find solutions by including inequality (4.29) into the USL-DCA and we refer to this special case as USL-SCA2. Inequality (4.29) restricts the number of pockets that a component can be installed at each assembly period.

$$f_{ktij} \leq e_{ij} \quad \forall k, i, j, t \quad (4.29)$$

This approach gives the flexibility of assigning multiple packs of some components on top of USL-SCA1 if it gives better results in terms of the total cost. If each component is assigned to only one pocket at optimality with USL-SCA2, USL-SCA2 gives the same solution with USL-SCA1. On the other hand, if it is advantageous to install multiple packs of some components at the same time, USL-SCA2 can give better results than USL-SCA1. As the assignments of components are incrementally fixed, the optimality of an assignment of multiple packs of components are not guaranteed for USL-SCA2.

4.3.3 SCA with Full Utilization of Pockets

On an SME, SMC pocket unit has a limited number of pockets. When component pack capacities and component usage of a product are known, the maximum number of a product that can be assembled by utilizing all pockets can be calculated. Let Q_j be the number of a product that can be assembled by fully utilizing all component pockets. Solving DCA-SI with $Q = Q_j$, $K = \left\lceil \frac{Q_j}{c_1} \right\rceil$, and with constraint $f_{kij} = 0$ for $k:2,3,..K$, $i:2,3,..I$, and for all j . Note that this constraint does not allow any component installation except for the first assembly period ($k=1$). Therefore the solution will give a component assignment scheme where all pockets are utilized at the beginning of the first assembly period. By letting $e_{ij} = f_{1ij}$, one can find solutions by including inequality (4.29) into USL-DCA and we refer to this special case as USL-SCA3.

This approach seeks the best assignment of multiple packs of components onto the SMC pocket unit. However, if it is not beneficial to use multiple packs at the same time in terms of the total cost, USL-SCA1 and USL-SCA2 can give better results than USL-SCA3.

4.4 Numerical Experiments

Having developed a mathematical model according to our problem formulation, the main goal in this section is to obtain more insight about the behavior of the models and to consider how we can make the best use of these models and information we have developed thus far. We examine the solution quality of our models in different cases. In this section, we present our experimental settings and the results of our numerical experiments for USL-DCA, CSL-DCA, USL-SCA, and CSL-SCA.

4.4.1 Test Data and Experimental Settings

We use the product that is illustrated in Table 3.3 and SME setting in Figure 3.4 as a basis of our experiments. The points below describe the nature of data used in our initial experiments and how we expand our test setting using the basis data throughout the text while discussing the results.

- *Number of production periods:* We use three different values for the number of production periods: 2, 3, and 4.
- *Demand:* We randomly generate demands for each production period, which are uniformly distributed with lower bound 0, and upper bounds 20 and 50. Using 20 as the upper bound represents the low demand, and using 50 as the upper bound represents the high demand in our setting. We generate five different demand set for each upper bound for each planning horizon. Since demand is discrete, we round up each number to the nearest integer if the generated number is not integer. We chose 0 as a lower bound while generating demands because of having some instances with production periods with zero or very low demand in our test bed.
- *Capacity:* We set the total number of items that can be assembled in a production period as the maximum demand in a planning horizon divided by a utilization coefficient. We used 0.8 as utilization coefficient which is

the highest setting used by Haase and Kims (2000). We calculate capacities as follows:

$$CA = \left\lceil \frac{\max\{d_t\}}{0.8} \right\rceil$$

- We use the following values for other parameters: $\beta=3$ minutes, $\gamma=10$ minutes, $\theta=5$ minutes, $\tau=20$ minutes, $\varepsilon=2.083$ USD per minute (adapted from a real-life application), and $h=2$ USD per unit per production period (calculated by assuming that the total cost of a product is 1,000 USD and interest rate for one production period is 0.002) all of which are based on the real cases that we know from the practice.

We coded our formulations on GAMS version 24.1.2 and used Cplex version 12.5.1.0 solver with default options on a computer with 32 bit Windows 7 Premium Operating System, Intel Core 2 Duo 2.13-2.13 GHz CPU and a usable memory of 2.96 GB. Note that 32 bit Windows Operating System limits the maximum amount of memory that a running application can use at 1700 MB. Also maximum CPU time is set to 7200 seconds for these solutions.

We present our test instances in Table 4.1, where we also include the generated demand data. In the option column, the first number represents the number of production periods, and the second number represents the upper bound while generating demands. We generated five instances for each option. We also numbered instances so that even numbers refer to instances where the upper bound is 20 and odd numbers refer to instances where the upper bound is 50.

Table 4.1 Characteristics and demand data for test instances

		d_t						d_t			
Option	Instance	1	2	3	4	Option	Instance	1	2	3	4
2 x 50	1	30	37	-	-	2 x 20	2	9	15	-	-
	3	20	17	-	-		4	7	10	-	-
	5	17	1	-	-		6	5	15	-	-
	7	34	28	-	-		8	13	5	-	-
	9	33	28	-	-		10	11	20	-	-
3 x 50	11	2	4	28	-	3 x 20	12	15	14	15	-
	13	17	8	35	-		14	8	9	8	-
	15	8	3	39	-		16	13	3	2	-
	17	30	38	12	-		18	20	2	16	-
	19	21	26	29	-		20	17	15	18	-
4 x 50	21	22	29	3	11	4 x 20	22	13	3	2	16
	23	32	32	2	22		24	19	15	11	7
	25	13	23	23	23		26	11	20	17	10
	27	32	35	32	21		28	19	9	17	9
	29	29	33	3	8		30	12	15	7	1

4.4.2 Results

In this section, we present the results of the test runs for the lot sizing model. For each option, the average results of the mixed integer programming (MIP) solutions are provided. We use the same tools with the same configuration as in as given in Section 3.3.2 for finding solutions, and collect the same statistics for evaluation.

We present the detailed results in Appendix B and consider only average statistics in this section. Table 4.2 shows the average results with dynamic component assignment (DCA) for both uncapacitated and capacitated cases.

Table 4.2 Average results for lot sizing with DCA

Option	USL-DCA			CSL-DCA		
	CPU	IGap%	#opt	CPU	IGap%	#opt
2 x 20	5.6	34	50	6.3	37	50
2 x 50	820.2	40	50	1145.3	40	50
3 x 20	256.5	44	50	866.0	44	50
3 x 50	3,405.6	48	3[2](0)	3,842.9	48	3[2](0)
4 x 20	4,004.6	50	31	4,358.0	51	2[3](0)
4 x 50	6,390.4	50	0[4](1)	4,804.0	49	0[3](2)
OVERALL	2,480.5	44	21[7](2)	2,503.8	45	20[8](2)

As it can be observed from Table 4.2, the solution times increase both as the upper bound on demands and the number of production periods increase. In addition, solution times are greater in capacitated cases except for the option (4x50) in Table 4.2 where the number of non-optimal solutions with memory limit is higher in capacitated case.

IGap% increases with increasing of demand upper bound and the number of production periods except for case (4 x50). In Chapter 3, we have observed that IGap% is greater if the lot size is small compared to component pack capacities. Therefore, the decrease in IGap% for case (4 x50) in Table 4.2 can be explained by this observation.

If we compare the resulting total major set up and total holding costs for USL-DCA and CSL-DCA, on average, we see that there is a 10% of gap between two cases for the total major set up cost, and there is a 20% of gap between two cases for the total holding cost. This result implies that, more items are assembled in the uncapacitated case when a major set up is performed and part of these products held in inventory for future demands. Therefore, the total major set up cost is higher for the capacitated case compared to uncapacitated one. Note that this result depends on the value of holding and major set up cost parameters.

Table 4.3 shows the average results with stationary component assignment (SCA) for both uncapacitated and capacitated cases. Optimal solution of all instances is obtained with the time limit. Note that IGap% is calculated by

considering the linear relaxation solutions of the corresponding mathematical model with DCA.

Table 4.3 Average results for lot sizing with SCA

Option	USL-SCA			CSL-SCA		
	CPU	IGap%	#opt	CPU	IGap%	#opt
2 x 20	1.7	37	50	1.7	37	50
2 x 50	27.4	40	50	30.7	40	50
3 x 20	13.4	44	50	32.6	45	50
3 x 50	100.0	47	50	193.3	48	50
4 x 20	88.9	50	50	242.9	51	50
4 x 50	1,288.7	49	50	1,521.4	49	50
OVERALL	253.4	45	300	337.1	45	300

In terms of the solution times, SCA always gives faster solution times compared to DCA. This result occurs because SCA uses a unique component assignment scheme for each production period, and DCA tries to find a component assignment scheme for each assembly period.

From this time on, we only consider the uncapacitated cases in our remaining experiments. The average results for the special cases of SCA are given in Table 4.4. All instances are solved optimally in very short CPU times compared to USL-DCA and USL-SCA. USL-SCA1 and USL-SCA2 give the same solution for all instances. Within the special cases, USL-SCA1 gives the best CPU time and USL-SCA3 gives the longest CPU times on the average.

We have seen that by utilizing USL-SCA and special cases of SCA, one can find a solution in a very short time compared to USL-DCA. Now, USL-SCA and special cases of SCA are evaluated in terms of solution qualities compared to USL-DCA. For this purpose, we define the following statistics. Note that we exclude instances that we cannot find an optimal solution.

Table 4.4 Average results for special cases of SCA

Option	USL-SCA1			USL-SCA2			USL-SCA3		
	CPU	IGap%	#opt	CPU	IGap%	#opt	CPU	IGap%	#opt
2 x 20	0.1	37	50	0.2	37	50	0.5	39	50
2 x 50	0.5	40	50	1.6	40	50	2.5	43	50
3 x 20	0.5	44	50	0.6	44	50	1.8	46	50
3 x 50	1.9	47	50	3.5	47	50	4.4	50	50
4 x 20	2.4	50	50	3.7	50	50	4.4	52	50
4 x 50	6.5	49	50	13.4	49	50	66.0	51	50
OVERALL	2.0	45	300	3.8	45	300	13.3	47	300

- *%Dev*: Percentage deviation of solution of SCA(.) from the solution of USL-DCA (.=0, 1, 2, or 3). This statistic is calculated by utilizing the following equation.

$$\%Dev = \frac{\text{Solution of SCA(.)} - (\text{Solution of USL} - \text{DCA})}{(\text{Solution of USL} - \text{DCA})} \times 100$$

- *%Time*: Percentage of CPU time for finding a solution to SCA(.) compared to CPU time of USL-DCA. This statistic is calculated by utilizing the following equation.

$$\%Time = \frac{\text{CPU Time of SCA(.)}}{\text{CPU Time of USL} - \text{DCA}} \times 100$$

Table 4.5 shows the comparative results of different cases of SCA with the results of USL-DCA. Superscript numbers on each option refers to the number of instances that USL-DCA can find an optimal solution in time limit.

Table 4.5 Comparative results

SCA(.)	Option	%Dev			%Time		
		Min	Average	Max	Min	Average	Max
USL-SCA	2 x 20 ⁵	0.00	0.00	0.00	25.69	53.49	93.59
	2 x 50 ⁵	0.00	0.00	0.00	2.23	20.42	85.47
	3 x 20 ⁵	0.00	0.00	0.00	2.65	23.95	84.94
	3 x 50 ³	0.00	0.00	0.00	2.44	6.62	8.73
	4 x 20 ³	0.00	0.00	0.00	2.57	8.48	14.84
	4 x 50 ⁰	-	-	-	-	-	-
USL-SCA1	2 x 20 ⁵	0.00	0.00	0.00	1.26	8.78	20.09
	2 x 50 ⁵	0.00	0.00	0.00	0.03	1.62	6.41
	3 x 20 ⁵	0.00	0.00	0.00	0.15	4.50	20.19
	3 x 50 ³	0.00	0.00	0.00	0.04	0.16	0.26
	4 x 20 ³	0.00	0.00	0.00	0.08	0.33	0.50
	4 x 50 ⁰	-	-	-	-	-	-
USL-SCA2	2 x 20 ⁵	0.00	0.00	0.00	2.77	15.97	33.33
	2 x 50 ⁵	0.00	0.00	0.00	0.05	7.01	33.33
	3 x 20 ⁵	0.00	0.00	0.00	0.16	4.65	20.19
	3 x 50 ³	0.00	0.00	0.00	0.05	0.40	0.77
	4 x 20 ³	0.00	0.00	0.00	0.15	0.41	0.68
	4 x 50 ⁰	-	-	-	-	-	-
USL-SCA3	2 x 20 ⁵	3.66	3.85	3.95	4.75	17.75	26.92
	2 x 50 ⁵	4.06	4.41	4.66	0.15	11.65	53.42
	3 x 20 ⁵	3.69	3.99	4.23	0.15	7.37	30.13
	3 x 50 ³	3.86	4.17	4.43	0.24	0.66	1.15
	4 x 20 ³	3.86	3.96	4.07	0.13	0.38	0.57
	4 x 50 ⁰	-	-	-	-	-	-

Solutions of USL-SCA, USL-SCA1 and USL-SCA2 do not have any deviation compared to USL-DCA solution for all instances where an optimal solution is found for USL-DCA. As it can be seen from Table 4.2, USL-DCA cannot find an optimal solution for any instance for option (4x50). Therefore, it is not clear that USL-SCA, USIL-SCA1 and USIL-SCA2 would give the same solution with USL-DCA if USL-DCA could find optimal solutions for all instances. USL-SCA3 gives the worst objective function values compared to other solution methods for given problem setting.

When we consider %Time, it can be seen from the table that %Time decreases with increasing number of production periods and demands. This result comes from the fact that CPU time of USL-DCA increases faster than solution methods with SCA.

When we compare SCA(.) in terms of %Time, USL-SCA1 always gives the best CPU time since it does not include any component assignment decision. Remember that in USL-SCA2 and USL-SCA3, pockets that a component can be assigned to are only restricted, not predefined. Therefore, USL-SCA2 and USL-SCA3 still make decisions the component assignment. As it can be seen from Table 4.5, these restrictions decrease the problem size and simplify finding good solutions.

Table 4.6 shows the comparative results without excluding the instances where USL-DCA cannot find an optimal solution. Note that, we can find at least a feasible solution by using USL-DCA for all instances. In the table, %Dev* is calculated by using the lower bound found by the solver at last cut instead of optimal solution for the instances where USL-DCA cannot find an optimal solution in time limit. On the other hand, best solution found by USL-DCA is used while calculating %Dev**. %Time* is calculated by considering CPU time of all instances in an option.

As can be seen from Table 4.6, %Dev** is negative for some instances. This shows that, the solution of SCA(.) is better than the nonoptimal solution of USL-DCA for some instances.

Table 4.6 Comparative results without excluding nonoptimal solutions

SCA(.)	Option	%Dev*	%Dev**	%Time*
USILS-SCA	2 x 20	0.00	0.00	53.49
	2 x 50	0.00	0.00	20.42
	3 x 20	0.00	0.00	23.95
	3 x 50	2.61	-0.16	6.44
	4 x 20	1.30	0.00	5.85
	4 x 50	20.36	-0.87	22.36
USILS-SCA1	2 x 20	0.00	0.00	8.78
	2 x 50	0.00	0.00	1.62
	3 x 20	0.00	0.00	4.50
	3 x 50	2.61	-0.16	0.12
	4 x 20	1.30	0.00	0.21
	4 x 50	20.36	-0.87	0.09
USILS-SCA2	2 x 20	0.00	0.00	15.97
	2 x 50	0.00	0.00	7.01
	3 x 20	0.00	0.00	4.65
	3 x 50	2.61	-0.16	0.27
	4 x 20	1.30	0.00	0.27
	4 x 50	20.36	-0.87	0.19
USILS-SCA3	2 x 20	3.93	3.85	17.75
	2 x 50	4.41	4.41	11.65
	3 x 20	3.99	3.99	7.37
	3 x 50	6.91	4.02	0.42
	4 x 20	5.39	4.03	0.26
	4 x 50	25.48	3.35	0.92

Up to this point, we consider only the standard (naive) setting while evaluating the results. However, solution efforts and solution qualities of different solution methods highly depend on values of parameters like holding cost, utilisation cost of SME, component pack capacities, layout of SME, the total number of component type, body loading capacity, component pack installation time etc. In the rest of this section, we generate test instances, based on some alternative settings by changing some of parameters while keeping others as they are in the standard setting and evaluate the results. We consider five different problem settings as described below:

1. *2 x Starting Time*: In the standard problem setting, we use the standard starting time of an assembly period as 20 minutes. In this problem setting, we use the starting time as 40 minutes. In our problem formulation, starting time can be seen as a penalty for stopping the assembly operation. Therefore, with increasing starting time, our mathematical model will become more sensitive to the stopping need of assembly operation.
2. *(3 x 2) Layout*: Up to this point, we use the SME layout, which is illustrated in Figure 3.4 and component pick-and-place times as in Table 3.3. In this setting we use the SME layout as in Figure 3.7 and component pick-and-place times as in Table 3.9. In this layout, there are more options while assigning components to pockets than the standard problem setting.
3. *(4 x 2) Layout*: In this setting, we use the SME layout as in Figure 3.8 and component pick-and-place times as in Table 3.11.
4. *(15 – 100) Pack Capacity*: We set the pack capacity for a component, say 3, as just enough for assembling 15 units and set remaining component pack capacities as just enough to assemble 100 units. Note that we do not change the blank body (component 1) loading capacity.
5. *(5 – 100) Pack Capacity*: We set the pack capacity for a component, say 3, as just enough to assemble 5 units and set remaining component pack capacities as just enough to assemble 100 units. Again, we do not change the blank body (component 1) loading capacity. Note that, in the standard

setting, body loading capacity is 10. Therefore, 2 packs of component 3 is needed to assemble a full body load.

We only consider the average comparative results for different settings. Figure 4.2 and Figure 4.3 illustrate the average %Dev and %Time results respectively for each SCA(.) for each problem setting. In figures, the horizontal axis shows SCA(.) for different problem settings and the vertical axis shows the related statistics.

As it can be seen from Figure 4.2, USL-SCA always performs better in terms of %Dev compared to other solution methods since for all problem settings %Dev for USL-SCA is equal to zero. However, solution time of USL-SCA is always larger than others. When the first five settings are considered, USL-SCA, USL-SCA1, and USL-SCA give the same solution with USL-DCA for instances where we can find an optimal solution with USL-DCA. USL-SCA3 is the worst in terms of solution quality for all problem settings.

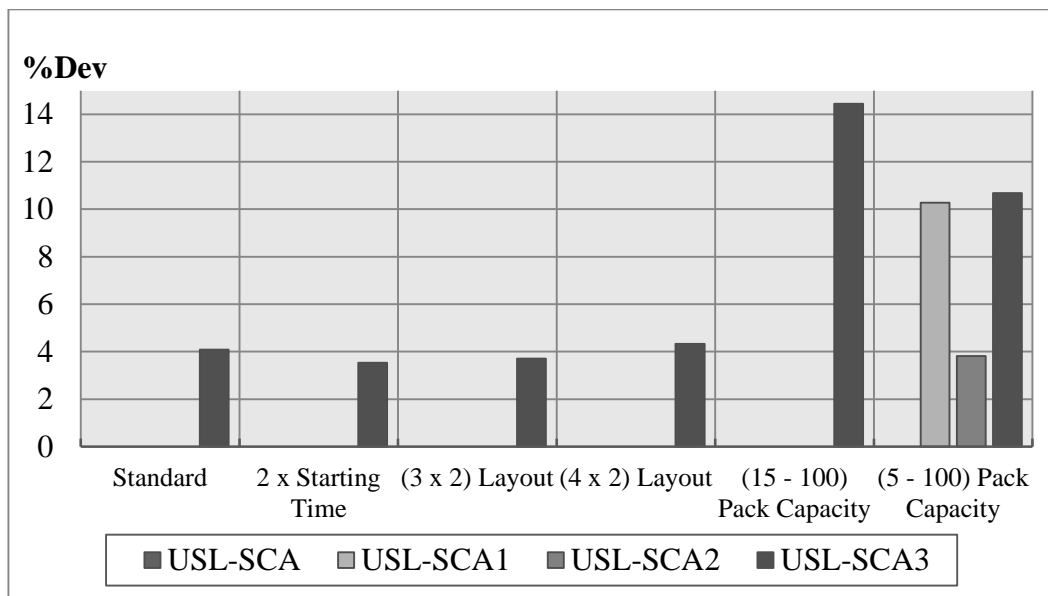


Figure 4.2 Average %Gap for solution methods for different settings

In terms of CPU time, USL-SCA1 always takes shorter time to find a solution. However, it does not give the best all the time. For instance, in problem setting with (5-100) pack capacities, assigning multiple packs of component 3 to different pockets is more efficient, and USL-SCA2 performs better in this problem setting.

Throughout our numerical experiments, we have seen that USL-DCA can be solved optimally for only small sized problems. We also have seen that USL-SCA gives solutions very close to USL-DCA and can be solved optimally for even larger problems than USL-DCA. However, USL-SCA might be limited in terms of finding an optimal solution in a reasonable sort time when a large real life problem is considered. On the other hand, by utilizing special cases of SCA, one can also obtain good solutions for the larger problems.

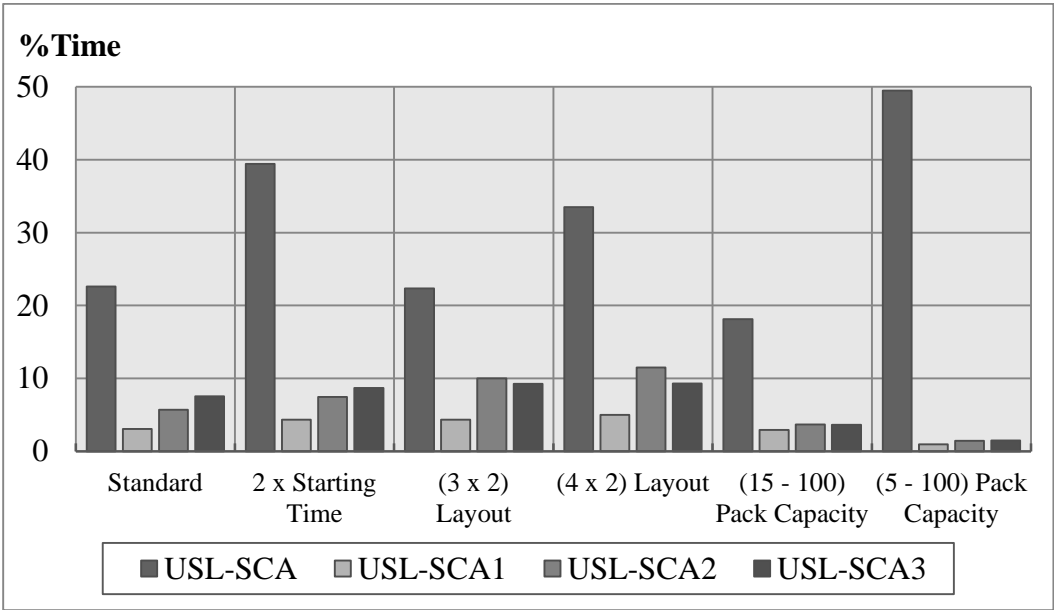


Figure 4.3 Average %Time for solution methods for different settings

4.5 Case Study

In this section, we solve a real sized problem by considering the product given in Section 3.4. Assume that a production period is one week and the planning horizon is three months. Therefore, there are 12 production periods in a planning horizon. Assume also that demands are distributed uniformly with lower bound 0 and upper bound 50.

Table 4.7 shows the policies that we use while solving the problem in our case study. Lot-for-lot production refers to the planning policy that demand of each production period is satisfied by the production in corresponding production period only. Therefore, there is no inventory carried in lot-for-lot production policy. For policies 1 to 6, lot sizing and component assignment problems are solved hierarchically. First, we solve the lot sizing problem and find the lot sizes for each production periods. Then we solve the component assignment problem with these lot sizes. Finally, we calculate the actual total cost after solving the component assignment problem. For policies 2, 4, and 6, we use DCA-SI policy to solve the component assignment problem with given lot sizes. If we cannot find an optimal solution, we compare the best feasible solution of DCA-SI policy with optimal solution of SCA-SI, and choose the best alternative.

While solving the classical SLP, we use the classical MIP model. We use the unit assembly times of given SCA while solving the lot sizing problem. For the set up time, we use two different set up times, underestimated or overestimated set up times. We select 20 minutes major set up time in our standard problem setting as underestimated set up time since it does not include any starting, component pack installation, and body loading times. For overestimated case, we use 200 minutes, which is 10 times of the underestimated set up time. We use other parameters as in our standard problem setting for solving the lot sizing problem.

Table 4.7 Different policies for the case study

	Lot Sizing Problem	Component Assignment Problem
Policy 1	Lot-for-lot production policy	Given SCA
Policy 2	Lot-for-lot production policy	DCA-SI and SCA-SI
Policy 3	Classical SLP with underestimated set up time	Given SCA
Policy 4	Classical SLP with underestimated set up time	DCA-SI and SCA-SI
Policy 5	Classical SLP wit overestimated set up time	Given SCA
Policy 6	Classical SLP wit overestimated set up time	DCA-SI and SCA-SI
Policy 7	Integrated solution	

In policy 7, we utilize our integrated solution methods (e.g. USL-DCA, USL-SCA, USL-SCA1, USL-SCA2 and USL-SCA3). Recall that we use other parameters as they are in our standard problem setting while solving the problem. Note that we can only find an optimal solution after 4 hours of CPU time with USL-SCA1 and other solution methods cannot give any solution because of the memory limit.

Table 4.8 shows the result of our case study. We illustrate lot sizes, total costs, absolute deviations, and %Dev for each policy in the table. Note that demand data can be observed from the lot size of policy 1 and policy 2.

At the bottom of the table, we provide three measures about the results. First, we present the total cost of each policy. For policy 7, the total cost is the optimal objective function value of USL-SCA1.

Absolute deviations are calculated by subtracting the total cost of policy 7 from the total cost of the corresponding policy. %Dev is the percentage deviation between the total cost of the corresponding policy from the total cost of policy 7 and calculated by utilizing the following equation for each policy.

$$\%Dev = \frac{\text{The total cost of the policy} - \text{The total cost of policy 7}}{\text{The total cost of policy 7}} \times 100$$

Table 4.8 Results for the case study

Production Period		Policy 1	Policy 2	Policy 3	Policy 4	Policy 5	Policy 6	Policy 7
1	Lot Size	36		46		129		50
2		10		0		0		0
3		47		47		0		80
4		36		36		0		0
5		37		37		101		80
6		34		44		0		0
7		8		0		0		0
8		2		0		0		0
9		20		20		0		20
10		48		48		85		85
11		36		37		0		0
12		1		0		0		0
Total Cost (USD)		4503.37	3956.84	4083.00	3480.26	4369.70	3785.70	3378.24
Abs. Dev. (USD)		1125.13	578.60	704.76	102.02	991.46	407.46	-
%Dev		33.31	17.13	20.86	3.02	29.35	12.06	-

From Table 4.8, we observe that the integrated solution approach gives the best solution for the problem. In addition, our component assignment solution approach with given lot size (policies 2, 4, and 6) also performs better than the given SCA (policies 1, 3, and 5).

In terms of the lot sizing problem, lot-for-lot production gives the worst result, which reveals the need of utilizing a lot sizing policy where set up and holding costs are balanced efficiently. Even if an efficient lot sizing policy is utilized, it is very hard to estimate the set up time without scheduling component pack installation and body loading activities. It can be observed from the table that both underestimating and overestimating the set up times results in inefficiencies

in the operations of an SME. Therefore, an integrated approach is better to solve both the lot sizing and the component assignment problem.

CHAPTER 5

CONCLUSIONS AND DIRECTIONS FOR FUTURE STUDIES

In this study, we have formulated the single-item lot sizing problem (SLP) in surface mount technology (SMT) production lines in which we also integrated the component assignment problem and proposed a mathematical model for solving the problem. During the formulation stage, we first focused on the component assignment problem where we tried to eliminate the complexity of the problem as much as possible while sustaining key features of the problem, and presented a mathematical model for the component assignment problem for given lot sizes for both dynamic and stationary cases. Then, we presented our integrated SLP model, which is a modification of the facility location based formulation of the SLP. Numerical experiments were conducted for both capacitated and uncapacitated cases with different special cases of the component assignment problem.

In our computational experiments, we tested the performance of our integrated formulation using a set of instances developed after preliminary experiments. We first checked the effects of lot size and product specific characteristics on our component assignment formulation. Results have shown that increasing lot sizes and decreasing pack capacities increase the solution times. Then, we test our SLP model, and our results have shown that, increasing demand and increasing number of production periods increase the solution times. In addition, we also have seen that solution times in capacitated instances are greater than the solution times in uncapacitated instances, as expected.

Considering the need to develop viable approximations for our formulation in order to tackle larger problems, we provided special cases of the component assignment problem where the total number of different component assignment schemes that can be used in a planning horizon is restricted. We tested these cases under different settings and our results have shown that these restrictions yielded

promising results with low computation times and good upper bounds that are very close to optimal solutions. This shows that, especially for problems with many production periods and high demands, these approximations are potentially sound options against vain efforts for finding optimal solutions.

In our opinion, the fundamental contribution of this thesis is establishing a connection between the lot sizing and component assignment problems for SMT production lines. This connection not only provides an extension of standard formulation of the SLP in the field of lot sizing, but also introduces the lot sizing problem in the field of SMT production line. Moreover, to our knowledge, we provide the first component assignment formulation based on the lot size for an SMT product in which component pack and blank body loading capacities are also incorporated.

Our experiments were limited in terms of SME layout and product specification, and confined to considerably small instances. As a future study, potentials of mathematical models and special cases should be tested for different SME layouts and product specifications for larger instances.

For further studies regarding the SLP in SMT production lines, extensions with overtime, backordering, set up carryover, and nonzero initial and ending inventory should be considered. In addition, the standard and shortest path formulations of the SLP should be extended by integrating the component assignment problem. Moreover, approximation methods used for solving the classical SLP formulations should be studied in the context of our problem formulation for SMT production lines.

Regarding the production planning problems in SMT production lines, studying the lot sizing problem in multiple item case is a promising research area in high-mix low-volume production environments. For multiple item case in SMT production lines, changeover times depend on the commonality of components for different products. Therefore, changeover times are sequence dependent, and the sequencing problem of different products in an SMT production line is one of the intensively studied problems in the field of SMT production line. Rajkumar and Narendran (1998), Carmon et al. (1989), Davis and Selep (1990), Barnea and

Sipper (1993), and Maimon et al. (1993) are some examples of such studies on sequencing problems in SMT production lines. However, to our knowledge, the lot sizing and scheduling problem with sequence dependent changeover times has never been studied in SMT production lines.

REFERENCES

- Aggarwal, A., & Park, J. (1993). *Improved Algorithms for Ecoeconomic Lot-size Problems*. *Operations Research*, 41, 549-571.
- Altinkemer, K., Kazaz, B., Koksalan, M., & Moskowitz, H. (2000). *Optimization of Printed Circuit Board Manufacturing: Integrated Modeling and Algorithms*. *European Journal of Operational Research*, 124, 409–421.
- Ayob, M., & Kendall, G. (2005). *A Triple Objective Function with a Chebychev Dynamic Pick-and-Place Point Specification Approach to Optimise the Surface Mount Placement Machine*. *European Journal of Operational Research*, 164, 609–626.
- Ayob, M., & Kendall, G. (2009). *The Optimisation of the Single Surface Mount Device Placement Machine in Printed Circuit Board Assembly: A Survey*. *International Journal of Systems Science*, 40, 553–569.
- Ayob, M., Cowling, P., & Kendall, G. (2002). *Optimisation for Surface Mount Placement Machines*. *Industrial Technology*, 1, 498-503.
- Baker, K., Dixon, P., Magazine, M., & Silver, E. (1978). *An Algorithm for the Dynamic Lot-Size Problem with Time Varying Production Capacity Constraints*. *Management Science*, 24, 1710–1720.
- Ball, M. O., & Magazine, M. J. (1988). *Sequencing of Insertions in Printed Circuit Board Assembly*. *Operational Research*, 36, 192–201.
- Bard, J., Clayton, R., & and Feo, T. (1994). *Machine Setup and Component Placement in Printed Circuit Board Assembly*. *International Journal of Flexible Manufacturing Systems*, 6, 5–31.
- Barnea, A., & Sipper, D. (1993). *Set-Up Seduction in PCB Automated Assembly*. *Computer Integrated Manufacturing Systems*, 6, 18-26.

- Bitran, G., & Yanasse, H. (1982). *Computational Complexity of the Capacitated Lot Size Problem*. *Management Science*, 28, 1174–1186.
- Bonert, M., Shu, L., & Benhabib, B. (2000). *Motion Planning for Multi-Robot Assembly Systems*. *International Journal of Computer Integrated Manufacturing*, 13, 301-310.
- Brahimi, N., Dauzere-Peres, S., Najid, N., & Nordli, A. (2006). *Single-item Lot Sizing Problems*. *European Journal of Operational Research*, 168, 1–16.
- Carmon, T., Maimon, O., & Dar-El, E. (1989). *Group Set-Up for Printed Circuit Board Assembly*. *International Journal of Production Research*, 27 , 1795-1820.
- Chan, D., & Mercier, D. (1989). *IC Insertion: An Application of the Traveling Salesman Problem*. *International Journal of Production Research*, 27, 1837–1841.
- Chang, T., & Terwilliger, J. (1987). *Rule-Based System for Printed Wiring Assembly Process Planning*. *International Journal of Production Research*, 25, 1465–1482.
- Chen, H., Hearn, D., & Lee, C. (1994). *A New Dynamic Programming Algorithm for the Aingle Item Capacitated Dynamic Lot Size Model*. *Journal of Global Optimization*, 4, 285–300.
- Chung, C., Flynn, J., & Lin, C. (1994). *An Effective Algorithm for the Capacitated Single-item Lot Size Problem*. *European Journal of Operational Research*, 75, 427–440.
- Crama, Y., Flippo, O., Klundert, J., & Spieksma, F. (1996). *The Component Retrieval Problem in Printed Circuit Board Assembly*. *International Journal of Flexible Manufacturing Systems*, 8, 287–312.

- Crama, Y., Klundert, J., & Spieksma, F. (2002). *Production Planning Problems in Printed Circuit Board Assembly*. *Discrete Applied Mathematics*, 123, 339–361.
- Davis, T., & Selep, E. (1990). *Group Technology for High-Mixed Printed Circuit Assembly*. *Electronic Manufacturing Technology Symposium*, 264-269.
- Duman, E., & Or, I. (2004). *Precedence Constrained TSP Arising in Printed Circuit Board Assembly*. *International Journal of Production Research*, 42, 67–78.
- Evans, J. (1985). *An Efficient Implementation of the Wagner–Whitin Algorithm for Dynamic Lot-Sizing*. *Journal of Operations Management*, 5, 229–235.
- Federgruen, A., & Tzur, M. (1991). *A Simple Forward Algorithm to Solve General Dynamic Lot-Sizing Models with n Periods in $O(n \log n)$ or $O(n)$ Time*. *Management Science*, 37, 909–925.
- Florian, M., Lenstra, J., & Rinnooy Kan, A. (1980). *Deterministic Production Planning: Algorithms and Complexity*. *Management Science*, 26, 669–679.
- Francis, R., Hamacher, H., Lee, C.-Y., & Yeralan, S. (1994). *Finding Placement Sequences and Bin Locations for Cartesian Robots*. *IIE Transactions*, 26, 47–59.
- Friedman, Y., & Hoch, Y. (1978). *A Dynamic Lot-Size Model with Inventory Deterioration*. *INFOR*, 16, 183–188.
- Grotzinger, S. (1992). *Feeder Assignment Models for Concurrent Placement Machines*. *IIE Transactions*, 24, 31-46.
- Grunow, M., Günther, H., Schleusener, M., & Yilmaz, I. (2004). *Operations Planning for Collect-and-Place Machines in PCB Assembly*. *Computers and Industrial Engineering*, 47, 409–429.

- Haase, K., & Kimms, A. (2000). *Lot Sizing and Scheduling with Sequence-Dependent Setup Costs and Times and Efficient Rescheduling Opportunities*. *International Journal of Production Economics*, 66, 159-169.
- Ho, W., & Ji, P. (2006). *A Genetic Algorithm Approach to Optimise Component Placement and Retrieval Sequence for Chip Shooter Machines*. *Journal of Advanced Manufacturing Technology*, 28, 556-560.
- Hsu, V. (2000). *Dynamic Economic Lot Size Model with Perishable Inventory*. *Management Science*, 46, 1159-1169.
- Jeevan, K., Parthiban, A., Seetharamu, K. A., & Quadir, G. (2002). *Optimization of PCB Component Placement Using Genetic Algorithms*. *Electronics Manufacturing*, 11, 69-79.
- Karmarkar, U., & Schrage, L. (1985). *The Deterministic Dynamic Product Cycling Problem*. *Operations Research*, 33, 326-345.
- Khoo, L., & Ong, N. (1998). *PCB Assembly Planning Using Genetic Algorithms*. *International Journal of Advanced Manufacturing Technology*, 14, 363-368.
- Klomp, C., Klundert, J., Spieksma, F., & Voogt, S. (2000). *The Feeder Rack Assignment Problem in PCB Assembly: A Case Study*. *International Journal of Production Economics*, 64, 399-407.
- Krarpup, J., & Bilde, O. (1977). *Plant Location, Set Covering and Economic Lot Size: An $O(mn)$ Algorithm for Structured Problems*. *International Series of Numerical Mathematics*, 36, 155-186.
- Kumar, R., & Luo, Z. (2003). *Optimizing the Operation Sequence of a Chip Placement Machine Using TSP Model*. *IEEE Transactions*, 26, 14-21.

- Leipälä, T., & Nevalainen, O. (1989). *Optimization of the Movements of a Component Placement Machine*. *European Journal of Operational Research*, 38, 167-177.
- Leu, M., Wong, H., & Ji, Z. (1993). *Planning of Component Placement/Insertion Sequence and Feeder Setup in PCB Assembly Using Genetic Algorithm*. *Journal of Electronic Packaging*, 115, 424-432.
- Lotfi, V., & Yoon, Y.-S. (1994). *An Algorithm for the Single-Item Capacitated Lot-Sizing Problem with Concave Production and Holding Costs*. *Journal of Operational Research Society*, 45, 934-941.
- Magyar, G., Johnsson, M., & Nevalainen, O. (1999). *On Solving Single Machine Optimization Problems in Electronics Assembly*. *Journal of Electronics Manufacturing*, 9, 249-267.
- Maimon, O., Dar-El, E., & Carmon, T. F. (1993). *Set-Up Saving Schemes for Printed Circuit Boards Assembly*. *European Journal of Operations Research*, 70, 177-190.
- Moyer, L., & Gupta, S. (1996). *SMT Feeder Slot Assignment for Predetermined Component Placement Paths*. *Journal of Electronics Manufacturing*, 6, 173-192.
- Nahmias, S. (1982). *Perishable Inventory Theory: A Review*. *Operations Research*, 30, 680-708.
- Rajagopalan, S. (1992). *Deterministic Capacity Expansion under Deterioration*. *Management Science*, 38, 525-539.
- Rajkumar, K., & Narendran, T. T. (1998). *A Heuristic for Sequencing PCB Assembly to Minimize Set-Up Times*. *Production Planning and Control: The Management of Operations*, 9, 465-476.

- Sambasivam, M., & Schmidt, C. (2000). *A Solution Procedure to Solve Uncapacitated Lot Sizing for Multi-Plant, Multi-Period Problems with Inter-Plant Transfers*. University of Putra Malaysia, Faculty of Economics and Management.
- Shaw, D., & Wagelmans, A. (1998). *An Algorithm for Single-item Capacitated Economic Lot Sizing with Piecewise Linear Production Costs and General Holding Costs*. *Management Science*, 44, 831–838.
- Shih, W., Srihari, K., & and Adriance, J. (1996). *Expert System Based Placement Sequence Identification for Surface Mount PCB Assembly*. *International Journal of Advanced Manufacturing Technology*, 11, 413–424.
- Sohn, J., & Park, S. (1996). *Efficient Operation of a Surface Mounting Machine with a Multihead Turret*. *International Journal of Production Research*, 34, 1131-1143.
- Su, Y., Wang, C., Egbelu, P., & Cannon, D. (1995). *A Dynamic Point Specification Approach to Sequencing Robot Moves for PCB Assembly*. *International Journal of Computer Integrated Manufacturing*, 8, 448-456.
- Sun, D., Lee, T., & Kim, K. (2005). *Component Allocation and Feeder Arrangement for a Dual-gantry Multi-head Surface Mounting Placement Tool*. *International Journal of Production Economics*, 95(2), 245–264.
- Sun, D., Tae-Eog, L., & Kyung-Hoon, K. (2005). *Component Allocation and Feeder Arrangement for a Dual-Gantry Multi-Head Surface Mounting Placement Tool*. *International Journal of Production Economics*, 95, 245–264.
- Tirpak, T., Nelson, P., & Aswani, A. (2000). *Optimization of Revolver Head SMT Machines Using Adaptive Simulated Annealing (ASA)*. *Electronics Manufacturing Technology Symposium*, 214–220.

- Van Hoesel, S., Kuik, R., Salomon, M., & Van Wassenhove, L. (1994). *The Single-item Discrete Lot-Sizing and Scheduling Problem: Optimization by Linear and Dynamic Programming*. *Discrete Mathematics*, 48, 289–303.
- Wagelmans, A., Hoesel, S., & Kolen, A. (1992). *Economic Lot Sizing: An $O(n \log n)$ that Runs in Linear Time in the Wagner–Whitin Case*. *Operations Research*, 40, S145–S156.
- Wagner, H., & Whitin, T. (1958). *Dynamic Version of the Economic Lot Size Model*. *Management Science*, 5, 89–96.
- Wang, W., & Nelson, P. T. (1999). *Optimization of High-Speed Multistation SMT Placement Machines Using Evolutionary Algorithms*. *IEEE Transaction on Electronics Packaging Manufacturing*, 22, 137-146.
- Yeo, S., Low, C., & Yong, K. (1996). *A Rule-Based Frame System for Concurrent Assembly Machines*. *International Journal of Advanced Manufacturing Technology*, 12, 370–376.
- Zangwill, W. (1969). *A Backlogging Model and a Multi-Echelon Model of a Dynamic Economic Lot Size Production System: A Network Approach*. *Management Science*, 15, 506–527.

APPENDIX A

EXPERIMENTAL RESULTS FOR DCA-SI AND SCA-SI

Table A.1 Results for option (6 x c_i) with DCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.09	10	5	12	27	3.07	30.07
3	1	Opt.	0.08	10	5	12	27	9.2	36.2
6	1	Opt.	0.09	10	5	12	27	18.4	45.4
9	2	Opt.	0.13	20	10	12	42	27.6	69.6
12	2	Opt.	0.08	20	10	12	42	36.8	78.8
15	3	Opt.	0.17	30	15	12	57	46	103
18	3	Opt.	0.19	30	15	12	57	55.2	112.2
21	4	Opt.	0.34	40	20	12	72	64.4	136.4
24	5	Opt.	1.08	40	20	15	75	73.6	148.6
27	6	Opt.	4.07	50	25	15	90	82.8	172.8
30	6	Opt.	1.28	50	25	15	90	92	182
33	8	Opt.	13.95	60	30	18	108	101.2	209.2
36	8	Opt.	8.32	60	30	18	108	110.4	218.4
39	9	Opt.	31.29	70	35	18	123	119.6	242.6
42	9	Opt.	31.92	70	35	21	126	128.8	254.8
45	11	Opt.	163.27	80	40	21	141	138	279
48	12	Opt.	272.89	80	40	24	144	147.2	291.2
51	13	Opt.	961.47	90	45	24	159	156.4	315.4
54	13	Opt.	182.68	90	45	24	159	165.6	324.6
57	14	Opt.	2,698.93	10	50	24	84	174.13	258.13
60	14	Opt.	2,126.79	100	50	24	174	186.9	360.9

Table A.2 Results for option ($6 \times c_i$) with SCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.14	10	5	12	27	3.07	30.07
3	1	Opt.	0.05	10	5	12	27	9.2	36.2
6	1	Opt.	0.03	10	5	12	27	18.4	45.4
9	2	Opt.	0.27	20	10	12	42	27.6	69.6
12	2	Opt.	0.08	20	10	12	42	36.8	78.8
15	3	Opt.	0.23	30	15	12	57	46	103
18	3	Opt.	0.09	30	15	12	57	55.2	112.2
21	4	Opt.	0.38	40	20	12	72	64.4	136.4
24	5	Opt.	0.55	40	20	15	75	73.6	148.6
27	6	Opt.	2.03	50	25	15	90	82.8	172.8
30	6	Opt.	0.58	50	25	15	90	92	182
33	8	Opt.	6.83	60	30	18	108	101.2	209.2
36	8	Opt.	1.17	60	30	18	108	110.4	218.4
39	9	Opt.	4.1	70	35	18	123	119.6	242.6
42	9	Opt.	2.98	70	35	21	126	128.8	254.8
45	11	Opt.	5.3	80	40	210	330	138	468
48	12	Opt.	7.36	80	40	24	144	147.2	291.2
51	13	Opt.	12.67	90	45	24	159	156.4	315.4
54	13	Opt.	5.54	90	45	24	159	165.6	324.6
57	14	Opt.	12.95	100	50	24	174	174.8	348.8
60	14	Opt.	9.61	100	50	27	177	184	361

Table A.3 Results for option (8 x c_i) with DCA-SI (Times are in minutes except for solution time)

<i>Q</i>	<i>K</i>	Optimality *	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.06	10	5	12	27	3.07	30.07
4	1	Opt.	0.11	10	5	12	27	12.27	39.27
8	1	Opt.	0.12	10	5	12	27	24.53	51.53
12	2	Opt.	0.14	20	10	12	42	36.8	78.8
16	2	Opt.	0.05	20	10	12	42	49.07	91.07
20	3	Opt.	0.3	30	15	12	57	61.33	118.33
24	4	Opt.	0.5	30	15	15	60	73.6	133.6
28	5	Opt.	3.57	40	20	15	75	85.87	160.87
32	5	Opt.	0.83	40	20	15	75	98.13	173.13
36	7	Opt.	8.27	50	25	18	93	110.4	203.4
40	7	Opt.	13.34	50	25	21	96	122.67	218.67
44	9	Opt.	93.68	60	30	21	111	134.93	245.93
48	10	Opt.	62.87	60	30	24	114	147.2	261.2
52	11	Opt.	307.92	70	35	24	129	159.47	288.47
56	11	Opt.	192.72	70	35	24	129	174.13	303.13
60	12	Opt.	217.01	80	40	24	144	184	328
64	12	Opt.	345.56	80	40	27	147	196.27	343.27
68	14	Opt.	2,692.05	90	45	27	162	208.53	370.53
72	15	Opt.	1,777.18	90	45	30	165	221.6	386.6
76	16	Opt.	3,605.82	100	50	30	180	233.07	413.07
80	16	-	7,200.00	100	50	33	183	245.33	428.33

*Note that non optimal solutions are represented by “-“.

Table A.4 Results for option ($8 \times c_i$) with SCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.05	10	5	12	27	3.07	30.07
4	1	Opt.	0.06	10	5	12	27	12.27	39.27
8	1	Opt.	0.09	10	5	12	27	24.53	51.53
12	2	Opt.	0.2	20	10	12	42	36.8	78.8
16	2	Opt.	0.05	20	10	12	42	49.07	91.07
20	3	Opt.	0.23	30	15	12	57	61.33	118.33
24	4	Opt.	0.28	30	15	15	60	73.6	133.6
28	5	Opt.	1.03	40	20	15	75	85.87	160.87
32	5	Opt.	0.58	40	20	15	75	98.13	173.13
36	7	Opt.	1.93	50	25	18	93	110.4	203.4
40	7	Opt.	2.86	50	25	21	96	122.67	218.67
44	9	Opt.	8.07	60	30	21	111	134.93	245.93
48	10	Opt.	3.54	60	30	24	114	147.2	261.2
52	11	Opt.	13.78	70	35	24	129	159.47	288.47
56	11	Opt.	11.81	70	35	27	132	171.73	303.73
60	12	Opt.	9.27	80	40	24	144	184	328
64	12	Opt.	5.87	80	40	27	147	196.27	343.27
68	14	Opt.	16.4	90	45	27	162	208.53	370.53
72	15	Opt.	17.04	90	45	30	165	221.6	386.6
76	16	Opt.	24.71	100	50	30	180	233.07	413.07
80	16	Opt.	26.52	100	50	33	183	245.33	428.33

Table A.5 Results for option $(10 \times \left\lceil \frac{c_i}{2} \right\rceil)$ with DCA-SI (Times are in minutes except for solution time)

Q	K	Optimality *	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.12	10	5	12	27	3.07	30.07
5	1	Opt.	0.08	10	5	12	27	15.33	42.33
10	1	Opt.	0.09	10	5	12	27	30.67	57.67
15	3	Opt.	0.66	20	10	15	45	46	91
20	4	Opt.	1	20	10	18	48	61.33	109.33
25	7	Opt.	9.78	30	15	24	69	76.67	145.67
30	7	Opt.	5.93	30	15	24	69	93	162
35	9	Opt.	44.2	40	20	30	90	107.33	197.33
40	10	Opt.	16.3	40	20	30	90	124	214
45	12	Opt.	549.67	50	25	36	111	138.97	249.97
50	14	Opt.	182.9	50	25	39	114	154.33	268.33
55	15	Opt.	1,602.01	60	30	42	132	169.97	301.97
60	16	Opt.	102.87	60	30	42	132	186	318
65	19	-	7,200.00	70	35	51	156	200.3	356.3
70	19	Opt.	3,812.01	70	35	51	156	216.67	372.67
75	21	-	920.98	80	40	54	174	231.97	405.97
80	21	Opt.	1,136.37	80	40	54	174	248	422
85	24	-	7,200.00	90	45	63	198	262.63	460.63
90	25	-	1,047.50	90	45	63	198	278.67	476.67
95	27	-	1,298.55	100.02	50	66	216.02	293.97	509.98
100	28	-	7,200.00	100	50	69	219	309.33	528.33

* Note that non-optimal solutions with solution times less than 7200 are integer solutions when memory limit is reached.

Table A.6 Results for option ($10 \times \left\lceil \frac{c_i}{2} \right\rceil$) with SCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.25	10	5	12	27	3.07	30.07
5	1	Opt.	0.08	10	5	12	27	15.33	42.33
10	1	Opt.	0.08	10	5	12	27	30.67	57.67
15	3	Opt.	0.45	20	10	15	45	46	91
20	4	Opt.	0.5	20	10	18	48	61.33	109.33
25	7	Opt.	6.68	30	15	24	69	76.67	145.67
30	7	Opt.	1.15	30	15	24	69	93	162
35	9	Opt.	10.25	40	20	30	90	107.33	197.33
40	10	Opt.	2.81	40	20	30	90	124	214
45	12	Opt.	27.66	50	25	36	111	139.5	250.5
50	14	Opt.	15.48	50	25	39	114	155	269
55	15	Opt.	22.56	60	30	42	132	170.2	302.2
60	16	Opt.	25.04	60	30	42	132	186	318
65	19	Opt.	70.78	70	35	51	156	200.87	356.87
70	19	Opt.	37.77	70	35	51	156	217	373
75	21	Opt.	29.58	80	40	54	174	232.43	406.43
80	21	Opt.	15.63	80	40	54	174	248	422
85	24	Opt.	148	90	45	63	198	263.1	461.1
90	25	Opt.	86.3	90	45	63	198	279	477
95	27	Opt.	41.29	100	50	66	216	294.5	510.5
100	28	Opt.	141.88	100	50	69	219	310	529

Table A.7 Results for option (10 x c_i) with DCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.05	10	5	12	27	3.07	30.07
5	1	Opt.	0.05	10	5	12	27	15.33	42.33
10	1	Opt.	0.09	10	5	12	27	30.67	57.67
15	2	Opt.	0.09	20	10	12	42	46	88
20	2	Opt.	0.05	20	10	12	42	61.33	103.33
25	4	Opt.	2.9	30	15	15	60	76.67	136.67
30	4	Opt.	0.5	30	15	15	60	92	152
35	6	Opt.	6.02	40	20	18	78	107.33	185.33
40	6	Opt.	3.28	40	20	18	78	122.67	200.67
45	8	Opt.	37.47	50	25	21	96	139.83	235.83
50	9	Opt.	22.25	50	25	24	99	153.33	252.33
55	10	Opt.	127.36	60	30	24	114	168.67	282.67
60	10	Opt.	32.59	60	30	24	114	184	298
65	12	Opt.	1,010.79	70	35	27	132	201.83	333.83
70	12	Opt.	324.17	70	35	30	135	214.67	349.67
75	14	Opt.	1,726.10	80	40	30	150	230	380
80	14	Opt.	572.06	80	40	30	150	245.33	395.33
85	15	Opt.	3,683.93	90	45	33	168	260.67	428.67
90	16	Opt.	3,784.94	90	45	36	171	276	447
95	18	-	7,200.00	100	50	36	186	294	480
100	19	-	3,606.34	100	50	39	189	306.67	495.67

Table A.8 Results for option (10 x c_i) with SCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.05	10	5	12	27	3.07	30.07
5	1	Opt.	0.05	10	5	12	27	15.33	42.33
10	1	Opt.	0.03	10	5	12	27	30.67	57.67
15	2	Opt.	0.09	20	10	12	42	46	88
20	2	Opt.	0.11	20	10	12	42	61.33	103.33
25	4	Opt.	0.52	30	15	15	60	76.67	136.67
30	4	Opt.	0.36	30	15	15	60	92	152
35	6	Opt.	1.61	40	20	18	78	107.33	185.33
40	6	Opt.	0.58	40	20	18	78	122.67	200.67
45	8	Opt.	10.64	50	25	24	99	138	237
50	9	Opt.	2.81	50	25	24	99	153.33	252.33
55	10	Opt.	6.05	60	30	24	114	168.67	282.67
60	10	Opt.	1.25	60	30	24	114	184	298
65	12	Opt.	18.35	70	35	30	135	199.33	334.33
70	12	Opt.	6.96	70	35	30	135	214.67	349.67
75	14	Opt.	13.43	80	40	30	150	230	380
80	14	Opt.	2.95	80	40	30	150	245.33	395.33
85	15	Opt.	12.39	90	45	33	168	260.67	428.67
90	16	Opt.	3.74	90	45	36	171	276	447
95	18	Opt.	128.95	100	50	39	189	291.33	480.33
100	19	Opt.	19.84	100	50	39	189	306.67	495.67

Table A.9 Results for option (10 x 2c_i) with DCA-SI (Times are in minutes except for solution time)

<i>Q</i>	<i>K</i>	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.08	10	5	12	27	3.07	30.07
5	1	Opt.	0.06	10	5	12	27	15.33	42.33
10	1	Opt.	0.06	10	5	12	27	30.67	57.67
15	2	Opt.	0.22	20	10	12	42	46	88
20	2	Opt.	0.13	20	10	12	42	61.33	103.33
25	3	Opt.	0.2	30	15	12	57	76.67	133.67
30	3	Opt.	0.2	30	15	12	57	92	149
35	4	Opt.	0.33	40	20	12	72	107.33	179.33
40	4	Opt.	0.23	40	20	12	72	122.67	194.67
45	5	Opt.	0.5	50	25	12	87	138	225
50	6	Opt.	2.14	50	25	15	90	153.33	243.33
55	7	Opt.	9.3	60	30	15	105	168.67	273.67
60	7	Opt.	4.96	60	30	15	105	184	289
65	9	Opt.	59.08	70	35	18	123	199.33	322.33
70	9	Opt.	31.84	70	35	18	123	214.67	337.67
75	10	Opt.	115.35	80	40	18	138	230	368
80	10	Opt.	38.5	80	40	18	138	245.33	383.33
85	11	Opt.	155.8	90	45	18	153	260.67	413.67
90	12	Opt.	589.87	90	45	24	159	276	435
95	14	Opt.	3,535.23	100	50	24	174	291.33	465.33
100	14	Opt.	1,297.60	100	50	24	174	306.67	480.67

Table A.10 Results for option (10 x 2c_i) with SCA-SI (Times are in minutes except for solution time)

<i>Q</i>	<i>K</i>	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.13	10	5	12	27	3.07	30.07
5	1	Opt.	0.06	10	5	12	27	15.33	42.33
10	1	Opt.	0.11	10	5	12	27	30.67	57.67
15	2	Opt.	0.23	20	10	12	42	46	88
20	2	Opt.	0.08	20	10	12	42	61.33	103.33
25	3	Opt.	0.22	30	15	12	57	76.67	133.67
30	3	Opt.	0.16	30	15	12	57	92	149
35	4	Opt.	0.28	40	20	12	72	107.33	179.33
40	4	Opt.	0.2	40	20	12	72	122.67	194.67
45	5	Opt.	0.42	50	25	12	87	138	225
50	6	Opt.	0.7	50	25	15	90	153.33	243.33
55	7	Opt.	1.73	60	30	15	105	168.67	273.67
60	7	Opt.	0.59	60	30	15	105	184	289
65	9	Opt.	4.79	70	35	18	123	199.33	322.33
70	9	Opt.	2.28	70	35	18	123	214.67	337.67
75	10	Opt.	9.61	80	40	18	138	230	368
80	10	Opt.	1.45	80	40	18	138	245.33	383.33
85	11	Opt.	7.05	90	45	18	153	260.67	413.67
90	12	Opt.	21.56	90	45	24	159	276	435
95	14	Opt.	41.34	100	50	24	174	291.33	465.33
100	14	Opt.	25.12	100	50	24	174	306.67	480.67

Table A.11 Results for option (12 x c_i) with DCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.13	10	5	12	27	3.07	30.07
6	1	Opt.	0.05	10	5	12	27	18.4	45.4
12	1	Opt.	0.06	10	5	12	27	36.8	63.8
18	2	Opt.	0.25	20	10	12	42	55.2	97.2
24	3	Opt.	0.11	20	10	15	45	73.6	118.6
30	4	Opt.	0.3	30	15	15	60	92	152
36	5	Opt.	0.76	30	15	21	66	110.4	176.4
42	6	Opt.	0.7	40	20	18	78	128.8	206.8
48	8	Opt.	6.21	40	20	24	84	147.73	231.73
54	9	Opt.	22.15	50	25	24	99	165.6	264.6
60	9	Opt.	92.98	50	25	27	102	187.3	289.3
66	11	Opt.	133.18	60	30	27	117	202.4	319.4
72	12	Opt.	149.82	60	30	30	120	221.6	341.6
78	13	Opt.	270.69	70	35	30	135	239.2	374.2
84	13	Opt.	2,610.38	70	35	36	141	260	401
90	15	Opt.	1,952.31	80	40	33	153	277.2	430.2
96	16	-	1,493.93	80	40	39	159	295.47	454.47
102	18	-	1,192.63	90	45	39	174	312.8	486.8
108	18	-	1,750.44	90	45	42	177	336.1	513.1
114	19	-	7,200.00	100	50	42	192	349.6	541.6
120	20	-	1,134.60	110	55	45	210	370.93	580.93

Table A.12 Results for option (12 x c_i) with SCA-SI (Times are in minutes except for solution time)

Q	K	Optimality	Solution Time	Total Start Time	Total Body Loading Time	Total Comp. Installation Time	Total Set Up Time	Total Assembly Time	Total Time
1	1	Opt.	0.05	10	5	12	27	3.07	30.07
6	1	Opt.	0.11	10	5	12	27	18.4	45.4
12	1	Opt.	0.06	10	5	12	27	36.8	63.8
18	2	Opt.	0.27	20	10	12	42	55.2	97.2
24	3	Opt.	0.27	20	10	15	45	73.6	118.6
30	4	Opt.	0.64	30	15	15	60	92	152
36	5	Opt.	1.03	30	15	21	66	110.4	176.4
42	6	Opt.	1.5	40	20	18	78	128.8	206.8
48	8	Opt.	3.92	40	20	24	84	147.73	231.73
54	9	Opt.	6.76	50	25	24	99	165.6	264.6
60	9	Opt.	10.7	50	25	30	105	184.53	289.53
66	11	Opt.	12.17	60	30	27	117	202.4	319.4
72	12	Opt.	4.37	60	30	30	120	221.6	341.6
78	13	Opt.	12.15	70	35	30	135	239.2	374.2
84	13	Opt.	28.39	70	35	36	141	260	401
90	15	Opt.	82.48	80	40	36	156	276	432
96	16	Opt.	18	80	40	39	159	295.47	454.47
102	18	Opt.	45.3	90	45	39	174	312.8	486.8
108	18	Opt.	38.55	90	45	45	180	332.27	512.27
114	19	Opt.	59.11	100	50	42	192	349.6	541.6
120	20	Opt.	35.68	100	50	45	195	372	567

APPENDIX B

RESULTS FOR THE LOT SIZING PROBLEM

Table B.1 Results for USL-DCA

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	8	42	63	31	31	153	30	350
	4	Opt.	1	42	42	21	25	109	20	258
	6	Opt.	1	42	42	21	25	96	20	245
	8	Opt.	0	42	42	21	25	115	10	254
	10	Opt.	19	42	83	42	31	198	40	436
2 x 50	1	Opt.	1564	83	146	73	69	428	0	798
	3	Opt.	18	42	83	42	38	236	34	474
	5	Opt.	0	42	42	21	25	115	2	246
	7	Opt.	1959	42	146	73	50	396	56	762
	9	Opt.	2185	42	146	73	50	390	56	756
3 x 20	12	Opt.	715	83	104	52	56	281	28	605
	14	Opt.	19	42	63	31	31	160	50	376
	16	Opt.	0	42	42	21	25	115	14	258
	18	Opt.	110	42	83	42	38	243	68	515
	20	Opt.	438	83	104	52	56	319	42	657
3 x 50	11	Opt.	214	83	83	42	56	217	8	490
	13	Opt.	1887	83	125	63	63	383	36	752
	15	Opt.	528	83	104	52	56	319	42	657
	17	-	7200	83	167	83	81	511	24	949
	19	-	7200	125	167	83	87	485	12	959
4 x 20	22	Opt.	134	83	83	42	50	217	14	489
	24	-	7200	83	125	63	63	332	44	709
	26	Opt.	5303	83	125	63	63	370	88	791
	28	-	7200	83	125	63	63	345	36	714
	30	Opt.	186	42	83	42	38	224	64	492
4 x 50	21	-	7200	83	146	73	69	415	56	842
	23	-	7200	125	187	94	94	562	24	1086
	25	-	3152	125	208	94	87	524	72	1110
	27	-	7200	167	250	125	125	766	24	1456
	29	-	7200	83	167	83	75	466	40	914

Table B.2 Results for CSL-DCA

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	7.6	41.6	62.5	31.2	31.2	153.2	30.0	349.8
	4	Opt.	0.6	41.6	41.6	20.8	25.0	108.5	20.0	257.6
	6	Opt.	0.4	41.6	41.6	20.8	25.0	127.7	30.0	286.8
	8	Opt.	0.5	41.6	41.6	20.8	25.0	114.9	10.0	254.0
	10	Opt.	22.5	83.3	83.3	41.6	50.0	197.9	0.0	456.1
2 x 50	1	Opt.	1,126.2	83.3	145.7	72.9	68.7	427.8	0.0	798.4
	3	Opt.	18.1	41.6	83.3	41.6	37.5	236.2	34.0	474.3
	5	Opt.	0.1	41.6	41.6	20.8	25.0	114.9	2.0	246.0
	7	Opt.	1,524.8	83.3	145.7	72.9	68.7	395.9	0.0	766.5
	9	Opt.	1,431.9	83.3	145.7	72.9	68.7	389.5	0.0	760.1
3 x 20	12	Opt.	1,035.9	83.3	104.1	52.1	56.2	280.9	38.0	614.6
	14	Opt.	18.9	41.6	62.5	31.2	31.2	159.6	50.0	376.2
	16	Opt.	0.4	41.6	41.6	20.8	25.0	114.9	14.0	258.0
	18	Opt.	99.3	83.3	83.3	41.6	50.0	242.6	32.0	532.8
	20	Opt.	3,175.6	83.3	83.3	41.6	50.0	223.5	24.0	505.6
3 x 50	11	Opt.	137.0	83.3	83.3	41.6	56.2	217.1	8.0	489.5
	13	Opt.	1,851.9	83.3	124.9	62.5	62.5	383.1	36.0	752.2
	15	Opt.	1,025.6	83.3	104.1	52.1	56.2	319.2	42.0	656.9
	17	-	7,200.0	83.3	166.6	83.3	75.0	510.8	44.0	962.9
	19	-	7,200.0	83.3	166.6	83.3	75.0	490.0	52.0	950.1
4 x 20	22	Opt.	108.5	83.3	83.3	41.6	50.0	217.1	14.0	489.3
	24	-	7,200.0	124.9	124.9	62.5	75.0	332.0	14.0	733.3
	26	-	7,200.0	124.9	124.9	62.5	75.0	370.3	48.0	805.6
	28	-	7,200.0	124.9	124.9	62.5	81.2	344.8	20.0	758.3
	30	Opt.	81.4	124.9	104.1	52.1	75.0	319.2	22.0	697.3
4 x 50	21	-	845.0	83.3	166.6	72.9	68.7	415.0	56.0	862.4
	23	-	7,200.0	124.9	187.4	93.7	93.7	561.9	24.0	1,085.5
	25	-	7,200.0	124.9	187.4	93.7	93.7	523.6	58.0	1,081.2
	27	-	7,200.0	166.6	249.8	124.9	124.9	766.2	24.0	1,456.4
	29	-	1,575.2	83.3	166.6	83.3	75.0	466.1	46.0	920.2

Table B.3 Results for USL-SCA

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	2.0	41.6	62.5	31.2	31.2	153.2	30.0	349.8
	4	Opt.	0.3	41.6	41.6	20.8	25.0	108.5	20.0	257.6
	6	Opt.	0.3	41.6	41.6	20.8	25.0	127.7	30.0	286.8
	8	Opt.	0.2	41.6	41.6	20.8	25.0	114.9	10.0	254.0
	10	Opt.	5.7	41.6	83.3	41.6	31.2	197.9	40.0	435.7
2 x 50	1	Opt.	34.8	83.3	145.7	72.9	68.7	427.8	0.0	798.4
	3	Opt.	1.7	41.6	83.3	41.6	37.5	236.2	34.0	474.3
	5	Opt.	0.3	41.6	41.6	20.8	25.0	114.9	2.0	246.0
	7	Opt.	50.5	41.6	145.7	72.9	50.0	395.9	56.0	762.1
	9	Opt.	49.8	41.6	145.7	72.9	50.0	389.5	56.0	755.7
3 x 20	12	Opt.	36.7	83.3	104.1	52.1	56.2	280.9	28.0	604.6
	14	Opt.	2.4	41.6	62.5	31.2	31.2	159.6	50.0	376.2
	16	Opt.	0.3	41.6	41.6	20.8	25.0	114.9	14.0	258.0
	18	Opt.	16.1	41.6	83.3	41.6	37.5	242.6	68.0	514.7
	20	Opt.	11.6	83.3	104.1	52.1	56.2	319.2	42.0	656.9
3 x 50	11	Opt.	18.6	83.3	83.3	41.6	56.2	217.1	8.0	489.5
	13	Opt.	46.1	83.3	124.9	62.5	62.5	383.1	36.0	752.2
	15	Opt.	45.9	83.3	104.1	52.1	56.2	319.2	42.0	656.9
	17	Opt.	122.5	83.3	166.6	83.3	81.2	510.8	24.0	949.1
	19	Opt.	266.9	83.3	166.6	83.3	81.2	485.2	52.0	951.6
4 x 20	22	Opt.	19.9	83.3	83.3	41.6	50.0	217.1	14.0	489.3
	24	Opt.	178.4	83.3	124.9	62.5	62.5	332.0	44.0	709.1
	26	Opt.	136.4	83.3	124.9	62.5	62.5	370.3	88.0	791.4
	28	Opt.	94.8	83.3	124.9	62.5	62.5	344.8	36.0	713.9
	30	Opt.	15.0	41.6	83.3	41.6	37.5	223.5	64.0	491.5
4 x 50	21	Opt.	132.8	83.3	145.7	72.9	68.7	415.0	56.0	841.6
	23	Opt.	598.7	124.9	187.4	93.7	93.7	561.9	24.0	1,085.5
	25	Opt.	1,919.5	83.3	187.4	93.7	87.4	523.6	92.0	1,067.3
	27	Opt.	2,719.8	124.9	249.8	124.9	118.7	766.2	64.0	1,448.5
	29	Opt.	1,072.5	83.3	166.6	83.3	75.0	466.1	40.0	914.2

Table B.4 Results for CSL-SCA

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	2.8	41.6	62.5	31.2	31.2	153.2	30.0	349.8
	4	Opt.	0.3	41.6	41.6	20.8	25.0	108.5	20.0	257.6
	6	Opt.	0.3	41.6	41.6	20.8	25.0	127.7	30.0	286.8
	8	Opt.	0.2	41.6	41.6	20.8	25.0	114.9	10.0	254.0
	10	Opt.	4.8	83.3	83.3	41.6	50.0	197.9	0.0	456.1
2 x 50	1	Opt.	52.9	83.3	145.7	72.9	68.7	427.8	0.0	798.4
	3	Opt.	2.2	41.6	83.3	41.6	37.5	236.2	34.0	474.3
	5	Opt.	0.2	41.6	41.6	20.8	25.0	114.9	2.0	246.0
	7	Opt.	32.8	83.3	145.7	72.9	68.7	395.9	0.0	766.5
	9	Opt.	65.5	83.3	145.7	72.9	68.7	389.5	0.0	760.1
3 x 20	12	Opt.	35.5	83.3	104.1	52.1	56.2	280.9	38.0	614.6
	14	Opt.	3.2	41.6	62.5	31.2	31.2	159.6	50.0	376.2
	16	Opt.	0.4	41.6	41.6	20.8	25.0	114.9	14.0	258.0
	18	Opt.	16.8	83.3	83.3	41.6	50.0	242.6	32.0	532.8
	20	Opt.	106.9	124.9	104.1	52.1	75.0	319.2	22.0	697.3
3 x 50	11	Opt.	14.2	83.3	83.3	41.6	56.2	217.1	8.0	489.5
	13	Opt.	35.9	83.3	124.9	62.5	62.5	383.1	36.0	752.2
	15	Opt.	48.4	83.3	104.1	52.1	56.2	319.2	42.0	656.9
	17	Opt.	102.0	83.3	166.6	83.3	75.0	510.8	44.0	962.9
	19	Opt.	765.9	83.3	166.6	83.3	81.2	485.2	52.0	951.6
4 x 20	22	Opt.	21.3	83.3	83.3	41.6	50.0	217.1	14.0	489.3
	24	Opt.	336.0	124.9	124.9	62.5	75.0	332.0	14.0	733.3
	26	Opt.	59.4	124.9	124.9	62.5	75.0	370.3	48.0	805.6
	28	Opt.	780.0	124.9	124.9	62.5	81.2	344.8	20.0	758.3
	30	Opt.	17.9	83.3	83.3	41.6	50.0	223.5	24.0	505.6
4 x 50	21	Opt.	199.9	83.3	145.7	72.9	68.7	415.0	56.0	841.6
	23	Opt.	398.4	124.9	187.4	93.7	93.7	561.9	24.0	1,085.5
	25	Opt.	3,525.4	83.3	187.4	93.7	87.4	523.6	92.0	1,067.3
	27	Opt.	3,753.1	166.6	249.8	124.9	124.9	766.2	24.0	1,456.4
	29	Opt.	1,336.0	83.3	166.6	83.3	75.0	466.1	40.0	914.2

Table B.5 Results for USL-SCA1

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	0.2	41.6	62.5	31.2	31.2	153.2	30.0	349.8
	4	Opt.	0.0	41.6	41.6	20.8	25.0	108.5	20.0	257.6
	6	Opt.	0.1	41.6	41.6	20.8	25.0	127.7	30.0	286.8
	8	Opt.	0.0	41.6	41.6	20.8	25.0	114.9	10.0	254.0
	10	Opt.	0.2	41.6	83.3	41.6	31.2	197.9	40.0	435.7
2 x 50	1	Opt.	0.4	83.3	145.7	72.9	68.7	427.8	0.0	798.4
	3	Opt.	0.3	41.6	83.3	41.6	37.5	236.2	34.0	474.3
	5	Opt.	0.0	41.6	41.6	20.8	25.0	114.9	2.0	246.0
	7	Opt.	0.9	41.6	145.7	72.9	50.0	395.9	56.0	762.1
	9	Opt.	0.9	41.6	145.7	72.9	50.0	389.5	56.0	755.7
3 x 20	12	Opt.	1.1	83.3	104.1	52.1	56.2	280.9	28.0	604.6
	14	Opt.	0.3	41.6	62.5	31.2	31.2	159.6	50.0	376.2
	16	Opt.	0.1	41.6	41.6	20.8	25.0	114.9	14.0	258.0
	18	Opt.	0.3	41.6	83.3	41.6	37.5	242.6	68.0	514.7
	20	Opt.	0.7	83.3	104.1	52.1	56.2	319.2	42.0	656.9
3 x 50	11	Opt.	0.4	83.3	83.3	41.6	56.2	217.1	8.0	489.5
	13	Opt.	0.8	83.3	124.9	62.5	62.5	383.1	36.0	752.2
	15	Opt.	1.4	83.3	104.1	52.1	56.2	319.2	42.0	656.9
	17	Opt.	2.7	83.3	166.6	83.3	81.2	510.8	24.0	949.1
	19	Opt.	4.3	83.3	166.6	83.3	81.2	485.2	52.0	951.6
4 x 20	22	Opt.	0.7	83.3	83.3	41.6	50.0	217.1	14.0	489.3
	24	Opt.	4.6	83.3	124.9	62.5	62.5	332.0	44.0	709.1
	26	Opt.	4.4	83.3	124.9	62.5	62.5	370.3	88.0	791.4
	28	Opt.	1.8	83.3	124.9	62.5	62.5	344.8	36.0	713.9
	30	Opt.	0.7	41.6	83.3	41.6	37.5	223.5	64.0	491.5
4 x 50	21	Opt.	3.1	83.3	145.7	72.9	68.7	415.0	56.0	841.6
	23	Opt.	3.7	124.9	187.4	93.7	93.7	561.9	24.0	1,085.5
	25	Opt.	9.0	83.3	187.4	93.7	87.4	523.6	92.0	1,067.3
	27	Opt.	11.0	124.9	249.8	124.9	118.7	766.2	64.0	1,448.5
	29	Opt.	5.6	83.3	166.6	83.3	75.0	466.1	40.0	914.2

Table B.6 Results for USL-SCA2

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	0.3	41.6	62.5	31.2	31.2	153.2	30.0	349.8
	4	Opt.	0.1	41.6	41.6	20.8	25.0	108.5	20.0	257.6
	6	Opt.	0.1	41.6	41.6	20.8	25.0	127.7	30.0	286.8
	8	Opt.	0.1	41.6	41.6	20.8	25.0	114.9	10.0	254.0
	10	Opt.	0.5	41.6	83.3	41.6	31.2	197.9	40.0	435.7
2 x 50	1	Opt.	0.8	83.3	145.7	72.9	68.7	427.8	0.0	798.4
	3	Opt.	0.2	41.6	83.3	41.6	37.5	236.2	34.0	474.3
	5	Opt.	0.1	41.6	41.6	20.8	25.0	114.9	2.0	246.0
	7	Opt.	2.9	41.6	145.7	72.9	50.0	395.9	56.0	762.1
	9	Opt.	4.2	41.6	145.7	72.9	50.0	389.5	56.0	755.7
3 x 20	12	Opt.	1.1	83.3	104.1	52.1	56.2	280.9	28.0	604.6
	14	Opt.	0.4	41.6	62.5	31.2	31.2	159.6	50.0	376.2
	16	Opt.	0.1	41.6	41.6	20.8	25.0	114.9	14.0	258.0
	18	Opt.	0.6	41.6	83.3	41.6	37.5	242.6	68.0	514.7
	20	Opt.	0.9	83.3	104.1	52.1	56.2	319.2	42.0	656.9
3 x 50	11	Opt.	0.8	83.3	83.3	41.6	56.2	217.1	8.0	489.5
	13	Opt.	0.9	83.3	124.9	62.5	62.5	383.1	36.0	752.2
	15	Opt.	4.0	83.3	104.1	52.1	56.2	319.2	42.0	656.9
	17	Opt.	4.4	83.3	166.6	83.3	81.2	510.8	24.0	949.1
	19	Opt.	7.0	83.3	166.6	83.3	81.2	485.2	52.0	951.6
4 x 20	22	Opt.	0.9	83.3	83.3	41.6	50.0	217.1	14.0	489.3
	24	Opt.	7.1	83.3	124.9	62.5	62.5	332.0	44.0	709.1
	26	Opt.	8.2	83.3	124.9	62.5	62.5	370.3	88.0	791.4
	28	Opt.	1.7	83.3	124.9	62.5	62.5	344.8	36.0	713.9
	30	Opt.	0.7	41.6	83.3	41.6	37.5	223.5	64.0	491.5
4 x 50	21	Opt.	5.2	83.3	145.7	72.9	68.7	415.0	56.0	841.6
	23	Opt.	8.9	124.9	187.4	93.7	93.7	561.9	24.0	1,085.5
	25	Opt.	14.4	83.3	187.4	93.7	87.4	523.6	92.0	1,067.3
	27	Opt.	17.1	124.9	249.8	124.9	118.7	766.2	64.0	1,448.5
	29	Opt.	21.3	83.3	166.6	83.3	75.0	466.1	40.0	914.2

Table B.7 Results for USL-SCA3

Option	Instance	Optimality	CPU Time	Major Set Up Cost	Start Cost	Body Loading Cost	Comp. Installation Cost	Assembly Cost	Inventory Cost	Total Cost
2 x 20	2	Opt.	0.4	41.6	62.5	31.2	31.2	166.6	30.0	363.1
	4	Opt.	0.1	41.6	41.6	20.8	25.0	118.0	20.0	267.1
	6	Opt.	0.2	41.6	41.6	20.8	25.0	138.8	30.0	297.9
	8	Opt.	0.1	41.6	41.6	20.8	25.0	124.9	10.0	264.0
	10	Opt.	2.0	41.6	83.3	41.6	31.2	215.1	40.0	452.9
2 x 50	1	Opt.	3.4	83.3	145.7	72.9	68.7	465.0	0.0	835.6
	3	Opt.	0.7	41.6	83.3	41.6	37.5	256.8	34.0	494.8
	5	Opt.	0.1	41.6	41.6	20.8	25.0	124.9	2.0	256.0
	7	Opt.	4.7	41.6	145.7	72.9	50.0	430.3	56.0	796.5
	9	Opt.	3.2	41.6	145.7	72.9	50.0	423.3	56.0	789.6
3 x 20	12	Opt.	4.0	83.3	104.1	52.1	56.2	305.4	28.0	629.0
	14	Opt.	0.5	41.6	62.5	31.2	31.2	173.5	28.0	368.1
	16	Opt.	0.1	41.6	41.6	20.8	25.0	124.9	14.0	268.0
	18	Opt.	3.8	41.6	83.3	41.6	37.5	263.7	68.0	535.8
	20	Opt.	0.7	83.3	104.1	52.1	56.2	347.0	42.0	684.6
3 x 50	11	Opt.	1.2	83.3	83.3	41.6	56.2	236.0	8.0	508.4
	13	Opt.	4.6	83.3	124.9	62.5	62.5	416.4	36.0	785.5
	15	Opt.	6.1	83.3	104.1	52.1	56.2	347.0	42.0	684.6
	17	Opt.	3.8	83.3	166.6	83.3	81.2	555.2	24.0	993.5
	19	Opt.	6.1	83.3	166.6	83.3	75.0	527.4	52.0	987.5
4 x 20	22	Opt.	0.8	83.3	83.3	41.6	50.0	236.0	14.0	508.1
	24	Opt.	8.4	83.3	124.9	62.5	62.5	360.9	44.0	738.0
	26	Opt.	7.0	83.3	124.9	62.5	62.5	402.5	88.0	823.6
	28	Opt.	5.0	83.3	124.9	62.5	62.5	374.8	36.0	743.9
	30	Opt.	0.8	41.6	83.3	41.6	37.5	242.9	64.0	510.9
4 x 50	21	Opt.	7.8	83.3	145.7	72.9	68.7	451.1	56.0	877.7
	23	Opt.	14.4	124.9	187.4	93.7	93.7	610.7	24.0	1,134.4
	25	Opt.	21.7	83.3	187.4	93.7	81.2	569.1	92.0	1,106.6
	27	Opt.	277.1	124.9	249.8	124.9	118.7	832.8	64.0	1,515.2
	29	Opt.	9.2	83.3	166.6	83.3	75.0	506.6	38.0	952.7