

AN APPROACH TO MULTI-MODEL ASSEMBLY LINE BALANCING

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

BY

TEVHİDE FATMA ALTEKİN

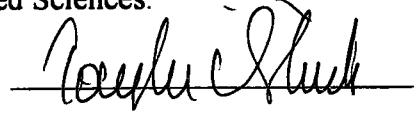
90688

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

DECEMBER 1999

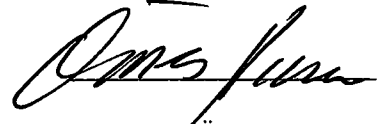
TC YÜKSEKÖĞRETİM KURULU
DOKÜMANTASYON MERKEZİ

Approval of the Graduate School of Natural and Applied Sciences.



Prof. Dr. Tayfur ÖZTÜRK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.



Prof. Dr. Ömer KIRCA
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.



Dr. Sedef MERAL
Co-Supervisor



Assoc. Prof. Dr. Levent KANDİLLER
Supervisor

Examining Committee Members

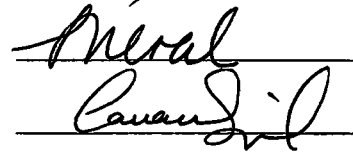
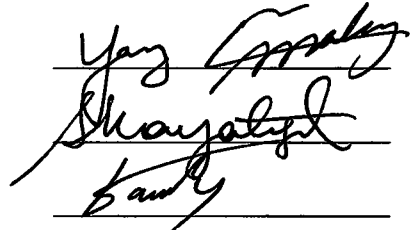
Asst. Prof. Dr. Yavuz GÜNALAY

Assoc. Prof. Dr. Sinan KAYALIGIL

Assoc. Prof. Dr. Levent KANDİLLER

Dr. Sedef MERAL

Assoc. Prof. Dr. Canan SEPİL



ABSTRACT

**AN APPROACH TO MULTI-MODEL
ASSEMBLY LINE BALANCING**

Altekin, Tevhide Fatma

M. S., Department of Industrial Engineering

Supervisor: Assoc. Prof. Dr. Levent Kandiller

Co-Supervisor: Dr. Sedef Meral

December 1999, 91 pages

In this study an assembly line which produces several models of a product in batches is analyzed. The objective is to minimize the number of workstations necessary to assemble all products. An approach is proposed that balances the assembly line under consideration using the multi-model assembly line balancing perspective. The proposed approach includes definition of some new concepts, upper and lower bounds and branch-and-bound procedures that facilitate multi-model assembly line balancing perspective. The proposed approach is compared with single-model and mixed-model assembly line balancing procedures. An experiment is conducted to compare these three methods and the results are analyzed statistically. Standard test problems taken from the literature are used in the experimentation.

Keywords: Multi-Model Assembly Lines, Assembly Line Balancing

ÖZ

ÇOKLU MODEL MONTAJ HATTI DENGEME PROBLEMİNE BİR YAKLAŞIM

Altekin, Tevhide Fatma

Yüksek Lisans Tezi, Endüstri Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Levent Kandiller

Yardımcı Tez Yöneticisi: Dr.Sedef Meral

Aralık 1999, 91 sayfa

Bu çalışmada bir ürünün birden fazla modelinin kafielerle üretildiği bir montaj hattı incelenmiştir. Amaç bütün modellerin üretilmesi için gerekli olan istasyon sayısının enazlanmasıdır. Söz konusu montaj hattının dengelenmesi için Çoklu-Model Montaj Hattı bakış açısını kullanan bir yaklaşım önerilmiştir. Önerilen yaklaşım Çoklu Model Montaj Hattı yaklaşımını uygulayabilmek için tanımlanmış kavramları, alt ve üst sınırları ve de dal-budak yöntemlerini içermektedir. Önerilen yaklaşım Tekli Model ve Karmaşık Model Montaj Hattı Dengeleme yordamlarıyla karşılaştırılmıştır. Bu üç yöntemi karşılaştırmak amacıyla düzenlenen deneyin sonuçları istatistiksel olarak analiz edilmiştir. Deneyde literatürden alınan standart test problemleri kullanılmıştır.

Anahtar Kelimeler: Çoklu Model Montaj Hatları, Montaj Hattı Dengeleme

ACKNOWLEDGEMENTS

Ever since I was a little girl I have wanted to be a marine biologist. In those days I said 'I want to be whatever Cousteau is', since I did not know the appropriate name for it. During the last few years the idea of diving in the oceans, documenting the wild life of the seven seas, has returned to me. Imagine the sun over the vast beach, fishermen catching tons of fish in big nets, a warm breeze and the smell of fresh air and salt water.

The only sun I ever see around here is a work station, the only net is Ethernet. The colorful fish are only a screensaver and instead of the smell of the sea I get stinking odors from the canteen. So, what have I been doing here? Well, it is all written in this thesis, but before I get into the twilight zone of Assembly Lines there are a number of people without whom this would never have been written.

I would like to express my gratitude to Assoc. Prof. Dr. Levent Kandiller and Dr. Sedef Meral for their valuable and patient supervision and support throughout this study. Studying with them was very instructive and fun for me. Their patience and trust for me are invaluable. Thank you for everything which I am unable to convey via words.

I would also like to thank Assoc. Prof. Sinan Kayaligil for his patience and invaluable comments that were very useful for this study. I would also like to thank him and Assoc. Prof. Dr. Yasemin Serin for the time and support they have given .

I would like to thank Armin Scholl for providing the executable form of SALOME-1 and Thomas .R. Hoffmann for providing the fortran code of EUREKA.

I owe many thanks to Cem Ulusoy, Pelin Bayındır, Oktay Türetken and Aslı Aytaç for bearing my complaints and giving me the encouragement and hope to start and finish this thesis.

Finally, I would like to express my deepest thanks to my family for providing the very best of everything throughout my life.

TABLE OF CONTENTS

ABSTRACT	iii
ÖZ	iv
ACKNOWLEDGEMENTS	v
TABLE OF CONTENTS	vi
LIST OF TABLES	viii
LIST OF FIGURES	ix
CHAPTERS	1
1. INTRODUCTION	1
2. LITERATURE REVIEW	4
2.1. Preliminaries	4
2.2. Assembly Line Balancing Problem	11
2.3. Classification of ALB Solution Procedures	14
2.4. Review of ALB Techniques	15
2.4.1. Single-Model Assembly Line Balancing	16
2.4.2. Multi/Mixed-Model Assembly Line Balancing	24
2.4.2.1. Mixed-Model Assembly Line Balancing	24
2.4.2.2. Multi-Model Assembly Line Balancing	27
3. PROPOSED APPROACH	29
3.1. Basics	32
3.2. Multi-Model Line Balancing Approach	34
3.2.1. Base Model Construction	34
3.2.2. Balancing for the Base Model	40
3.2.3. Balances for Individual Models	44
3.3. Solution Tool Box	53
4. EXPERIMENTAL ANALYSIS	54
4.1. Design of the Experiment	54
4.1.1. Factors and Factor Levels	54
4.1.2. Response Variables	56
4.1.3. Problem Generation	57

4.2. Experiment Design	58
4.3 Analysis of the Experimental Results	59
5. CONCLUSION AND FURTHER RESEARCH ISSUES	66
REFERENCES	71
APPENDICES	
A. CONSTRUCTION OF THE BASE MODEL	76
B. DETERMINATION OF LB_2	77
C. MODIFIED EUREKA ALGORITHM	79
E. THE MEAN PLOTS OF THE EXPERIMENTS	82



LIST OF TABLES

TABLES

2.1 ALB techniques.....	14
3.1 Notation for multi-model assembly line balancing.....	32
3.2 Demand and task times of the models.....	38
3.3 Similarity indeces of the models.....	38
4.1 Factors and their levels.....	55
4.2 Problem set M1.....	55
4.3 Problem set M2.....	55
4.4 Summary of ANOVA 1.....	60
4.5 Summary of Tukey follow up analysis 1.....	61
4.6 Summary of ANOVA 2.....	63
4.7 Summary of Tukey follow up analysis 1.....	64

LIST OF FIGURES

FIGURES

2.1 Single-model, mixed-model, multi-model assembly lines	7
2.2 Example of a precedence diagram	10
2.3 The precedence matrix of the example in Figure 2.2.....	10
2.4 A classification of the assembly line balancing techniques.....	12
2.5 Precedence diagram composed of two models	25
3.1 Precedence diagrams of models 1, 2 and 3	37
3.2 Combined precedence diagram of models 1, 2 and 3	37
3.3 Base Model	40
3.4 Modified EUREKA solution for the base model	44
3.5 Demonstration of upper bound	52
E.1 Average number of stations with maximum cycle time	83
E.2 Average number of stations with medium cycle time	84
E.3 Average number of stations with minimum cycle time	85
E.4 Maximum number of stations with maximum cycle time	86
E.5 Maximum number of stations with medium cycle time	87
E.6 Maximum number of stations with minimum cycle time	88
E.7 Balance delay with maximum cycle time	89
E.8 Balance delay with medium cycle time	90
E.9 Balance delay with minimum cycle time	91

CHAPTER 1

INTRODUCTION

An assembly line basically consists of a finite set of work elements or tasks and a set of precedence relationships that specify the permissible ordering of the tasks. The fundamental line-balancing problem is to assign the tasks to an ordered sequence of stations, such that the precedence relationships are satisfied and some measure of effectiveness is optimized.

Buxey, Slack and Wild (1973) identify three varieties of manual assembly lines, namely, single-model lines dedicated to the production of a single product; multi-model lines on which two or more similar models of a product are produced separately in batches, and mixed-model lines on which two or more similar models of a product are produced simultaneously on the line where the batch size is usually very small or even one. The multi-model and mixed-model assembly line design includes several other issues besides the fundamental line balancing problem. These issues are determining the sequence of the models to be produced on the line, the batch sizes of the products -for multi-model assembly lines only, and the size and place of the buffers to be placed on the line.

In today's industry, based on our limited observations, assembly lines devoted to a single product are relatively less common. This is especially true for the automobile and home appliances' assembly. Due to the increasing competitiveness in the global market, companies are trying to enhance their production flexibility through reducing their batch sizes and increasing product

varieties. Resultantly, the batch sizes of the various models are made smaller through time, but not yet as small as one. Hence, especially in Turkey, the family of products under consideration is assembled on *multi-model assembly lines* rather than *mixed-model assembly lines*. Therefore, the main motivation of our study for working on multi-model assembly lines rather than the others stems from our limited observations in industrial applications.

There exists a rich literature on the solution procedures for balancing Single-Model Assembly Lines. Mixed-Model Assembly Line balancing and Single-Model Assembly Line balancing differ only in the number of different precedence diagrams that they handle. Therefore the solution procedures proposed for the Single-Model Assembly Lines can, as well, be used for balancing Mixed-Model Assembly Lines after the consolidation of the precedence diagrams of individual models into a single combined precedence diagram. However, the studies proposing solution procedures for Multi-Model assembly line balancing are yet quite a few.

In a multi-model assembly line environment, it is possible to balance the line for each model separately as in the case of single-model assembly line. This has got some advantages in terms of the efficiency of the line, and some disadvantages in terms of excessive setup operations and some learning curve effect. It is also possible to balance the line for only once for the given model mix using the mixed-model assembly line balancing solution procedures. This second approach has got some advantages in terms of setup operations requires and learning curve effect, since the same task in different models is always performed on the same workstation. However, this approach has been reported so far to be inferior in terms of the line efficiency with respect to the first approach.

Multi-model line balancing when used has got some advantages and disadvantages as well. In terms of line efficiency, it may perform better than mixed-model line balancing and worse than single-model line balancing. In terms of setup operations and learning curve effect, it may perform better than single-

model line balancing and worse than mixed-model line balancing. In multi-model line balancing the batch sizing and batch sequencing are critical issues.

The aim of this study is to propose an approach, which can be used in multi-model line balancing. A formal framework is developed which includes multi model assembly line balancing. Our approach includes the definition of a set of tasks called the *Fixed Tasks Set*. Due to the setup considerations and learning curve effects on the operators, it is desired to perform the fixed tasks at the same workstation for all models. The other tasks of the models are allowed to be performed at different workstations for different models. Thus, it may be possible to obtain better line balances in terms of measures related to the performance of the line than those obtained using the mixed-model line balancing approach for the multi-model case.

The thesis includes five chapters. In Chapter 2, the introduction of basic assembly line balancing concepts is presented first then the assembly line balancing problem is defined and available solution procedures are presented while the studies that are more relevant to our study are summarized in the last section of the chapter. In Chapter 3, the problem under consideration and its environment is defined. In this context, the basic multi-model assembly line considered in this study is discussed first. Then an outline of the line-balancing approach proposed is presented and finally the stages of our approach and its components are discussed in detail. Chapter 4 presents the design and analysis of the experiments together with the discussion of the results. The performance of our multi-model line balancing approach is tested against the single-model line balancing and mixed-model line balancing approaches that can also be practiced for the multi-model lines. It is concluded in Chapter 5 with a summary and results of the study along with the suggestions for future research.

CHAPTER 2

LITERATURE REVIEW

2.1. Preliminaries

It becomes economical to design and layout a special facility dedicated exclusively to a product (or a family of technologically similar products), when the product (or a family of products) under consideration exhibits high volume and stable demand over lengthy periods of time. The manufacturing stages are physically arranged in a tandem sequence according to their technological ordering. The objective is to reduce the work-in-process inventories and non-productive times due to loading, unloading and transportation between successive stages. Resulting facility is called an *assembly line* if the production process is assembly or *fabrication line* if it is fabrication (Hax and Candea, 1984).

In many high-volume parts manufacturing and assembly operations, assembly lines are used, since they affect certain economies of operation and provide a more uniform flow than might be normally expected from a series of individual operations. They are extensively used in mass production systems where consumer durables are produced. Extensive research has been conducted on assembly lines since the late 50s due to their practical significance.

The complete assembly activity of the product (or the family of products) on the assembly line is divided into *tasks*. A *task* is the smallest indivisible work element that adds value to the product. A subset of these tasks is assigned to each

workstation. A *workstation* is a certain segment along the line where work is performed on the product usually by the addition of parts. A workstation consists of operators, machinery and/or equipment. A material handling system passes down the products on the line. The products visit each workstation in sequence. Upon exiting the final workstation, the products become completed.

The operation modes of assembly lines can either be manual or automatic. If an assembly lines' operation mode is manual, it is called a *manual assembly line*; and if its operation mode is automatic, it is called an *automatic assembly line* (Wild, 1974). In the former, operators carry out the corresponding manufacturing operations in each workstation while in the latter, operations are carried out automatically. Manual assembly lines are studied in this study.

Workstations on an assembly line are typically connected by continuous material handling systems. According to the type of the material handling system used, assembly lines are categorized into two classes. The first class is *non-mechanical lines*, where operators are normally free of any mechanical pacing effect. Blocking (i.e. the inability of an operator to pass his work) and starvation (i.e., lack of available work arriving at the station) can cause idle times at the workstations. Hence, the use of buffer stocks between stations is an important feature for non-mechanical lines. The second type is *moving-belt lines*. Two sub-categories of moving-belt lines exist; depending on whether items are removable from the belt or not. Physical buffer stocks of items are quite rare on moving-belt type lines. Idle time at stations on moving-belt type lines result only e due to starving (Buxey, Slack and Wild, 1973).

Assembly lines rely heavily on two concepts. One of these concepts is the *principle of interchangeability*, which suggests that individual components that make up a product should be interchangeable between product units. The other concept is the *division of labor* and includes concepts of work simplification, standardization, and specialization. This principle facilitates the subdivision of complex activities into tasks. These two concepts yielded mass production, allowed replacement parts to be used to lengthen a product's useful life and made

possible the development of assembly lines that was pioneered by Henry Ford and others (Askin and Standridge, 1993).

Advantages of assembly lines include the ability of keeping the labor and/or automated machines busy doing productive work. Since the tasks are repeated through cycles in assembly lines, the setup requirements are normally minimal. The repetition of tasks also helps workers operate at effective points on their learning curves. Assembly lines do not require large amount of work-in-process inventory. Consequently, there is less space required, inventory holding costs are decreased, flow time is shortened and throughput is increased. The high utilization of available resources and continuous nature of assembly lines also allow matching of production and demand rates (Askin and Standridge, 1993).

Bhattacharjee and Sahu (1988) state the benefits of balanced assembly lines as follows:

- Large quantities of standardized items are produced at low costs.
- Work-flow through the plant is efficient.
- Employment and quicker training of less skilled operators is possible.
- Productivity is increased.
- Work congestion is decreased.
- Materials handling is reduced.

Assembly Line Design

There are numerous factors that can be considered when designing, balancing and/or scheduling assembly lines. Ghosh and Gagnon (1989) have defined the integrated consideration of designing, balancing and scheduling of assembly lines as the 'Assembly Line Design' Problem. The most significant problems within the assembly line design are briefly introduced in this section.

Assembly Line Balancing problem deals with the assignment of the tasks to the successive workstations in the line without violating the technological sequencing requirements of the tasks, i.e. the *precedence relations*, such that cycle

time or the number of stations is minimized. This problem is related to the design of flow lines. Buxey, Slack and Wild (1973) define three varieties of production lines: *single-model lines* are devoted to the production of one model; *multi-model lines* produce two or more models of a product in batches, and *mixed-model lines* produce two or more models of a product simultaneously on the line. Figure 2.1 presents assembly lines consisting of eight workstations working as single-model lines, multi-model lines and mixed-model lines (workstations are shown with squares and the models being worked on with letters).

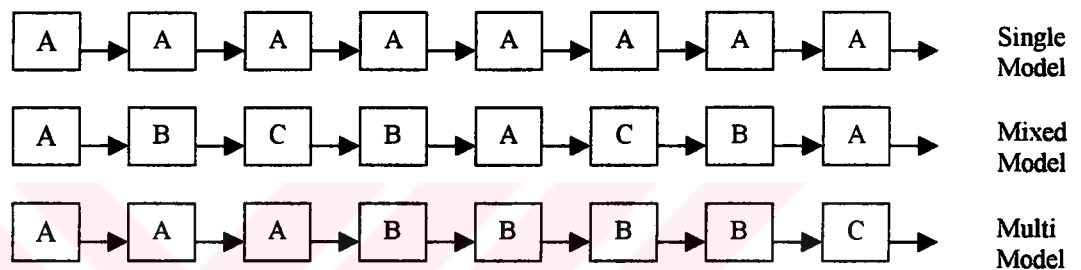


Figure 2.1 Single-model, mixed-model, multi-model assembly lines

Allocation of Models to Lines is relevant to multi-model and mixed-model production lines. The problem addressed is to assign models to several lines so as to minimize the production cost associated with each line (Lehman, 1969).

Model Sequencing is likewise relevant to multi-model and mixed-model production environments. Set-up or conversion costs in these environments might affect the order in which the products are scheduled on the line. The cost of switching the assembly line from one model to another depends upon the number of set-up operations and the labor hours required. On the assembly line, if some of the tooling is common for two or more models, the scheduling of the models with common tooling one after another might incur a set-up cost less than a sequence of models with non-common tooling. Hence, the problem becomes the determination of the best order to schedule models through the assembly line -in batches if it is a multi-model line, so as to minimize line set-up or change-over costs while meeting the demand (Young, 1967).

Model Batch Sizing is related to multi-model lines where models are produced in batches. The batch sizes may vary depending on the model and its total demand. The batch sizes affect the profitability of an assembly line schedule, as set-up costs or conversion costs and the storage costs for the finished products are involved. Hence the problem is to determine the economic batch sizes and the number of batch runs required to meet a given demand and minimize storage and line set-up costs (Young, 1967).

Buffer Stocks are provided to avoid station idle time or underutilization. While establishing buffer capacities, conflicting requirements of station idle time minimization and work in process minimization are concerned with (Buxey, Slack and Wild, 1973).

Layout of the assembly line modelled in traditional line balancing problem is usually linear. Recently, as a consequence of Just-In-Time production principles, new assembly lines are being arranged in “U-lines” rather than straight lines. U-lines have some advantages over traditional straight lines. These include improvement in communications via the close proximity of operators to each other; having multi-skilled operators; adjusting the output rate by the addition or removal of operators; required number of stations that is never more than that required by the traditional line (Miltenburg and Wijngaard, 1994).

Parallel or Multiple Stations might be required when processing of some tasks exceeds the cycle time. Also, it might be required to do so, if a satisfactory line balance is aimed at (Young, 1967). The problem is the determination of the best possible combination of the number of parallel stations and the work elements at each station such that the efficiency of the line is maximized (Akagi et al, 1983).

Pacing (of the Assembly Line) is the choice between *paced* and *unpaced lines*. In a paced line (synchronous), each workstation is given exactly the same amount of time to operate on each unit of product. At the end of this time, the material handling system automatically indexes each unit to the next station. In an

unpaced line (asynchronous), the station removes a new unit from the material handling system as soon as it completes the previous unit, performs the required tasks and then forwards the unit to the next station. The choice between paced and unpaced lines is affected by the processing time variability mainly, which may give rise to items not completed on time in the paced-line case. If buffers are allowed between workstations, the unpaced lines may be preferable to paced lines (Askin and Standridge, 1993).

Assembly Line Balancing

Among the assembly line design issues presented, *assembly line balancing* has been studied more in the literature. The line balancing process is intended to achieve the best compromise between labor and facilities to meet a given demand requirement (Milas, 1990). Prior to discussing the benefits of assembly line balancing, some related terms and concepts are introduced.

The time required to complete a task i is termed as *task time (task performance time)* and is denoted by t_i . The actual amount of work assigned to a workstation is the sum of task times of the tasks assigned to that workstation, which is called the *work content* of that station. *Cycle Time*, C , is the time available on each workstation. The work content of a workstation should not exceed the cycle time. The cycle time is based on the demand for the product in the given time horizon and/or the given operating time for the manufacturing system in that time horizon. Thus, the *production rate* for the line is $1/C$. *Idle time* of a workstation is the difference between the cycle time and its work content. *Total idle time* is the sum of the station idle times that can be a measure of the efficiency of a line design. A related measure of efficiency is the *balance delay*, which is the proportion of total idle time to the total time spent by the product on the line moving throughout the line.

The precedence relations of the products prevent the arbitrary assignment of tasks to workstations. The processing of a task may not start until certain tasks, i.e., its *immediate predecessors* have been processed. A *precedence diagram* is a

graphical description of the ordering in which tasks must be performed for the total assembly of the product. In this diagram, nodes represent the tasks. If task i is an immediate predecessor of task j , a directed arc (i,j) is constructed to represent this relationship. Figure 2.2 gives an example of a precedence diagram with five tasks. Also a *precedence matrix* can be used demonstrate the precedence relations. The precedence matrix has an entry '1' for the i th row and j th column if task j follows task i in the precedence graph; otherwise the corresponding entry is '0'. Figure 2.3 gives the precedence matrix for the example provided in Figure 2.2.

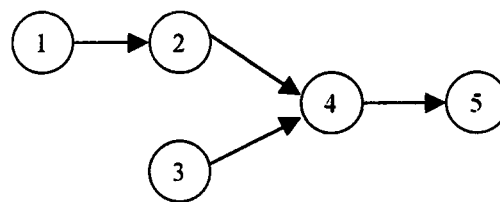


Figure 2.2 Example of a precedence diagram

	1	2	3	4	5
1	0	1	0	0	0
2	0	0	0	1	0
3	0	0	0	1	0
4	0	0	0	0	1
5	0	0	0	0	0

Figure 2.3 The precedence matrix of the example in Figure 2.2

Assembly line balancing aims at the minimization of balance delay, labor force and overall assembly cost per unit (Bhattacharjee and Sahu, 1988). The assembly line is said to be *balanced* if the sum of the idle times of the stations along the line is as low as possible. The assembly line is said to be *perfectly balanced* if the tasks can be grouped so that work contents of each station are exactly equal. In most practical, situations, a perfect balance is very difficult to achieve (Baybars, 1986).

2.2. Assembly Line Balancing Problem

The *Assembly Line Balancing* (ALB) problem, is “to assign tasks to an ordered sequence of workstations, such that precedence relations are satisfied and some measure of effectiveness is optimized” (Ghosh and Gagnon, 1989).

The most commonly used objectives in assembly line balancing are of two categories. The first category involves the minimization of total idle time for a given production rate. This objective is in fact equivalent to the minimization of number of stations given the cycle time. The second category involves the minimization of cycle time for a fixed number of stations. When the assembly line balancing problem considers the first objective, it is called *Type I problem* and it is called *Type II problem* otherwise (Erel and Sarin, 1998). Most of the procedures developed deal with the Type I problem. However, in some situations, the management may want to maximize the utilization of the existing work force on the assembly line. When direct labor cost is fixed, this issue may arise and is handled as the Type II problem (Uğurdağ, Rachamadugu and Papachristou, 1997). Still there are some other objectives studied in the ALB literature, like minimizing balance delay or minimizing inventory, set-up and idle time. Other technical and economical objective criteria used in the ALB literature are listed by Ghosh and Gagnon (1989).

The ALB problem in its simplest form is solved subject to the following constraints: 1) All tasks have to be processed. 2) A task cannot be split among two or more stations. 3) Tasks cannot be processed in arbitrary sequences due to technological precedence requirements. 4) The work content in any workstation cannot exceed the cycle time (it is assumed that the cycle time is greater than the maximum task time of the product).

The ALB problem and the research conducted are classified into four categories as shown in Figure 2.4: Single-Model Deterministic, Single-Model Stochastic, Multi/Mixed-Model Deterministic, and Multi/Mixed-Model Stochastic.

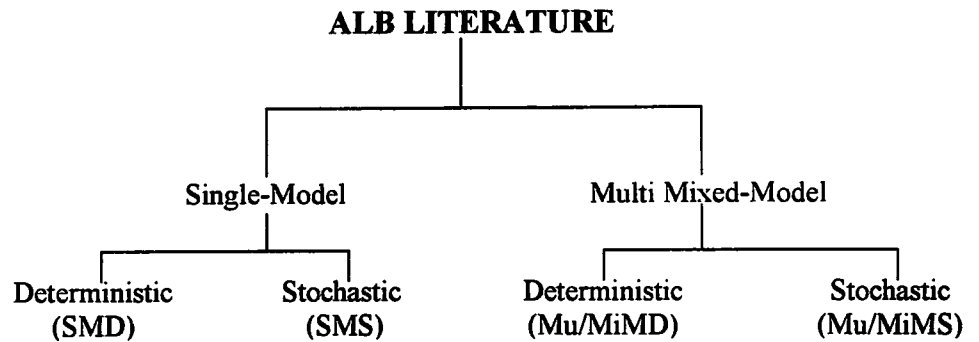


Figure 2.4 A classification of assembly line balancing literature

Ghosh and Gagnon (1989) describe these four categories as follows:

1. *Single-Model Deterministic (SMD)*: In SMD version, dedicated, single-model assembly lines with known task times are assumed. This is the simplest form of the Assembly Line Balancing Problem (SALB). When other factors or restrictions like paralleling of stations, restricting the grouping of certain tasks at the same station, e.g. *zoning restrictions*, or having *must-do-tasks* that have to be processed at a particular station are introduced, the problem becomes General Assembly Line Balancing Problem (GALB).
2. *Single-Model Stochastic (SMS)*: The concept of task time variability is introduced in this category. In manual assembly lines, where operators' operation times are rarely constant; the line is SMS type. Many other issues become relevant with the introduction of stochastic task times, such as work contents of stations exceeding the cycle time (and perhaps defective production or unfinished parts), pacing effects on operators' task times, station lengths, the size and location of work-in-process inventory buffers, launch rates and allocation of line imbalances.

3. *Multi/Mixed-Model Deterministic (Mu/MiMD)*: This version assumes deterministic task times, but introduces the concept of an assembly line where several models of a product are produced. Mu/MiMD problem includes several other issues. Here, model selection, model sequencing and launching rate(s) and batch sizes are also critical.
4. *Multi/Mixed-Model Stochastic (Mu/MiMS)*: All factors arising from stochasticity that are relevant in the SMS version of the problem are also pertinent to this problem. Moreover, factors such as learning curve effects, operator skill level, job design and operator task time variability become more difficult to track. This case is even more complex since the rebalancing of the line for each model makes the above factors more difficult to track.

The single-model deterministic and the multi/Mixed-Model deterministic versions of the assembly line balancing problem are within the scope of this study.

The SALB problem is easy to formulate. However, even the SALB problem is NP-hard (Ignall, 1965). The enumeration of the feasible task sequences requires an enormous effort. The SALB problem has a finite but extremely large number of feasible solutions. The problem's inherent integer restrictions result in enormous computational difficulties. There are $n!$ different sequences of n tasks, without considering the precedence constraints. However, the precedence and cycle time constraints drastically reduce this number. For r precedence relations among n tasks, there are roughly $n!/2r$ distinct sequences. Even this is too large to handle (Erel and Sarin, 1998).

Due to the computational complexity of the problem, to achieve optimal or at least acceptable heuristic solutions, various solution methodologies have been suggested in the literature and these are presented in the next section.

2.3. Classification of ALB Solution Procedures

The assembly line balancing solution procedures are analyzed in two categories. The first category consists of the *optimal seeking procedures*, while the second category consists of the *heuristic procedures*. Table 2.1 presents the optimal seeking and heuristic solution procedures for the deterministic case with their frequencies of use in the literature. These figures have been taken from Ghosh and Gagnon (1989) and updated by the inclusion of the studies made since then.

Table 2.1 ALB techniques

<i>ALB Solution Procedures</i>	<i>Frequency of Use</i>	
	<i>SMD</i>	<i>Mu/MiMD</i>
<i>Optimal Seeking Methods</i>	<i>31</i>	<i>6</i>
Linear Programming	1	0
Integer Programming	7	1
Dynamic Programming	4	0
Goal Programming	2	0
Branch-and-bound	13	3
Network Flows		
Shortest Path	3	2
Maximal Path	1	0
<i>Heuristic Methods</i>	<i>27</i>	<i>9</i>
Priority ranking and assessment	10	7
Tree search (heuristic B & B)	8	0
Randomized	3	0
Other	6	2

The optimal seeking solution procedures include *Linear Programming*, *Integer Programming*, *Dynamic Programming*, *Goal Programming*, *Branch-and-bound* and *Network Flows as Shortest Path or Maximal Path*. Integer and dynamic programming formulations have been used extensively to express the ALB problem, but neither can be solved efficiently. Branch-and-bound techniques, based on general integer programming methods, have gained

widespread use for integer programming formulations. Specialized branch-and-bound algorithms represented by the tree networks, in which each path corresponds to a feasible solution and each arc represents a station, have also been frequently used to solve the ALB problem. However, the branching and fathoming rules in specialized branch-and-bound algorithms are heuristics (Ghosh and Gagnon, 1989).

The heuristic solution procedures are actually more difficult to categorize. The most widely used heuristic method is the *priority ranking and assessment approach*, where tasks are ranked according to some criteria or priority rule (e.g. largest task time or largest positional weight) and then assigned to stations. The second widely used heuristic method is *tree search* or enumerative methods. They are essentially branch-and-bound methods, with different bounding, branching, fathoming and termination rules. They are considered as heuristics, since they use heuristic fathoming rules and do not explore the whole branch-and-bound tree. Ghosh and Gagnon (1989) have used the term ‘branch-and-bound’ for optimal seeking procedures and the term ‘tree search’ for heuristic procedures. Another heuristic method used is the *random sampling* method where tasks are assigned to stations randomly or a rule for assigning a task to a station is selected randomly (from a set of rules). Other heuristic methods used in the ALB literature include (Ghosh and Gagnon, 1989): 1) “*Trade and transfer*” methods where a balance is first achieved by various methods and then tasks are interchanged to achieve a better balance. 2) “*Task grouping*” methods where tasks are grouped into compound tasks. 3) “*Successive approximation*” methods where an optimal algorithm is solved using an approximation technique. 4) “*Learning mechanism*”s in which heuristics developed from experience are later used to solve larger problems. 5) *Simulation*.

2.4. Review of ALB Techniques

Since Salveson (1955) first published the analytical statement of the ALB problem, the ALB problem has been studied by the academicians.

Buxey, Slack and Wild (1973) and Johnson (1981), besides many other researchers, reviewed the work published on this subject. The most recent review articles are as follows. Baybars (1986) discusses the development of simple assembly line balancing problem along with the modifications and generalizations over time and he describes and comments on the optimal seeking methods for the single-model deterministic case. Ghosh and Gagnon (1989) report the results of a comprehensive review and analysis of the ALB literature. They establish numerous quantitative and qualitative factors mentioned in the literature that could affect the design, balancing and scheduling of assembly systems, into an eight level hierarchical taxonomy. Then they use this taxonomy to assess the progress of ALB literature in designing and operating assembly systems. Erel and Sarin (1998) in their survey critically examine and present the heuristic procedures developed in the ALB literature. Besides evaluating the procedures, they point out the design issues that might need further research.

In the following section, our aim is to discuss the part of the ALB literature that is closely related to our study. In this study we introduce an approach to Multi-Model Line Balancing. However, for comparison purposes, Single-Model Line Balancing and Mixed-Model Line Balancing procedures have been included in the study as well. Therefore, the following section includes brief reviews of the literature for all three cases.

2.4.1. Single-Model Assembly Line Balancing

In Single-Model Assembly Line Balancing, most of the research focuses on Type I problem. Most procedures exactly solving Type I problem, are based on branch-and-bound procedures as well as on dynamic programming. In the recent years, mostly branch-and-bound procedures have been developed (Klein and Scholl, 1996). Therefore, only the branch-and-bound procedures are presented here. The solution procedures developed to solve Type II problem mostly include modified versions of the procedures for Type I problem. Only a few exceptions exist in the literature and they are presented after solution procedures for Type I problem are discussed.

Type I Problem

Type I problem consists of minimizing the number of workstations given a set of partially ordered assembly tasks and a cycle time. In branch-and-bound procedures, the feasible solutions to the problem are represented by enumeration trees.

Jackson (1956) constructed the first branch-and-bound procedure. In this procedure, all assignments except for the last station are examined explicitly before any assignment is made to the last station. Although the procedure is not a complete enumeration, it was found to be time-consuming. He also showed that an optimal solution exists only in a full enumeration tree where branches represent *maximal* stations. A station is maximal, if no unassigned task can be added feasibly without exceeding the cycle time or violating the precedence constraints.

Since then, several branch-and-bound procedures have been developed in the literature. Moreover, there are studies where the most effective branch-and-bound procedures for Type I problem have been compared for their performances. Such studies include Johnson (1981), Hoffmann (1992), Nourie and Venta (1996), Scholl and Klein (1999) and Sprecher (1999). The branch-and-bound procedures used in the comparison studies usually include Johnson's (1988) FABLE, Hoffmann's (1992) EUREKA, Scholl and Klein's (1997) SALOME and Sprecher's (1999) AGSA. Until 1999, these studies showed that SALOME was so far the best branch-and-bound procedure. However, the most recent study of Sprecher (1999) showed that his procedure clearly outperforms SALOME.

Before getting into the details of each of the above branch-and-bound procedures, some general concepts regarding branch-and-bound algorithms are introduced. Any branch-and-bound procedure can be characterized by the following concepts (Scholl and Klein, 1997):

1. **Number of solutions generated:** The *single solution (lower bound) methods* determine one feasible solution. Starting with the lower bound on the number of stations, a solution with an objective function equal to the lower bound is sought. The first solution found is the optimal. The *multiple solution (upper bound) methods* generate several solutions successively. As a rule these successively generated solutions are of increasing quality. Hence, the last solution found is optimal if all of the branch-and-bound tree is constructed.
2. **Planning direction and relabelling:** Among branch-and-bound methods, *forward planning procedures* make use of the original precedence graph, while *backward planning procedures* make use of the reversed precedence graph. Procedures, which make use of both the original precedence graph and the reversed precedence graph, are called *bi-directional procedures*. *Relabelling* of the tasks via a priority rule can be used in order to speed up the convergence of branch-and-bound procedures (Sprecher, 1999).
3. **Branching scheme:** In branch-and-bound procedures, the branching scheme can be either task-oriented or station-oriented. In *task-oriented approach*, one task per node of the branch-and-bound tree is assigned. In *station-oriented approach* a complete load to one workstation is assigned per node of the branch-and-bound tree. At a certain level of the branch-and-bound tree, the search strategy can be one of the followings:

In *depth-first (laser) search strategy*, each node created is immediately branched again. In *breadth-first search with complete branching*, all subproblems of a node to be branched are generated and the one with the best value for the lower bound is selected for branching. The remaining subproblems are sorted in increasing order of bound values and stored in a local candidate list. At each re-visit of the current node, the next subproblem is removed from the list and selected for branching. In the *best-first search strategy*, again all subproblems of a node to be branched are generated, but this time all known but unbranched nodes of the tree are stored in a global list. From the list, the subproblem with the lowest bound value is taken for

branching. The advantage of laser search over the other two search strategies is that, only a single branch of the tree is stored at a time, whereas the other two require storing of local or global candidate lists. Enormous memory requirements arise for large problem instances with several hundreds of tasks and this requirement still exceeds the availability of current computer systems. Moreover, best-first search may fail finding feasible solutions to complex problem instances when there is a limit imposed on the computation time. However, best-first search strategy might find optimal solutions faster than laser search for small instances, since the search is directed towards promising subproblems (Scholl and Klein, 1999).

4. **Dominance concepts employed:** Different logical tests in terms of dominance concepts are employed in branch-and-bound procedures for fathoming. The most intuitive dominance rule is *maximum load rule*, which allows fathoming of any subproblem whose partial solution contains a non-maximal load. The *Jackson's dominance rule* is an extension of the maximum load rule and based on potential dominances. A task h *potentially dominates* a task j if t_h is greater than or equal to t_j and all successors of j are also successors of h . Whenever a task j contained in a station load can feasibly be replaced by an unassigned task h which potentially dominates task j , the respective node can be fathomed. Besides these two very commonly used dominance rules, many others have been defined and used in different branch-and-bound procedures (Johnson, 1988; Scholl and Klein, 1997 and Sprecher, 1999).

5. **Lower bounds applied:** There are two bounds which are widely used in different branch-and-bound procedures. The assembly line balancing problem can be regarded as bin packing problems with additional (precedence) constraints. Therefore, any valid lower bound for the bin packing problem is also valid for this problem. The first lower bound (LB_1) uses this idea and it is

$$LB_1 = \left\lceil \sum_{i=1}^n t_i / C \right\rceil$$

A number of additional bound arguments exist which are also based on correspondences of the assembly line balancing problem with one-machine and parallel-machine sequencing and vehicle routing problems. For example, the number of tasks whose processing time is equal to the cycle time or exceed the cycle time is a lower bound on the number of stations, since each of these tasks require a station of its own. Instead of LB_1 , Berger, Bourjolly, and Laporte (1992) have used a stronger lower bound developed by Labbe, Laporte and Mercure (1991) in vehicle routing problem context. The main idea is that two tasks with task times greater than half of the cycle time cannot be assigned to the same workstation. They describe a procedure that calculates a lower bound by making use of this fact. We have also used this bound, LB_2 , within our approach, hence a complete description of the procedure is presented in Chapter 4.

Johnson (1988) has developed a branch-and-bound algorithm called **Fast Algorithm for Balancing Lines Effectively (FABLE)**. FABLE is a multiple-solution method. It performs the planning in a forward manner and the branching scheme is task-oriented. The search strategy used is laser search. The tasks are relabelled in FABLE. Since FABLE performs a laser search, a feasible solution is found in the first complete branch of the tree. The quality of the solution depends on the task-renumbering scheme used. Several different logical tests including maximum load rule are employed. FABLE uses four different lower bounds for bounding. It is an effective algorithm but has some limitations. The logical tests and bounds used reduce the size of the branch-and-bound tree such that an optimal solution can be found in a reasonable amount of time. However, it may be trapped within its rigid enumeration scheme so that the computation time available may be reached prior to finding the optimal (or at least near optimal solution).

Hoffmann's (1992) EUREKA is a single solution method. It performs forward planning; and if no solution is found within the allowed time, then it performs backward planning. If no solution is found within the allowed time during backward planning, a heuristic procedure developed by Hoffmann (1963) is used to determine a feasible solution. The branching scheme used is station-

oriented. The search strategy used is laser search. The tasks are not relabelled. For bounding, LB_1 is used implicitly. EUREKA has the advantage due to the heuristic application, which sometimes finds a good or even optimal solution. When the heuristic solution is poor or the initial lower bound (LB_1) is significantly smaller than the optimal number of stations, then both the versatile bounds and the lack of dominance rules are responsible for not being able to find the optimum or to prove it within the computation time available. However, the additional backward planning of EUREKA sometimes quickly finds optimal solutions where the forward planning fails. This considerably increases the number of proven optima found by EUREKA (Scholl and Klein, 1999).

Scholl and Klein (1997) developed Simple Assembly Line Balancing Optimization Method (SALOME). SALOME is a multiple solution method that performs bidirectional search. The tasks are relabelled and both forward and backward planning is performed simultaneously. A local lower bound method is used in each node to dynamically decide on the planning direction. The branching scheme used is station-oriented. SALOME integrates and improves the most promising components of FABLE and EUREKA. Furthermore, it uses some additional bounding and dominance rules. This approach clearly outperforms FABLE and EUREKA. Though SALOME contains no initial heuristic, its flexible bidirectional enumeration scheme aiming at finding very good solutions as early as possible leads to good and often optimal solutions in a very short time. In SALOME, small branch-and-bound trees are constructed due to the dynamic rule controlling the bidirectional selection of stations. The construction of small branch-and-bound trees allow the finding of optimal solutions in time for many cases (Scholl and Klein, 1999).

Sprecher (1999) has also developed a branch-and-bound algorithm that is called Adapted General Sequencing Algorithm (AGSA). The algorithm proposed relies on the precedence-tree-guided enumeration scheme introduced for dealing with a broad class of resource-constrained project scheduling problems. The Type I problem is reformulated as a resource-constrained project scheduling problem with a single renewable resource whose availability varies with time. The general

enumeration scheme ranked among the most powerful algorithms for solving the single- and multi-mode resource-constrained project scheduling problem is then used. Only minor adaptations are made to obtain an efficient assembly line balancing procedure. Dominance rules proposed by Jackson (1956) and Johnson (1988) are generalized. The computational complexities of some bounds are reduced while the effects of others are strengthened. The computational results with FABLE, EUREKA and SALOME show that the algorithm is capable of competing with the best algorithms currently available for solving the Type I problem. Actually, the results provided show that AGSA clearly outperforms even SALOME.

Type II Problem

The Type II problem which can be restated as minimization of the cycle time C for the given number of workstations n , can be considered as a feasibility problem, that is finding a feasible task assignment to m stations for a given cycle time C or ascertaining that none exists. Hence, the Type II problem can be solved by successively applying Type I solution procedures to instances of feasibility problem with m stations and various trial cycle times.

Helgeson and Birnie (1961) originally proposed solving Type II problems as a sequence of Type I problems. Other iterative methods have also been suggested by other studies. Among these, Hackman, Magazine and Wee's (1989)'s study where they have proposed tighter bounds on the optimal solution for the Type II problem should be noted. By embedding tighter bounds into their heuristic procedure for the Type I problem, they have developed a procedure for the Type II problem.

Direct methods solving Type II problems have been also proposed that are quite a few include Scholl (1994), Klein and Scholl (1996) and Uğurdağ, Rachamadugu and Papachristou (1997) are overviewed below.

Scholl and Klein (1996) summarizes Scholl's (1994) specialized branch-and-bound for the Type II problem as follows. A tabu search strategy is used to

determine good initial upper bounds. Branching is performed on a depth-first-search basis by assigning a single task to a workstation in each step. Priority rules control the choice of the task-station-combinations. The algorithm does not consider the workstations as if they are in a fixed order. Structural properties of the Type II problem have been used to exploit different ways of calculating the lower bounds. It is found that the most effective bounding method is based on minimal idle times in every station, which is determined by solving particular knapsack problems. Furthermore, the algorithm has made intensive use of dominance and reduction rules.

Klein and Scholl (1996) have presented an adaptation of their SALOME to the Type II problem and they have named it SALOME-2. This is also a branch-and-bound method that directly solves the Type II problem. They have used a new enumeration technique called *Local Lower Bound Method*, which is complemented by a number of bounding and dominance rules. The computational results indicate that this new procedure is very efficient.

Uğurdağ, Rachamadugu and Papachristou (1997) have proposed a two-stage heuristic procedure which also directly solves the Type II problem. Their approach is based on the integer formulation of the problem. Thus, they have the feature that the vertices of the polytope for the feasible space are integers. Hence, each vertex of the polytope constitutes a feasible solution to the Type II problem. The first stage of their procedure, which is based on a heuristic procedure they have developed, provides an initial solution to the problem. The second stage of their procedure improves the initial solution using a simplex like algorithm. Their procedure, besides minimizing the cycle time also smooths out the workload among the stations. Their computational studies have also showed that their procedure performed well and provided near optimal solutions to the Type II problem.

2.4.2. Multi/Mixed-Model Assembly Line Balancing

The solution procedures developed for Multi/Mixed-Model Assembly Line Balancing (Mu/MiALB) when compared to Single-Model Line Balancing are very few. Fokkert and de Kok (1997) state that most of the studies in this area date back before 1980s. They foresee a renewed interest in Mu/MiALB due to the evolution of changeover problems in mass assembly, which has gained more and more, an organizational character instead of a technical one. The solution procedures developed for Mixed-Model Assembly Line Balancing (MiALB) and Multi-Model Assembly Line Balancing (MuALB) are discussed separately in the following sections.

2.4.2.1. Mixed-Model Assembly Line Balancing

The SALB and MiALB differ in the precedence constraints. In SALB, only one precedence diagram is given. However in MiALB every model has its own precedence diagram and a balance may not violate any of these orderings. The MiALB is transformed into SALB in almost all the papers that deal with MiALB. Fokkert and de Kok (1997) present a review of the literature on MiALB and MuALB and compare different balancing heuristics on their performance.

The mixed-model assembly lines (implicitly) assume that both changeover times and resulting changeover costs are negligible. In most cases, these changeover times can take only the idle time of the workstation into account (Fokkert and de Kok, 1997). On a mixed-model assembly line, the lot sizes are usually one or very small.

There are mainly two methods used in transforming MiALB problem into SALB problem. The first approach combines the precedence diagram of the different models into a single so-called *combined precedence diagram*. The second method uses *adjusted task times*. Since the adjusted task times method is appropriate only when different models have the same precedence diagram but

different task times, the combined precedence diagram is more widely used in the literature. Both methods are briefly discussed below.

Combined Precedence Diagram Methods

Macaskill (1972) gives the formal description of combining m single-models into a single combined precedence diagram. This procedure is as follows: Precedence diagram of model i is represented by a graph $G_i=(V_i, A_i)$, where V_i is the set of tasks of model i and A_i is the set of precedence relations. The combined precedence diagram is represented by the graph $G=(V, A)$, where $V=\cup_i V_i$ and $A=\cup_i A_i \setminus \{\text{redundant arcs}\}$. An arc (i,j) is redundant if there exists another path from i to j in G . The processing time of $i \in V$ is equal to the total time required for the processing of this task in a given model mix. The model mix defines the number of units to be produced from each model during a shift duration of T . Figure 2.5 illustrates the above procedure. In this figure the numbers above each node represent the task time of the corresponding task. Note that the redundant arc is indicated with dotted line.

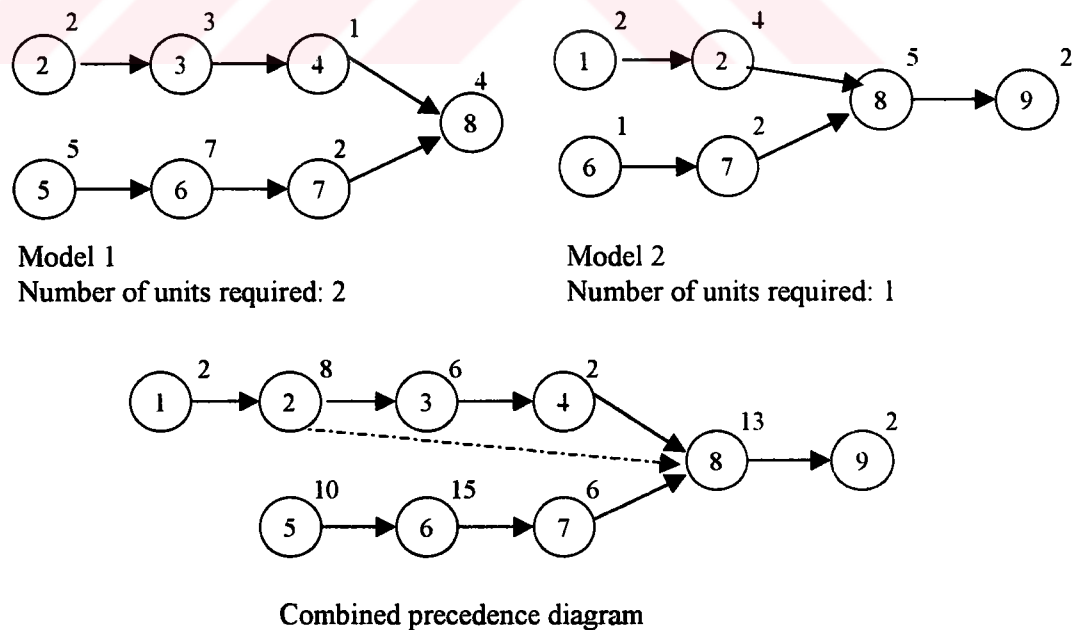


Figure 2.5 A combined precedence diagram composed of two models

The balancing of the mixed-model line using the combined precedence diagram approach is similar to the balancing of a single-model assembly line. The only difference is that in the former case the tasks are assigned to stations on T basis, instead of cycle time, C, basis.

Thomopoulos (1967) and Macaskill (1972) proposed methods based on the combined precedence diagram. They applied heuristics developed for SALB to their combined precedence diagrams to minimize the number of stations.

Fokkert and de Kok (1997) also proposed a method using the combined precedence diagram approach. They modified the branch-and-bound algorithm developed by Berger, Bourjolly, and Laporte (1992) for MuALB, so that lot sizes are one and full enumeration is made to find the optimum. Some further detail of the study conducted by Berger, Bourjolly, and Laporte (1992) is presented in the next section. They also conducted an experimental study where they compared their results with Berger, Bourjolly, and Laporte (1992), Thomopoulos (1967) and Macaskill (1972).

Fokkert and de Kok (1997) summarize the advantages and disadvantages of the combined precedence diagram approach as follows. An advantage of this method is that every repetition of a task is carried out by the same workstation. Thus, this results in minimum learning costs. A disadvantage of this method is related to the balancing on shift basis. Another model mix can lead to another balance and this might create some confusion on the shop floor. Another disadvantage of this method is that it might lead to unequal distribution of the total work content of single-models among the workstations. Thomopoulos (1970) proposes a heuristic that aims to evenly distribute the work content of the single-models among the workstations given a balance found by the combined precedence diagram approach.

Adjusted Task Times Methods

The second method that is used to transform the MiALB problem into SALB problem, determines average task times for the tasks that are required by more than one model (Johnson, 1983). The average task time of task i , t_i , is calculated with the following formula:

$$t_i = \sum_k f_k \cdot t_i^k,$$

where t_i^k , is the task time for task i on model k and f_k is the relative frequency of model k .

Fokkert and de Kok (1997) summarize the advantages and disadvantages of the adjusted task times procedure as follows. An advantage of this method is that the balancing procedure is based on the cycle time C , instead of the shift base in the combined precedence diagram method. A disadvantage of this method is that there is no method that determines the sequence in which the models are produced. Another disadvantage is that this method does not take the eventually different precedence diagrams of the models into account. Thus this method can be only useful when all models have the same precedence diagram but their task times are different.

2.4.2.2. Multi-Model Assembly Line Balancing

MiALB and MuALB differ from each other by means of lot sizes. Multi-Model Assembly Lines have lot sizes greater than one. Thus, they have production costs which include changeover costs. Hence, Buxey, Slack and Wild (1973) state the objective of MuALB as minimization of production costs.

Wild (1972) suggests balancing of multi-model assembly lines with the solution procedures developed for SALB and MiALB. He proposes MiALB solution procedures when the lot sizes are small and when it is important that every repetition of a task is carried out by the same workstation. If the lot sizes are large then he proposes successive application of SALB solution procedures. This procedure suggests balancing of the multi-model line for the model that has the largest frequency and then balancing the remaining models such that similar tasks

are allocated to the same workstation, in order to obtain minimum changeover activities.

Chakravarty and Shtub (1985), propose a MuALB method that considers labor, set-up and inventory costs. They assume the models that are produced in batches can be transported to the following station in their entirety. By placing buffers between two adjacent workstations, they allow the batch sizes to vary between two workstation. They deal with the problem of combining line balancing with lot sizing in a multi product environment. They apply the multi-echelon production/inventory systems and develop algorithms to determine the number of workstations required along the line, tasks assigned to each workstation and the cycle time. They use the combined precedence diagram approach to transform their MuALB problem into SALB problem.

Berger, Bourjolly, and Laporte (1992) describe a branch-and-bound algorithm for MuALB that is also based on the combined precedence diagram approach. Their objective is to minimize the number of stations necessary to manufacture all the products so that tasks are performed in proper order and so that all tasks assigned to any given station can be executed within the prescribed time frame. They present a new branch-and-bound approach, which makes a depth-first search. The nodes in level i of the search tree correspond to a maximal assignment of tasks to the i th workstation. An assignment is maximal if it cannot be extended with another task without exceeding the cycle time. In each node, a lower bound and an upper bound for the number of stations is determined. Their study contains two main improvements, the first one is an improved lower bounding procedure and the second one is a more powerful partitioning scheme. The proposed algorithm can be used either as a heuristic (in its truncated version) or as an optimum seeking procedure (in its full version). They have defined multi products having only one possible order in which tasks can be executed. Moreover, they claim that if two products have k tasks in common, then these are the first k tasks that are required for the production of both products. This latter restriction, results in a combined precedence diagram that is a forest. But actually this restriction may not be required for most of the products produced in multi product environments.

CHAPTER 3

THE PROPOSED APPROACH

An assembly line is dedicated to a family of similar products. In today's industry, assembly lines devoted to a single product have nearly diminished. This is especially true for the automobile and home appliances' assembly. Due to the increased global competitiveness, companies in the industry have achieved flexibility via reducing their batch sizes and increasing their product varieties. The batch sizes of the various different models are small but not actually equal to one. Hence, especially in Turkey, the family of products under consideration are assembled on *Multi-Model Assembly Lines* rather than *Mixed-Model Assembly Lines*. Therefore, the motivation of our study for working on multi-model assembly lines comes from the industry. Batch sizing and batch sequencing are also relevant issues to the multi-model assembly line environment under consideration. However the batch sizing and batch sequencing issues are beyond the scope of this study.

A trivial approach to Multi-Model Assembly Line Balancing (MuMLB) is balancing of the line for each model separately. This approach might allot different number of stations to different models and might yield little number of workstations usage. Thus, low balance delays might be achieved for each model. Milas (1990) defines a good balance delay as a balance delay that is about ten percent or lower. However, excessive amounts of idle time can be realized in stations at the end of the line for the models in this approach, which do not make use of these stations. A more serious issue is that a certain task may be assigned to different stations in different models. When the models have similar work

contents, it might be possible to ensure that a particular task is not widely scattered over the stations for different models. However, for example, if the whole set of tasks in one model i comprise the second half of the tasks in model j , then the scattering would be intolerable (Macaskill, 1972). This scattering of the task to different workstations is not desirable since considerable learning curve effect might be realized along the line resulting in a decrease in the effectiveness of the line. A final drawback of this approach is related to the setups. Switching from one model to another might require shutting down the whole line resulting with long setup times and setup costs.

Mixed-model assembly line balancing (MiMLB) is another approach that can be used in balancing multi-model assembly lines. For a given model mix, this method would find a unique balance and would ensure that a given task is assigned to the same workstation for all models. Hence, every repetition of the same task will be made on the same workstation. Therefore, compared to the former approach, the learning curve effect would be negligible here. In this approach, during the shift from one model to another, the setup times and setup costs are negligible. This approach when applied to a multi-model line has some drawbacks. The first of these drawbacks is related to the unique balance established for all models. In this approach, the same number of workstations are used for all models and this is usually more than the number of workstations used in the former approach. The second drawback is also related to the unique balance established for all models. If the models are very different in terms of their work contents, high balance delays might be realized. Mixed-model lines use sequencing as a tool to ensure that each workstation realize small idle times. However although a sequencing tool as above is used (in the multi-model assembly lines), high balance delays can be realized since the batch sizes are greater than one. A workstation that has a certain amount of idle time will be idle for all the batch which will be hard to compensate.

The above presented two approaches that are applicable to multi-model assembly lines have their advantages and disadvantages on their own. Approaches specially designed for to multi-model assembly lines might be more appropriate.

However, the studies available in the literature in MuMLB as summarized in Chapter two, have their restrictive assumptions on their own and are actually not applicable to most of the real life industry cases. Among the available studies, the suggestion of Wild (1972) might be a good starting point for the development of an approach that is applicable to most of the real life cases.

The aim of multi-model line balancing should be the minimization of total production costs (Buxey, Slack and Wild, 1973). Here, the additional factor of setup costs must be also incorporated. When the batch sizes of the models are very large then the successive application of the SMLB is reasonable to minimize the total production costs. When the batch sizes are very small then order to minimize total production costs it is reasonable to apply MiMLB approaches. For instances where the batch sizes are of intermediate size, these two approaches are not appropriate for MuMLB as presented in the earlier discussions. In these instances, the reallocation of the parts and equipment to workstations make up the main portion of the setup cost in the total production cost. In order to reduce the total production costs, the number of stations and location of equipment should be kept constant whenever possible. Moreover, tasks that are common in more than one model should always be performed at the same workstation (Buxey, Slack and Wild, 1973). In such instances such as these Wild (1972) suggests balancing of the line for the most popular model and then adjusting this basic arrangement methods for the other models by empirical. When the resulting balances are unsatisfactory, the model with the second highest production volume is suggested to be used and the same steps are repeated, and so on.

Wild's (1972) suggestion sounds reasonable but it is difficult to apply. The first reason is the ambiguity in usage of empirical methods in adjusting the basic arrangement for each model. The second reason is related to the selection of the model. If the models in the product family are very different in terms of tasks performed or in terms of their work contents, then selecting the most popular model will result in unsatisfactory results. Therefore, it is necessary to define other means for the selection of the model according to which the basic arrangement will be determined.

The purpose of our study is to propose and evaluate a formal algorithm, which makes use of Wild's (1972) suggestion. In our approach, it is desired to perform the tasks that are common in all models and tasks common in most frequent models in the same workstation for all models. It is also desired to perform tasks that must be done at a specific station for all models. Such *must-do tasks* can be related to equipment constraints like hard-tooled robot workstations, which are permanently placed on the line. Removal or repositioning of such equipment is quite expensive and therefore should be avoided in practice. The same equipment constraint is valid for pallet transfer stations, crossovers and elevation changes in conveyors (Milas, 1990).

3.1. Basics

Table 3.1 includes the notations and their descriptions used.

Table 3.1 Notation for multi-model assembly line balancing

Task index: $i= 0, 1, \dots, M, M+1$
Station index: $j= 1, \dots, N$
Model index: $k= 1, \dots, K$
Planning horizon: T
Demand for model k over T : D_k
Total demand for all K models over T : D
Total work content for model k : TWC_k
Cycle time: C
Lower bound and upper bound: LB and UB
Number of stations on the line N
Number of stations for model k N_k
Set of tasks of model k : V_k
Set of precedence relations of model k : A_k
Precedence relation/graph/ network of model k : $G_k= (V_k, A_k)$
Combined precedence diagram of k models: $G= (V, A)$
Set of predecessors of task i in model k : $P_k(i)$
Set of immediate predecessors of task i in model k : $P_k^*(i)$
Set of successors of task i in model k : $S_k(i)$
Set of immediate successors of task i in model k : $S_k^*(i)$
Task time of task i in model k : t_i^k
Set of tasks that are assigned to station j : $I(j)$
The station that task i is assigned to: $J(i)$

In the proposed approach, the multi-model line envisaged is follows: K different models from a family product are to be assembled on the line under consideration for a planning horizon T. The static demands for all K models are known and add up to D for the given planning horizon. The line is to be balanced with the multi-model line balancing approach.

Our assumptions are stated as follows:

- *The precedence diagrams of the individual models in the product family under consideration are not conflicting.* Therefore, the combined precedence diagram is acyclic.
- *The common tasks of models have the same task times.* This means that irrespective of the model, a task has the same processing time on any model in each instance (Fokkert and de Kok, 1997). Therefore the task time of task i in model k (t_i^k) is denoted by only the task index, as t_i .
- *The cycle time, C, is the same for all models, and satisfies the following condition: $C \geq \max_i \{t_i\}$.*
- *The assembly sequence of the models on the line and their batch sizes are predetermined.*
- *The assembly line is constructed for a certain number of workstations.* However, a model may require and use only some -but not all- of the stations making up the line. To facilitate this, workstations can be opened up or closed according to the requirements of the models that will be processed. However, only the stations that are assigned the 'fixed tasks' cannot be closed. The workers of the closed stations are assumed to be kept busy in activities like maintenance, cleaning, preprocessing, training, etc. Moreover, the opened or closed stations can be anywhere along the line, not necessarily at the beginning or at the end.

- *The demand is deterministic and static.* For the planning horizon T the demand for all models is known and is deterministic.

3.2. Multi-Model Line Balancing Approach

Our approach for multi-model line balancing consists of three main stages:

1. Construction of the base model
2. Balancing of the line for the base model
3. Balancing of the line separately for each individual model.

3.2.1. Base Model Construction

In our approach, it is desired to perform some of the tasks in the same station for all models whenever possible. The remaining tasks are allowed to be performed in different stations for different models. Therefore, a set of tasks that are to be performed on the same station for each model need to be determined. This set of tasks is called the *fixed tasks set* and they consist of mainly from three different sets.

In our approach, it is desired to construct a basic arrangement called the *base line*. The cycle time for the base line C is given using the total demand D and this cycle time C is valid for all models. Each model will flow over this base line. Each model's fixed tasks will be performed on the predefined stations. Therefore, before deciding on the balance for each model, the stations of the fixed tasks need to be determined. For this purpose, a *base model* is defined using the K models and the defined fixed task set.

In the proposed approach, first a base model is constructed, and then the line is balanced for the base model so as to obtain the line design that may undergo some minor changes later -like closing/opening up some stations and adding/deleting tasks to/from stations- during model changeovers.

In the construction of the base model, two issues are to be resolved. The first issue is the selection of a single model among the models, while the second issue is the determination of the fixed tasks set.

Model Selection

Three selection alternatives can be used in model selection. The first is to select the model with the highest demand (Wild, 1972). The second alternative is to select the model with the highest work content. The third alternative is to select the model that is representative of the family of products under consideration. For this purpose, a similarity measure or index needs to be defined and the model with the highest similarity value can then be selected.

In Prenting and Thomopoulos (1974), a similarity index based on the task times of different models is described. Since we assumed that the task time of a common task is the same in all models, this index may not be useful in our case. We therefore propose another similarity index that is based on the precedence diagrams of the models. Precedence relations of model k is represented by a graph $G_k=(V_k, A_k)$, where V_k is the set of tasks of model k and A_k is the set of precedence relations. The combined precedence diagram is represented by the graph $G=(V, A)$, where $V=\cup_k V_k$ and $A=\cup_k A_k \setminus \{redundant\ arcs\}$. A similarity index value is computed for each model with respect to the combined precedence diagram since we are interested in finding the model that is most representative of the family of products. This similarity index can be computed separately in terms of both the number of tasks and number of precedence relations. Below their determinations and interpretations are given.

- Similarity index for model k in terms of the number of tasks is denoted by $SI_t(k)$, where $SI_t(k)=|V_k \cap V|/|V|$, where $|V|$ denotes the cardinality of the set N . Selecting a model based on this similarity index is equivalent to selecting the model with the maximum number of tasks.

- Similarity index for model k in terms of the number of arcs is denoted by $SI_a(k)$, where $SI_a(k) = |A_k \cap A| / |A|$. Selecting a model based on this similarity index is equivalent to selecting the model with the maximum number of precedence relations.

The criterion used in the selection of the base model is very important in our approach in that it may affect the latter stages of our approach. Their effects are explored through the experiments and discussed in Chapter 5.

Fixed Tasks Set

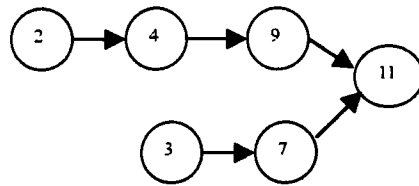
The fixed tasks set is composed of three subsets:

- 1) *The Common Tasks Set (CT₁)* is composed of tasks that are common in all models. Therefore, it is determined as $CT_1 = N_1 \cap \dots \cap N_{k-1} \cap N_k$.
- 2) *The Most Frequent Models' Common Tasks Set (CT₂)* is composed of tasks that are common in most frequent models. First, the most frequent models are determined through the analysis of demands of individual models in the planning horizon. Let $N_{(1)}, N_{(2)}, \dots, N_{(p)}$ represent the tasks of p models with the highest demands given in a non-increasing order. Then $CT_2 = N_{(1)} \cap N_{(2)} \cap \dots \cap N_{(p-1)} \cap N_{(p)}$.
- 3) *The Must-Do Tasks Set (MT)* is defined by tasks that have to be processed at a specific workstation mainly due to machinery constraints. Hence, $MT = \{\text{Tasks that have to be performed at a specific workstation}\}$.

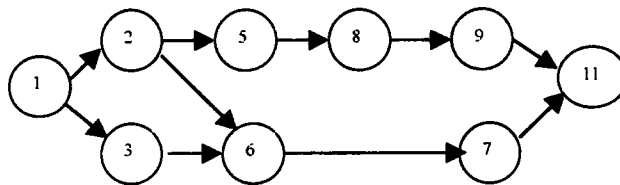
After these three subsets are determined, the *Fixed Tasks Set (FT)* is established as a union of these three sets: $FT = MT \cup CT_1 \cup CT_2$. The size of the fixed tasks set may affect the results which is also explored in the experimentation presented in Chapter 4.

Consider a product family with three models. Let Figure 3.1 represent the precedence diagrams of models and Figure 3.2 represent the combined precedence diagram. Table 3.2 contains the demand and task times, while Table 3.3 gives the similarity indices of models. Assume that task number 4 is performed by a special

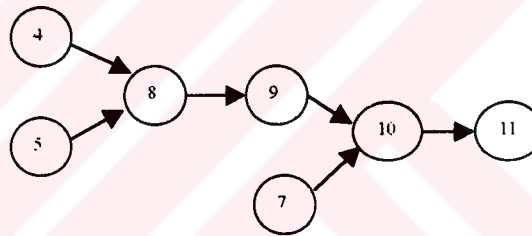
industrial robot. Since it is very expensive to change the station of the robot for different models, task 4 is performed in the same station for all models.



Model 1



Model 2



Model 3

Figure 3.1 Precedence diagrams of models 1, 2 and 3

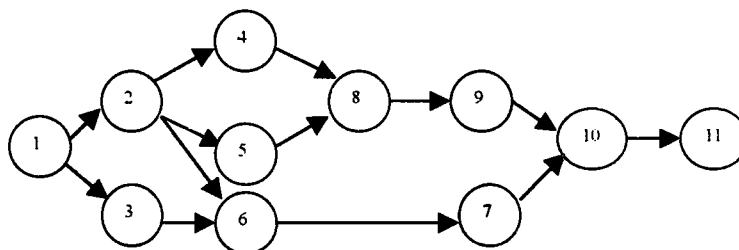


Figure 3.2 Combined precedence diagrams of models 1, 2 and 3

Table 3.2 Demand and task times of models

Model (k)	1	2	3
Demand (D_k)	9	8	3
Task i	t_i^1	t_i^2	t_i^3
1	0	5	0
2	35	35	0
3	25	25	0
4	60	0	60
5	0	30	30
6	0	10	0
7	60	60	60
8	0	25	25
9	35	35	35
10	0	0	70
11	30	30	30
Σt_{ik}	245	255	310

Table 3.3 Similarity indices of the models

Task, i	$t_{ik}=t_i$	Models, k		
		1	2	3
1	5		✓	
2	35	✓	✓	
3	25	✓	✓	
4	60	✓		✓
5	30		✓	✓
6	10		✓	
7	60	✓	✓	✓
8	25		✓	✓
9	35	✓	✓	✓
10	70			✓
11	30	✓	✓	✓
Demand	D_k	9	8	3
TWC_k	Σt_i^k	245	255	310
Similarity index, $SI_i(k)$		5/11	9/11	7/11
Similarity index, $SI_a(k)$		5/13	10/13	6/13

According to the given alternatives for model selection in this example,

- Model 1 is selected if criterion is the highest demand.
- Model 3 is selected if criterion is the highest work content.
- Model 2 is selected if criterion is similarity index in terms of number of tasks or similarity index in terms of number of arcs.

Assume that Model 2 is selected according to one of the presented criteria. Then our fixed tasks set can be established as follows:

1) *The Common Tasks Set (CT₁)*

$$CT_1 = N_1 \cap N_2 \cap N_3 = \{7, 9, 11\}.$$

2) *The Most Frequent Models' Common Tasks Set (CT₂)*

Models 1 and 2 are the most frequent models according to the given demand figures.

$$CT_2 = N_{(1)} \cap N_{(2)} = \{2, 3, 7, 9, 11\}$$

3) *The Must-Do Tasks Set (MT)*

$$MT = \{4\}.$$

Thus *Fixed Task Set (FT)* is established as:

$$FT = MT \cup CT_1 \cup CT_2 = \{2, 3, 4, 7, 9, 11\}.$$

Construction of the Base Model

After a model is selected and the set of fixed tasks is determined, the method described for constructing combined precedence diagrams is used to establish the base model. The construction of the base model is explained in detail in Appendix A.

The base model constructed may not correspond to any of the models within the product family. Actually, the base model can be considered an augmented version of the selected model with the addition of the tasks in the fixed task set.

Figure 3.3 presents the base model for the given example. The dashed node and arcs correspond respectively to the task and precedence relations added to the selected model for the given fixed tasks set. For this example, the base model constructed does not resemble to any one of the actual models.

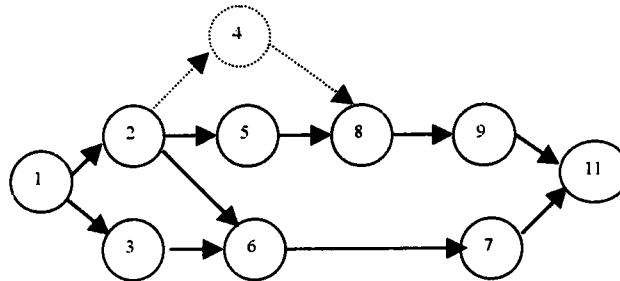


Figure 3.3 Base Model

3.2.2 Balancing for the Base Model

After the construction of the base model, the second step in our approach is to balance the line for the base model. The aim of this step is to assign the fixed tasks to their corresponding permanent stations.

For this purpose, single-model line balancing procedures might be used. The balance obtained for the base model has to be ensured that it is also feasible for all models. Infeasibility due to the cycle time may arise for individual models. For instance, consider a workstation where two fixed tasks (k and l) are assigned. Assume there exists a task (m), which is on a path between (k) and (l) in the precedence diagram of a model. This means that, (m) is a successor of task (k) and a predecessor of task (l). Due to this precedence relation, task (m) has to be performed in the same station together with tasks (k) and (l). If the sum of task times of k, l, and m exceeds the cycle time, infeasibility occurs. This type of infeasibility may occur at any workstation to which a pair of fixed tasks has already been assigned. Therefore, the available single-model line balancing procedures cannot be used directly in this step of our approach.

We decided to modify Hoffman's (1992) EUREKA to solve this partial line balancing problem. There are several reasons for choosing EUREKA. It is a single-solution method, station oriented and simple. It does not employ dominance concepts or relabelling. Thus, we can perform the cycle time feasibility check for every station constructed in the modified EUREKA easily. Another reason for choosing EUREKA is to be able to perform forward and backward planning for a given time frame. As discussed in Chapter 2, the additional backward planning of EUREKA sometimes finds optimal solutions quickly where the forward planning fails. Furthermore, EUREKA applies a heuristic (Hoffmann, 1963) when it fails to find an optimal solution within the given time frame.

The modified *EUREKA* can be summarized in the following steps:

1. Enumerate in forward direction.
2. If there exists no optimal solution after M minutes, enumerate in backward direction.
3. If there exists no optimal solution after M minutes, use Immediate Update First Fit (IUFF) heuristic (Hackman, Magazine, and Wee, 1989).

Heuristic

The heuristic we used, IUFF, involves three steps:

1. Initially assign a numerical score, s_i to each task i .
2. At each iteration update the list of candidate tasks. If the list is empty stop.
3. Assign the candidate task with the largest numerical score on the first feasible station. Go to Step 2.

Among various numerical score functions available in the literature, we used $s_i=t_i$, i.e. the task times are directly used as numerical scores. We choose this

numerical score function, since its effectiveness has been demonstrated by computations in Hackman, Magazine, and Wee (1989) and Berger, Bourjolly, and Laporte (1992).

Lower Bound

Hoffmann (1992) suggests the simple lower bound: $LB_1 = \lceil \sum t_i / C \rceil$. We also use a stronger bound, LB_2 , proposed by Berger, Bourjolly, and Laporte (1992) in our approach. A brief summary for computing LB_2 is also provided in Appendix B. Thus the lower bound used in our search procedure is LB that is the maximum of LB_1 and LB_2 .

Before proceeding with the branch-and-bound procedure, let us determine the lower bound for our base model in our example. Assume that a cycle time of 72 time units is given. Then our lower bounds are as follows:

$$LB_1 = \lceil TWC_2 / C \rceil = \lceil 255 / 72 \rceil = 4$$

$$LB_2 = 4$$

$$LB = \max\{LB_1, LB_2\} = 4$$

Branch-and-Bound

The branch-and-bound procedure used generates a set of tasks assigned to a station. After generating one set of tasks at a station, it continues to generate the following station. While tasks are generated in our procedure, we perform the cycle time feasibility check. If the last generated task i is an element of the fixed tasks set, we check if there is any other task from the fixed tasks set already assigned to the current station s . If there is not any, we proceed with the algorithm, otherwise, for each $j \in FT$ that is assigned to the current station and for the task i , we check the following condition:

$$\sum_{\substack{j \in FT \\ i \in s}} t_j + t_i + \sum_{k \in X} t_k > C$$

where $X=\{j \in FT \text{ and assigned to current station and all tasks in paths between } j \text{ and } i \text{ on the combined precedence diagram}\}$. If the above condition holds then that node is fathomed since no solution can be obtained without violating the cycle time feasibility for the individual models. Otherwise, we proceed with our algorithm as described below.

After stations are generated, the cumulative sum of station slack (idle) time is calculated. If this sum exceeds the theoretical minimum total slack time, then all emanating branches are fathomed; since there can be no solution along this branch which has the theoretical minimum number of stations. The algorithm searches in an orderly manner for an alternative set at this station: If one is found that does not result in excess slack, it goes on to the next station; otherwise it backtracks to the previous station and generates another alternative set of tasks there and moves on. This process continues until all the tasks have been assigned and the theoretical minimum total slack has not been exceeded or all branches have been fathomed. If the latter occurs, this implies that no “theoretical minimum” solution exists; therefore the algorithm increases the theoretical minimum slack time by an amount equal to the cycle time. This is equivalent to adding one more station to the lower bound on the number stations. This procedure is repeated until an optimal solution is found. A step-by-step description of the whole algorithm is presented in Appendix C.

When the lower bound is increased, EUREKA starts generating the branch-and-bound tree from scratch. EUREKA systematically rebuilds the branch-and-bound tree of the previous iteration and continues from that point on. Instead of starting from scratch each time, we can make use of the information we obtain from the previous branch-and-bound tree, like the fathomed nodes and their respective information. For this purpose, we use the available memory. But, when the available memory is not enough, we save the branch-and-bound tree in disk. By doing so, we have saved a considerable amount of processing time.

For the example base model, the result of our branch-and-bound algorithm is presented in Figure 3.4. The rectangles represent the stations, while the circles

correspond to the tasks assigned to the stations, the black-filled circles being the fixed tasks.

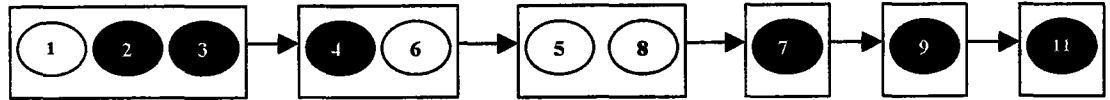


Figure 3.4 Modified EUREKA Solution for the Base Model

In this solution, tasks 9 and 11 are not assigned to the same workstation due to the cycle time constraint. Although the total of task times of tasks 9 and 11 (65 time units) for the base model is less than the cycle time (72 time units), they are not assigned to the same workstation. Task 10 is an immediate successor of task 9 and an immediate predecessor of task 11 in the combined precedence diagram. Thus, if task 9 and 11 were assigned to the same workstation, task 10 had to be assigned to the same station also in balancing the line for Model 3. Since the total of these three tasks' task times (135 time units) exceeds the cycle time, we would face the violation of cycle time constraint. This would occur if the available single-model line balancing solution procedures had been directly used. The optimal solution for the base model balance is found to consist of five workstations, if the cycle time feasibility for all models is not taken into account. The modified EUREKA we propose can handle such infeasibility instances.

3.2.3. Balances for Individual Models

In our approach, this problem of assigning the remaining tasks of individual models given the fixed tasks assignments is called *Partial Type I* problem. After determining the number of stations for each model, the maximum number of stations required by one of them, N_{max} , determines the length of the base line in terms of number of workstations. The base line is constructed according to this maximum number of workstations. However, each model is allowed to operate on their own number of stations that is determined by the solving of its own Partial Type I problem. If the Partial Type I problem has found

N_i as the number of stations required for model i , exactly N_i stations will be used during the assembling of model i . In order to facilitate this, the closing of the $N_{\max} - N_i$ stations are needed during the production of model i . This opening and closing of the workstations according to the needs of different models permits our approach to work almost with the minimum number of stations similar, to the case where balances for each model were determined individually using the SMLB approach. Having a fixed tasks set and freezing the places of the fixed tasks set allow us to perform common and must do tasks in the same station for all models similar to the MiMLB approach.

We propose another branch-and-bound approach for solving the Partial Type I problem. The upper and lower bounding schemes (which make use of the partial assignments) are defined. Some further definitions are made below to determine the bounds and perform the balancing of individual models in a partial manner.

Index Set of Stations, $j=1, \dots, n$. This is the index set of stations to which fixed tasks are assigned. They do not correspond to the actual station numbers to which assignments of the fixed tasks are made.

In our example, the index set of stations consists of 1, 2, 3, 4, and 5. Although the base model's balance includes six stations, only five of them include the fixed tasks.

Set of Tasks, $M=\{1,2, \dots, m\}$. This is the set of tasks of the model for which the Partial Type I Problem is solved.

Let us solve the Partial Type I Problem for Model 3 in our example. The set of tasks for Model 3 are $M=\{4, 5, 7, 8, 9, 10, 11\}$

Set of In-Tasks, T_j , consists of the fixed tasks assigned to station j in the balance of the base model plus all tasks along all paths between any two fixed tasks assigned to station j .

For our example, the in-tasks for each station are as follows:

$$T_1 = \{2, 3\}$$

$$T_2 = \{4\}$$

$$T_3 = \{7\}$$

$$T_4 = \{9\}$$

$$T_5 = \{11\}$$

Set of Between-Tasks, S_j^p , consists of tasks $i \in M \cup T_j$, which is, at least, on one of the paths between tasks k and l in the precedence diagram such that k is assigned to station $j-p$ and task l is assigned to station j . Thus task i can be assigned to any station between station $j-p$ and p . Although p can be between 1 and $n-1$, task i is assigned to the set S_j^p with the lowest possible value for the index p .

For our example, the procedure for forming the set of Between-Tasks starts with the determination of the tasks which are not yet assigned to stations: $i \in M \cup T_j = \{5, 8, 10\}$.

- For $p=1$,
 - Model 3 has no tasks that are to be between stations 1 and 2.
 - Model 3 has no tasks that are to be between stations 2 and 3.
 - Model 3 has no tasks that are to be between stations 3 and 4.
 - Task 10 of model 3 is to be between stations 4 and 5. $S_5^1 = \{10\}$.

- For $p=2$,
 - Model 3 has no tasks that are to be between stations 1 and 3.
 - Task 8 of model 3 is to be between stations 2 and 4. $S_4^2 = \{8\}$.
 - Model 3 has no tasks that are to be between stations 3 and 5.

- For $p=3$,
 - Task 5 of model 3 is to be between stations 1 and 4. $S_4^3 = \{5\}$.

Thus, the unassigned tasks of model 3 have been classified now.

Set of To-Tasks, U_j , consists of tasks $i \in M \setminus [((\cup_j T_j) \cup (\cup_p \cup_j S_j^P))]$ which have an immediate successor in station j . Therefore, these tasks have to be performed either in station j or in any station prior to station j . Task i should be assigned to a set U_j with the lowest possible value for index j .

There is no task in the set of To-Tasks for Model 3. But, if we were solving the Partial Type I problem for Model 1, task 1 would be a member of U_1 , since all of its successors are already assigned to station 1. Since our example does not provide any instance for the following definitions, they are presented without referring to our example.

Set of From-Tasks, V_j , consists of tasks $i \in M \setminus [((\cup_j T_j) \cup (\cup_p \cup_j S_j^P) \cup (\cup_j U_j))]$ which have an immediate predecessor in station j . Therefore, these tasks can be performed in station j or in any station following station j . Task i should be classified into a set V_j with the lowest possible value for index j .

Set of Independent-Tasks, consists of tasks $i \in I = M \setminus [((\cup_j T_j) \cup (\cup_p \cup_j S_j^P) \cup (\cup_j U_j) \cup (\cup_j V_j))]$. The partial assignment of the fixed tasks set does not bring any restrictions regarding the station where the independent tasks can be assigned. Therefore, they can be assigned to any workstation.

Remaining Time in Station j , R_j , is defined as: $R_j = C - \sum_{i \in T_j} t_i$, that is, the time that remains on station j from the fixed tasks and all tasks along all paths between any two fixed tasks assigned to station j . The remaining times in workstations for our example are as follows:

$$R_1 = 72 - 60 = 12 \text{ time units,}$$

$$R_2 = 72 - 60 = 12 \text{ time units,}$$

$$R_3 = 72 - 60 = 12 \text{ time units,}$$

$$R_4 = 72 - 35 = 37 \text{ time units,}$$

$$R_5 = 72 - 30 = 42 \text{ time units.}$$

Lower Bound

In addition to the lower bounds LB_1 and LB_2 , another lower bound, LB_3 , is defined for the Partial Type I problem. Thus, the lower bound is the maximum of these three lower bounds.

As calculated in the previous section the lower bounds $LB_1 = LB_2 = 4$ in our example.

The lower bound LB_3 is calculated as follows:

1. $A=n$, current number of workstations.

2. Number of stations required prior to station 1, B, is

$$B = \begin{cases} 0, & \text{if } \sum_{i \in U_1} t_i < R_1 \\ \lceil ((\sum_{i \in U_1} t_i) - R_1)/C \rceil, & \text{otherwise} \end{cases}$$

3. Number of stations required following station n, D, is

$$D = \begin{cases} 0, & \text{if } \sum_{i \in V_n} t_i < R_n \\ \lceil ((\sum_{i \in V_n} t_i) - R_n)/C \rceil, & \text{otherwise} \end{cases}$$

4. Number of additional stations required between stations 1 and n, F, is defined below.

$$E = \sum_{j=1}^{n-1} \sum_{i \in V_j} t_i + \sum_{j=2}^n \sum_{i \in U_j} t_i + \sum_{i \in I} t_i + \sum_{p=1}^{n-1} \sum_{j=p+1}^n \sum_{i \in S_p^j} t_i$$

$$F = \begin{cases} 0, & \text{if } E < \sum_{j=1}^n R_j \\ \lceil (E - \sum_{j=1}^n R_j)/C \rceil, & \text{otherwise.} \end{cases}$$

$$LB_3 = A + B + D + F.$$

We can define $n*(n-1)/2$ such bounds for the number of additional stations required for any pair of stations. If the number of additional stations is not conflicting, we may improve this bound. For instance, if we need a station between stations 1 and 3 and another station between stations 4 and 6, the bound is increased definitely by two. On the other hand, if we need a station between 1 and 3 and another between 3 and 5, we need one or two additional stations. In that case, we have to determine if we need two more stations; or we may simply assume that we need one more station for the lower bound. We should evaluate the bound given above for all pairs, because we cannot detect if we consider only the bound between stations 1 and n. That is the extra station needed can be compensated by the slacks in prior and following stations. In this study, unfortunately, we used only the bound between the first and last station.

In our example, the LB_3 for Model 3 is determined as follows:

$$1) A=5.$$

$$2) B=0 \text{ since the set of To-Tasks to workstation 1 is empty.}$$

$$3) C=0 \text{ since the set of From-Tasks following workstation 5 is empty.}$$

$$4) E=t_5+t_8+t_{10}=125 \text{ time units.}$$

$$\sum_{i=1}^n R_i=115 \text{ time units}$$

$$F=\lceil (125-115)/72 \rceil=1,$$

$$\text{Therefore, } LB_3=5+0+0+1=6 \text{ workstations.}$$

LB_1 and LB_2 do not make use of the partial assignment information. The simplest form of LB_3 is the one presented above and makes use of the partial assignment information. LB_3 can further be improved by applying the LB_2 idea to calculation of B, D and F separately. The LB_2 scheme is presented in Appendix C. In our example the first and second lower bounds proposed for LB_3 , yield the same lower bound, if the additional number of stations is found using the simple lower bound. However, the second proposal when used with the LB_2 scheme detects additional stations and thus yields a stronger lower bound.

In our experimentation we used the above introduced LB_3 . However, a more efficient Lower Bound, LB_3 , is defined with a similar logic to the one presented above. This lower bound also makes use of the class definitions previously made. This time conditions for opening new stations are defined. Whenever the condition is satisfied, new workstations are opened up immediately. The number of stations to be opened up, when a given condition is satisfied, is determined with an approach similar to that of LB_1 and LB_2 . LB_3 has an initial value that is equal to the number of stations to which fixed tasks are assigned. The number of stations to be opened up depends on the following conditions:

1. *Prior to Station 1*

$$\sum_{i \in U_1} t_i > R_1.$$

2. *Following Station n*

$$\sum_{i \in V_n} t_i > R_n$$

3. *Between Stations k and l*

$$\sum_{p=1}^{l-k-1} \sum_{j=p+k}^l \sum_{i \in S_p} t_i > \sum_{j=k}^l R_j$$

4. *Prior to Station k*

$$\sum_{j=1}^k \sum_{i \in U_j} t_i + \sum_{p=1}^{k-1} \sum_{j=p+1}^k \sum_{i \in S_j^p} t_i > \sum_{j=1}^k R_j$$

5. *Following Station k*

$$\sum_{j=k}^n \sum_{i \in V_j} t_i + \sum_{p=k}^{n-1} \sum_{j=p+1}^k \sum_{i \in S_j^p} t_i > \sum_{j=k}^n R_j$$

6. *Anywhere*

$$\sum_{i \in \bar{T}} t_i > \sum_{j=1}^n R_j$$

Upper Bound

An upper bound is also proposed. This upper bound motivated the lower bound (LB_3) determined above. Then Immediate Update First Fit Heuristic (IUFF) logic is used while tasks are assigned to their corresponding workstations. The starting number of workstations, N , is LB_3 .

The unassigned tasks are assigned to their earliest feasible stations defined by their classes in first half of the workstations. Likewise unassigned tasks are assigned to their latest feasible stations defined by their classes in the second half of the workstations.

While this assignment is made an order is followed. The order corresponds to the class definitions and is as follows:

$$\begin{array}{l}
 U_1 \\
 V_N \\
 S_j^1 \\
 S_j^2 \\
 \vdots \\
 S_j^p \\
 U_2 \\
 V_{N-1} \\
 U_p \\
 V_{N-p} \\
 S_j^{p+1} \\
 S_j^{p+2} \\
 \vdots \\
 S_j^{N-1} \\
 \vdots \\
 U_{p+1} \\
 V_{N-p-1} \\
 \vdots \\
 U_{n-p}, \\
 V_p \\
 \vdots \\
 \vdots \\
 I
 \end{array}$$

Here 'p' turns out to be a decision variable. The class definitions of the tasks are not deterministic. As tasks are assigned to workstations, the class definitions of the tasks that are successors or predecessors of the assigned task are updated.

When this heuristic is applied to Model 3's Partial Type I problem in the example, it proceeds as follows: Our unassigned tasks and their corresponding classes are as follows: $S_5^1=\{10\}$, $S_4^2=\{8\}$, and $S_4^3=\{5\}$. Our candidate list consists of tasks whose precedence relations are satisfied, thus are feasible for assignments $\{5, 10\}$. Task 5 is assigned to the earliest feasible station between stations 1 and 3. Since its task time is greater than remaining times in stations 1, 2 and 3, a new station is opened between 1 and 3, and task 5 is assigned to this new workstation. Let the place of the new station be as illustrated in Figure 3.5.

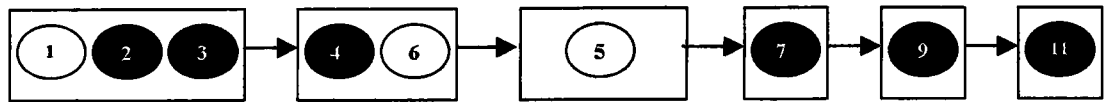


Figure 3.5 Demonstration for the upper bound

Then the class definitions of the unassigned tasks are updated as follows: $S_6^1=\{10\}$, $S_5^2=\{8\}$. Our updated candidate list consists of tasks $\{8, 10\}$. Task 8 is assigned to the earliest possible station between workstations 3 and 5, which is workstation 3. Finally, task 10 is assigned to the latest possible station between stations 5 and 6. Due to its task time requirement, a new station is to be opened between stations 5 and 6.

As a result of this upper bounding scheme, the number of stations for Model 3 turns out to be 7. The lower bounds found out with the first and second proposals for LB_3 are equal to 6 and 7, respectively. The optimal solution found by the following branch-and-bound procedure is also 7. Based on this example only, we see that the upper and lower bounds proposed are strong.

Branch-and-Bound

Given the partial assignments and the class definitions for the unassigned tasks, a branch-and-bound procedure solving Partial Type I problem is proposed. The branch-and-bound procedure used here is also a modification of EUREKA.

Starting with the number of stations determined by the lower bound, it assigns the feasible tasks to the stations. Classes of the unassigned tasks are updated each time a task assignment is made. Our procedure applies enumeration instead of applying the Immediate Update First Fit Heuristic presented above. Forward and backward planning is applied according to classes of the tasks. No labelling or dominance rule is applied.

Theoretical minimum total slack time is used for fathoming branches. As new stations are opened up, the lower bound is increased and the theoretical minimum total slack time is increased by an amount equal to the cycle time.

The results of the branch-and-bound procedure is the same as the one presented in the upper bound. In this step of our approach, when the determined upper bound is equal to the lower bound, the balance yielded by the upper bounding heuristic is therefore optimal.

3.3. Solution Tool Box

In coding the proposed approach BORLAND C has been used. During the coding, advanced data structure elements like linked lists, doubly linked lists, queues and stacks have been used extensively. This usage has facilitated our tree and network searches during different stages of our approach. The codes, sample problems and a readme.doc to guide the procedure are presented in Appendix D.

CHAPTER 4

EXPERIMENTAL ANALYSIS

The proposed approach's performance is compared with Single-Model Line Balancing (SMLB) and Mixed-Model Line Balancing (MiMLB) approaches. An experiment is designed to explore:

- which method results in the best performance in terms of the defined response variables
- which factors have significant effect on the response variables
- are there significant differences between the levels of the significant factors?

4.1. Design of the Experiment

The design of the experiment consists of determining the factors and their levels, the response variables and the generation of problems on which the experimentation will be conducted.

4.1.1. Factors and Factor Levels

Five factors are defined for this experiment. Table 4.1 summarizes the factors and their levels. The first factor determines the number of models assembled on the line. The second and third factors are define the structure of the precedence diagrams. The fourth factor is the model mix and is related to having demand distributions with one dominant model, two dominant models or even distribution. To determine these we have used available problem sets from the

Table 4.1 Factors and their Levels

Factors	Levels		
	1	2	3
Number of Models	5	10	-
% of common tasks	50	80	-
Place of common tasks	Front	Back	-
Model Mix	1-Dom	2-Dom	Even
Cycle Time	Min	Medium	Max

literature (Ding and Cheng, 1993 and Sumichrast and Russel, 1990). Table 4.2 and 4.3 give demand alternatives for 5 models and 10 models, respectively. Among ten alternatives for each we have selected three instances corresponding to one dominant, two dominant and even distributions. In each case, alternatives C, E, and I are used. Here note that in all cases the total demand constant and is equal to 20. The fifth factor is the cycle time. The three levels corresponding for each problem are take from the literature. Section 4.1.3. explains issues regarding the problems and cycle time in detail.

Table 4.2 Problem set M1, D=20, K=5

	D1	D2	D3	D4	D5
A	16	1	1	1	1
B	15	2	1	1	1
C	13	4	1	1	1
D	10	5	2	2	1
E	8	7	2	2	1
F	6	6	5	2	1
G	5	5	5	3	2
H	5	4	4	4	3
I	4	4	4	4	4

Table 4.3 Problem set M2, D=20, K=10

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
A	11	1	1	1	1	1	1	1	1	1
B	10	2	1	1	1	1	1	1	1	1
C	9	3	1	1	1	1	1	1	1	1
D	8	4	1	1	1	1	1	1	1	1
E	7	5	1	1	1	1	1	1	1	1
F	6	5	2	1	1	1	1	1	1	1
G	5	5	3	1	1	1	1	1	1	1
H	4	4	4	2	1	1	1	1	1	1
I	2	2	2	2	2	2	2	2	2	2

4.1.2. Response Variables

Five response variables have been defined to compare the different methods and to measure the effects of the factors:

1. *Balance Delay (BD)*: Balance delay is one of the response variables related to the efficiency of the line. The total balance delay is the total of balance delays for each model which is weighted in proportion to demand ratios (Buxey, Slack and Wild, 1973). The lower the value of this response variable the better it is.
2. *Average Number of Stations (AnST)*: This response variable corresponds to the number of workstations. Since in SMLB and MuMLB balances are constructed for each model individually, they have to be transformed into an aggregated value. The average number of stations is determined by the weighted sum of the individual models' number of stations. The weight the proportion of their demand ratios. The lower the value of this response variable the better it is.
3. *Maximum Number of Stations (XNST)*: For SMLB and MuMLB approaches this response variable gives the number of station for which the line will be built actually. It is determined by looking at the balances of the individual models and taking the maximum value. The lower the value of this response variable the better it is.
4. *Average Number of Setups (ANSE)*: This response variable is used for measuring the number of setups and in a way the learning curve effect. It is an indicator regarding the different number of stations a task is performed in different models. For instance, if a task is assigned to workstations 2 5 3 3 4 in models 1 through 5, the number of setups is $3-1$ (at least one will be performed) = 2. For all tasks this variable is summed and then is divided by the number of tasks to yield the number of different workstations a task is performed. The lower the value of this response variable the better. In its ideal

case this value is zero, where every task is performed in the same workstation for all models, i.e. MiMLB case.

5. **Percentage of Stations (SEP):** This response variable is determined by dividing the Average number of setups by the average number of workstations. Here the aim is to determine whether 2.16 setups are performed over a ten workstation range or 56 workstation range. The lower the value of this variable the better.

4.1.3. Problem Generation

The problem set used in ALB literature is employed. Recently, Scholl and Klein, 1999 and Sprecher, 1999 have used these data sets in their experiments and the URL of the data sets is as follows: <http://www.bwl.tu-darmstadt.de/bwl3/forsch/projekte/alb/index.htm>. What is changing in the individual models, is the place of the common tasks, and the position of the common tasks in the precedence diagrams of the models. For each problem (PROBLEM), ten model instances are generated. The precedence diagram of the problem is used as the combined precedence diagram and models are generated randomly according to the treatments. For each problem a total of forty model instances have been generated. For number of model levels equal to 5 only the first five of the ten models are used.

The acronyms for the factor names are as follows:

- PLACE: Place of common tasks in the precedence diagram
- COMMON: Percentage of common tasks
- MIX: Model mix
- MODELS: Number of models
- C: Cycle Time

4.2. Experiment Design

In order to perform the experiment using the nineteen problems in the literature $19*2*2$ model instances are generated by taking PLACE COMMON factors into account.

For each problem we have defined three MIX and C factors, resulting in 1540 treatments.

Five methods have been used for comparison purposes. The first one is MiMLB. SALOME (Scholl and Klein, 1997) is used in determining the optimal solutions. Combined precedence diagram approach is used in transforming the MiMLB into SMLB. The second approach used is SMLB. SALOME is used for finding the results of the individual models, then results of the response variables are aggregated for 5 and 10 model levels. The next three approaches are the ones we propose. In the first one the selection criterion is the similarity index in terms of the number of tasks. In the second one the model with highest work content is selected. In the third one the most frequent model is selected.

Thus our design is $19*2*2*2*3*3*5$ factorial design. So we made 6840 comparisons. The number of ALB instances solved is 59508.

Two experiments are conducted: The first one is for the average number of setups and the percentage of setups. The second one is for the average and maximum number of stations and the balance delay. In the first analysis the PROBLEMS constitute the replicates. METHOD, C, MIX, COMMON, PLACE, MODEL are the factors. In the second analysis, however, the factors are PROBLEMS, C and METHOD. The other factor levels constitute the replicates for the analysis.

The first analysis is valid since the above performance measures are scaled according to the number of tasks in the problems. However same type of analysis cannot be conducted for the other response variables, i.e. maximum number of

stations, average number of stations and balance delay. These response variables are highly affected by the cycle time, problem and method.

4.3 Analysis of the Experimental Results

In statistical analysis, the results of the experiment “SAS System” software installed on the main frame running on IBM Model RS/6000 is used. The files corresponding to the SAS output of the statistical analysis are long and hence are included in the enclosed diskette hence Appendix D.

Analysis of variance (ANOVA) is performed for the response variables and the Tukey Grouping is used for comparisons of the mean. In ANOVA, the reported factors and reported interactions do not exceed the critical F-ratio value and thus that are significant. In this experiment the level of significance α is chosen as 0.05 for all test.

Two different analysis are employed here. The analysis for the response variables related to setups, i.e. average number of setups per task and the percentage of stations where setups are performed. Table 4.4 presents a summary of ANOVA

- The ANSE (average number of setups) has an R^2 value of 0.635. The utility of the analysis may be accepted. The results of this analysis is as follows:
 - All the main factors except the MODEL MIX are significant.
 - The METHOD employed and MODELS are the most significant effects.
 - Among all way interactions, the METHOD and MODELS two way interaction and the METHOD, MODELS and C three way interactions are significant.

- The SEP (percentage of stations where setups) are performed has an R^2 value of 0.79. The utility of the analysis can be accepted. According to this analysis
 - METHOD has a much more higher significance than the previous case.
 - METHOD and MODELS is significant.

Table 4.4 Summary of ANOVA 1

RESPONSE VARIABLE (R ²)	AnSE (0.63)	SEP (0.79)
SOURCE	P VALUE	P VALUE
METHOD	0.0001	0.0001
PLACE	0.0001	0.0001
COMMON	0.0001	0.0001
C	0.0001	0.0001
MODELS	0.0001	0.0001
METHOD*PLACE	0.0001	0.0001
METHOD*COMMON	0.0001	0.0001
METHOD*C	0.0001	0.0001
METHOD*MODELS	0.0001	0.0001
PLACE*C	0.0004	0.0001
PLACE*MODELS	0.0152	0.0277
COMMON*MODELS	0.0043	0.0099
C*MODELS	0.0001	0.0014
METHOD*C*MODELS	0.0001	
METHOD*PLACE*COMMON		0.0011
PLACE*C*MODELS		0.0002

As a follow up analysis, the means of these response variables are contrasted using Tukey Test. Table 4.5 presents a summary of the Tukey Follow Up Analysis. For both performance measures under consideration the following are observed:

- All METHOD levels are significantly different. The MiMLB has the best results while SMLB has the worst result, as we expected. In MiMLB, all tasks

Table 4.5 Summary of the Tukey Follow Up Analysis 1

METHODS	SINGLE	Mu-SI	Mu-WC	Mu-FQ	MIXED
AnSE	1.176	0.804	0.732	0.079	0.000
SEP	0.151	0.059	0.053	0.009	0.000
MIX	2-DOM	1-DOM	EVEN		
AnSE	0.680	0.680	0.680		
SEP	0.055	0.054	0.054		
PLACE	BACK	FRONT			
AnSE	0.745	0.616			
SEP	0.058	0.051			
COMMON	%50	%80			
AnSE	0.749	0.612			
SEP	0.06	0.048			
C	MIN	MED	MAX		
AnSE	0.865	0.657	0.519		
SEP	0.061	0.056	0.047		
MODELS	10	5			
AnSE	0.812	0.548			
SEP	0.063	0.046			

are performed in the same workstation for all models. Thus, the number setups and learning curve effects are negligible. Our approach with three different model selections, performs the best with the selection of the most frequent model as base and relatively inferior if the similarity index (based on number of tasks) is used for base model selection. This result is also as expected, since our approach achieves less number of setups with respect to SMLB case by fixing some tasks. As these measures are computed in proportion to demand ratios, constructing a balance according to the model with highest demand gives the best results. This performance is considerably close to the performance of the ideal case imposed by MiMLB.

- The MIX is not a significant factor. Therefore, the three different levels are not significantly different.
- The PLACE factor is significant. Better results are obtained for the placing of the common tasks to the front case.
- COMMON factor is considerably significant. The eighty- percent level yields better results as expected.
- C factor is also significant. The average number of setups per task measure is the best when the maximum cycle time is used. The minimum cycle time level yields the worst average number of setups per task. For an instance the number of tasks are constant. As the cycle time is increased, same work content are assigned to small number of workstations. Hence, the average number of setups per task decreases.
- In the percentage of stations where set ups are performed, the performance of the cycle times levels is in reversed order. Since the number of workstations increase as the cycle time is decreased, the number of setups divided by number of stations being increased. Hence, better results are observed with the minimum cycle time.
- The MODELS factor is also significant. As the number of models are increased the number of setups increase as expected. It is interesting that if we double the number of individual models the increase is performance measures approximately 50%.

Another ANOVA analysis for the results is conducted here with respect to number of stations, maximum number of stations and balance delay. These three response variables are influenced by the PROBLEM, C and METHOD factors as explained in the design section. The results of this analysis of variance are summarized in Table 4.6 and are as follows:

Table 4.6 Summary of ANOVA 2

RESPONSE VARIABLE (R ²)	ANST (0.998)	XNST (0.998)	BD (0.746)
SOURCE	P VALUE	P VALUE	P VALUE
C	0.0001	0.0001	0.0001
PR	0.0001	0.0001	0.0001
METHOD	0.0001	0.0001	0.0001
C*PR	0.0001	0.0001	0.0001
C*METHOD	0.0001	0.0001	
PR*METHOD	0.0001	0.0001	0.0001
C*PR*METH	0.0001	0.0001	0.0001

- The R² values for ANST (average number of stations) and XNST (maximum number of stations) are 0.99. Hence, our setting explains the variance almost perfectly.
- The R² value for the BD (balance delay) is 0.75. Thus, these three factors and their all way interactions are capable of explaining 75% of the changes in balance delay. The utility model can be accepted. A further analysis needs to be conducted to explain the remaining variance.
- In this analysis, all three factors, and their all way interactions are found significant. In balance delay measure, only the C and METHOD interaction is not significant.
- For average and maximum number of stations the highest significance factors is PROBLEM, and C factor is the second. Figures E.1-E.6 in Appendix E demonstrate the effect of these factors.
- For balance delay, all main factors and their all way interactions are equally significant in explaining 75% of the variance.

The follow-up analysis is summarized in Table 4.7

Table 4.7 Summary of the Tukey Follow Up Analysis 2

C	MIN	MED	MAX		
ANST	22.24	14.19	10.03		
XNST	22.79	14.63	10.41		
BD	0.083	0.106	0.097		
METHODS	Mu-WC	Mu-SI	Mu-FQ	SINGLE	MIXED
ANST	16.72	16.48	15.76	14.42	14.04
XNST	17.21	16.98	16.26	15.25	14.04
BD	0.105	0.103	0.098	0.090	0.088

The follow-up analysis yields:

- The C factor is significant and as cycle time increases, the average number of stations and the maximum number of stations will decrease as expected. In case of the balance delay, the best results are coming from minimum cycle time level and the worst results are from the medium cycle time level.

- All methods are significantly different, in terms of all the response variables best results are obtained by MiMLB, which is followed by SMLB. See Appendix E for corresponding plots of the response variables. This is actually contrary to our expectations. The reason why, MiMLB yields good results cannot be explained. However, the underlying assumption of MiMLB is to have batch sizes being one. In our experimentation the model mixes includes batch sizes greater than one for some of the models. In terms of balance delay, in 36 out of the 57 problems, MiMLB performs better than the other two approaches. However, in the remaining 21 instances, our approach is in between these two approaches as we expected. This might indicate that the results are highly problem dependent.

- The levels of PROBLEM factor are analyzed below:
 - For average number of stations, the problems are significantly different. The increase in the number of tasks is not the whole reason. The task times given to the problems are also related.
 - For maximum number of stations, the problems 3, 9 and 4, 5 yield the same results, which is only a coincidence.
 - The balance delay of problems 1, 7, 9 and 12 are very high. Like problem 12 which not be solved optimally, these constitute the most though problems set leading to high balance delay values. The rest of the problems have similar balance delays.

Both of the analysis conducted showed that conducting the analysis on many problems that are very different from each other in terms of number of tasks, task times and precedence relations should be avoided. The multi-model problems generated, turned out to be highly case dependent. Instead focus should be placed on similar problems, so that those specifics regarding the METHOD and other factors can be captured. The averaging out procedure conducted in analysis of variance in that case would be more appropriate.

This analysis especially in cases with low R^2 values showed the existence of other factors that need to be incorporated. Other characteristics of the precedence diagram (topological aspects) can be included for further analysis.

CHAPTER 5

CONCLUSIONS AND FURTHER RESEARCH ISSUES

In this study, we deal with multi-model assembly line balancing (MuMLB) problem. There are only a few studies in the literature that deal with MuMLB. Some researches suggest that the Single-Model Line Balancing (SMLB) procedures to be if the batch sizes of the models are large and that it is not important to process the same task at the same workstation for all models. Mixed-Model Line Balancing (MiMLB) procedures are suggested if the batch sizes are very small and it is desired to process the same task at the same workstation for all models. Both approaches when used with a multi-model line have their own advantages and disadvantages with respect to different performance measures regarding the efficiency of the assembly line. In this study, we attempt to propose an approach for MuMLB problem and perform experimental analysis to compare the three methods used to balance the multi-model line.

In this study, we propose an approach for balancing the multi-model lines. In this approach, we define a *fixed tasks set*, that consists of the tasks are to be assigned to the same workstations permanently for all models. A mechanism facilitating the assignment of these tasks to the stations is constructed, that first selects a model among models to be assembled according to a given criterion and then includes in this model those tasks that lack in the fixed tasks set. The selected model is thus augmented so as to obtain a *base model*. Our approach, using this base model, assigns tasks of the base model to workstations for a given cycle time. A branch-and-bound procedure available in the literature-EUREKA- is modified

and used for this purpose. Thus, the fixed tasks are assigned to their permanent stations.

Given the assignment of the fixed tasks to their corresponding workstations and a cycle time, a *Partial Type I Problem* is defined and solved for constructing the balances of the individual models. The solution procedure includes definition of the upper and lower bounds for Partial Type I problem. The previously used branch-and-bound procedure is modified to solve the Partial Type I problem.

In the proposed approach, we intend to have a saving in the number of stations making up the line and lower the balance delay in comparison to MiMLB approach; to decrease the number of setups required when shifting tasks between stations and learning curve effects of workers in comparison to the SMLB approach. Our approach is expected to be in between these two approaches for each of these performance measures.

The proposed approach can also be used by practitioners as a learning tool. This procedure may guide them in understanding their multi-model lines and may increase their capabilities in balancing their line.

In this study, we conduct an experiment to compare the three approaches and see the performance of our approach in terms of the above stated performance measures. Factors considered in the experiment include cycle time, number of models assembled on the line, the model mix, the position of common tasks in the precedence diagram and the percentage of common tasks. Different instances have been generated using the test problems available in the ALB literature. The effects of these factors on the response variables representing the performance measures of our interest are statistically analyzed. The main results can be summarized as follows:

In terms of performance measures related to number of setups and learning curve effect, our approach seems to outperform SMLB which is an expected result. The result is as we expected. Hence our approach by assigning the fixed tasks set

to their stations permanently achieves a reduction in the number of setups and therefore in the learning curve effect. Among our three different criteria for selecting the model to form the base model, the selection criterion based on the highest demand outperforms the other two alternatives criteria. Its result is very close to the ideal case, which is imposed by the MiMLB approach.

In terms of performance measures related to the number of stations and balance delay, our approach performs worse than the other approaches. Our approach's performance was expected to be worse than SMLB in terms of the number of stations. However, we cannot explain how MiMLB outperforms SMLB in terms of this performance measure. In terms of balance delay, in 36 out of the 57 problems, MiMLB performs better than the other two approaches. However, in the remaining 21 instances, our approach is in between these two approaches. This might indicate that the results are highly problem dependent.

The results of the statistical analysis showed that our approach needs further improvement. A further improvement is related to the fixed tasks set. Finding the fixed tasks set, determining the size of the fixed task set is a big problem and needs further analysis. Tasks that are known to have long setup times with respect to the setup time of the other tasks can also be incorporated in the fixed tasks set. Another further improvement is related to the base model. The base model idea needs to be sophisticated. For this purpose, similarity indexes can be further explored. The similarity index used in this study turned out to be naive. Similarity indices that can represent several aspects of the precedence diagram, including the topology of it should be considered.

The performance measures used in this study should also be further improved. Certain surrogate measures have been used to represent the number of setups and learning curve effect. Our approach due to the fixing of some of the tasks reduces raw materials and components handling. Hence less congestion is achieved. However congestion is not considered as a performance measure in our study. There may be also other performance measures that we did not include in this study but would be in favour of the proposed approach.

We could solve problems with 148-200 tasks. We failed to solve the 297-task problem instances generated from test problem in the literature. Hence, the problem size –number of tasks- should be increased in our algorithm.

A further analysis includes a comparison of our results with the optimal solutions. This requires determining the literature problems, which are not solved optimally with the techniques known up to date. If the number of such instances is high, ways for improving our upper bound can be sought.

The lower bounds and upper bounds defined for Partial Type I problem can be used in SMLB. The analysis of these bounds' tightness is also an issue that needs further analysis.

The results of the analysis also show that better factor levels need to be defined for the experimentation. For instance, lower values for the percentage of common tasks need to be explored. The random scattering of the common tasks on the precedence diagram of the models can also be analyzed further. Also other factors regarding topological aspects of the precedence diagram can be incorporated in further analysis.

The analysis of variance due to its nature takes averages. Hence the averaging out process of the problems that are very different from each other turned out to be not appropriate. The instances generated for experimentation are taken from 19 problems available in the literature. The results of the analysis show that the results are significantly problem-dependent. Therefore, a further experimental analysis can be conducted on random instances generated from a single problem set or by focusing on similar problems.

In the literature usually the fixing of certain tasks to certain stations is handled by decomposition of the problem into smaller subproblems. However, our approach without decomposing handles it with an overall approach. This study defines Partial Type I problem and solves it. A further research direction can be defining Partial Type II problem and proposing similar solution procedures.

Batch sizing and batch sequencing issues can be incorporated in our MuMLB approach. Also performances of the different approaches should be compared over different model sequences. The performance measure associated with the number of setups and the learning curve effect should be enhanced to incorporate the sequencing effect.

In this study we have not considered batch size as a factor. If batch sizes are defined as a factor with levels very small batches, large batches and moderate batches, it would be more appropriate in distinguishing which balancing method to use under given batch size level. Comparing the three methods using three different batch size levels need also further analysis. A simulation type of analysis can also be very useful in demonstrating differences of the three methods.

A final further analysis can be conducted using the Multi Criteria Decision-Making approach. The balance delay, number of stations and number of setup performance measures can be combined on a single graph to examine the tradeoffs. Scatter type of graphs can be used to closely examine a problem and a pair of the related performance measures using this approach.

REFERENCES

- Agaki, F., Osaki, H. and Kikuchi, S., 1983. 'A method for assembly line balancing with more than one worker in each station', *International Journal of Production Research*, **21** (5), 755-770.
- Askin, R. G. and Standridge, C. H., 1993. *Modelling and Analysis of Manufacturing Systems*, John Wiley & Sons, New York.
- Baybars, I., 1986, 'A survey of exact algorithms for the simple assembly line balancing problem', *Management Science*, **32** (8), 909-932.
- Berger, I., Bourjolly, J. -M. and Laporte, G., 1992. 'Branch-and-bound algorithms for the multi-product assembly line balancing problem', *European Journal of Operations Research Practical, Combinatorial Optimization*, **58**, 215-222.
- Bhattaacharjee, T. K. and Sahu, S., 1988. 'A critique of some current assembly line balancing techniques', *International Journal of Production Management*, **7** (6), 32-43.
- Buxey, G. M., Slack, N. D. and Wild, R., 1973. 'Production flow line system design - A review', *AIIE Transactions*, March, 37-48.
- Chakravarty, A. K. and Shtub, A., 1985. 'Balancing mixed model lines with in-process inventories', *Management Science*, **31** (9), 1161-1174.

Ding, F.-Y., and Cheng, L., 1993. 'An effective mixed-model assembly line sequencing heuristic for JIT production systems', *Journal of Operations Management*, **11**, 45-50.

Erel, E. and Sarin, S. C., 1998. 'A survey of the assembly line balancing procedures', *Production Planning & Control*, **9** (5), 414-434.

Fokkert, J. I. Z. and de Kok, T. G., 1997. 'The mixed and multi-model line balancing problem: A comparison', *European Journal of Operations Research*, **100**, 399-412.

Ghosh, S., and Gagnon, R. J., 1989. 'A comprehensive literature review and analysis of the design, balancing and scheduling of the assembly systems', *International Journal of Management Science*, **27** (4), 637-670.

Hackman, S. T., Magazine, M. J. and Wee, T. S., 1989. 'Fast, effective algorithms for simple assembly line balancing problems', *Operations Research*, **37** (6), 916-924.

Hax, A. C. and Candea, D., 1984. *Production and Inventory Management*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Helgeson, W. P. and Birnie, D. P., 1961. 'Assembly line balancing using the ranked positional weight technique', *Journal of Industrial Engineering*, **12** (6), 394-398.

Hoffmann, T. R., 1963. 'Assembly line balancing with a precedence matrix', *Management Science*, **9**, 551-562.

Hoffmann, T. R., 1992. 'EUREKA: A hybrid system for assembly line balancing', *Management Science*, **38**, 39-47.

Ignall, E. J., 1965. 'A review of assembly line balancing', *Journal of Industrial Engineering*, **16**, 244-254.

Jackson, J.R., 1956, 'A computing procedure for a line balancing problem', *Management Science*, **2**, 261-271.

Johnson, R. V., 1981. 'Assembly line balancing algorithms: computation comparisons', *International Journal of Production Research*, **19** (3), 277-287.

Johnson, R. V., 1983. 'A branch-and-bound algorithm for assembly line balancing problems with formulation irregularities', *Management Science*, **29** (11), 1309-1323.

Johnson, R. V., 1988. 'Optimally balancing large assembly lines with FABLE', *Management Science*, **34** (2), 240-253.

Klein, R. and Scholl, A., 1996. 'Maximizing the production rate in simple assembly line balancing-A branch-and-bound procedure', *European Journal of Operational Research*, **62**, 367-385.

Labbe, M., Laporte, G., and Mercure, H., 1991. 'Capacitated vehicle routing on trees', *Operations Research*, **39**, 616-622.

Lehman, M., 1969. 'On criteria for assigning models to assembly lines', *International Journal of Production Research*, **7** (4), 269-285.

Macaskill, J. L. C., 1972. 'Production-line balances for mixed model lines', *Management Science*, **19** (4), 423-434.

Milas, G. H., 1990. 'Assembly line balancing...Let's remove the mystery', *Industrial Engineering*, May, 31-36.

Miltenburg, G. J. and Wijngaard, J., 1994. 'The U-line balancing problem', *Management Science*, **40** (10), 1378-1388.

Nourie, F. J. and Venta, E. R., 1996. 'Note: Microcomputer performance of OptPack on Hoffmann's data sets: comparison with Eureka and FABLE', *Management Science*, **42** (2), 304-306.

Prenting, T. O. and Thomopoulos, N. T., 1974. *Humanism and Technology in Assembly Line Systems*, Hayden Book Co.

Salveson, M. E., 1955. 'The assembly line balancing problem', *Journal Industrial Engineering*, **6**, 18-25.

Scholl, A., 1994. Ein B&B-Verfahren zur Abstimmung von Fließbändern bei gegebener Stationsanzahl, in: H. Dyekhoff, U. Derigs, M. Salomon and H.J. Tijms (eds.), *Operations Research Proceedings 1993*, Springer-Verlag, Berlin, 175-181.

Scholl, A. and Klein, R., 1997. 'SALOME: A bidirectional branch-and-bound procedure for assembly line balancing', *INFORMS Journal on Computing*, **9**, 319-334.

Scholl, A. and Klein, R., 1999. 'Balancing assembly lines effectively-A computational comparison', *European Journal of Operational Research*, **114**, 50-58.

Sprecher, A., 1999. 'A competitive branch-and-bound algorithm for the simple assembly line balancing problem', *International Journal of Production Research*, **37** (8), 1787-1816.

Sumichrast, R. T., and Russel, R. S., 1990. 'Evaluating mixed-model assembly line sequencing heuristics for JIT production systems', *Journal of Operations Management*, **9** (3), 371-390.

Thomopoulos, N. T., 1967. 'Line balancing-sequencing for mixed-model assembly', *Management Science*, **14** (2), 59-75.

Thomopoulos, N. T., 1970. 'Mixed model line balancing with smoothed station assignments', *Management Science*, **16** (9), 593-603.

Uğurdağ, H. F., Rachamadugu, R., and Papachristou, C. A., 1997. 'Designing paced assembly lines with fixed number of stations', *European Journal of Operational Research*, **102** (3), 488-501.

Wild, R., 1972. *Mass-production management. The Design and Operation of Production Flow-line Systems*, John Wiley and Sons, New York.

Wild, R., 1974. 'Mass production in engineering', *International Journal of Production Research*, **12** (5), 533-545.

Young, H. H., 1967. 'Optimization models for production lines', *Journal of Industrial Engineering*, **18** (1), 70-78.

APPENDIX A

CONSTRUCTION OF THE BASE MODEL

After a model is selected and the set of fixed tasks is determined, the method described for constructing combined precedence diagrams is used to establish the base model. $G_b = (V_b, A_b)$ represents the graph of the base model where V_b is the set of tasks of the base model and A_b is the set of precedence relations. $G_s = (V_s, A_s)$ represents the graph of the selected model. The tasks in the base model, V_b , are simply the tasks of the selected model and the tasks in the fixed tasks set (FT), i.e.

$$V_b = V_s \cup FT.$$

The precedence relations of the base model, A_b , is established in an iterative procedure summarized as follows:

1. Starting with $A_b = A_s$ and let $LT = FT$.
2. Take $i \in LT$
 - a. For all j that are immediate predecessors of task i perform the following
If $j \in V_b$ $A_b = A_b \cup (j, i)$ go to (b)
Otherwise if task j has no predecessors go to (b)
Else find a predecessor k of task j where $k \in N_b$ and
 $A_b = A_b \cup (k, i)$ go to (b)
 - b. For all j that are immediate successors of task i perform the following
If $j \in V_b$ $A_b = A_b \cup (i, j)$ go to step 3
Otherwise if task j has no predecessors go to step 3
Else find a predecessor k of task j where $k \in N_b$ and
 $A_b = A_b \cup (i, k)$ go to step 3
3. $LT = LT \setminus \{i\}$
If LT is empty STOP.
Otherwise go to step 2.

APPENDIX B

DETERMINATION OF LB_2

Here a brief summary of the determination of LB_2 is also provided. Define a and b such that $0 \leq a < b \leq C$ and $W(a, b) = \{i \in N: a < t_i \leq b\}$. Order the tasks in nondecreasing order of task times.

1. Starting with the longest task assign the tasks of $W(1/2 * C, C)$ to different stations, since any of these two tasks cannot be assigned to the same workstation without exceeding the cycle time C .
2. Starting with the shortest task in $W(1/3 * C, 1/2 * C)$ and all stations where tasks of $W(1/2 * C, C)$ are assigned, successively assign the tasks of $W(1/3 * C, 1/2 * C)$ to the first idle station. Note that the stations are in order of nondecreasing idle times. When all tasks have been assigned or when no stations are left stop.
3. Let K be the number of unassigned tasks in $W(1/3 * C, 1/2 * C)$ left over from step 2. Then at least $\lceil 1/2 * K \rceil$ additional stations are required since at most two tasks of this set can be assigned to the same workstation.
4. Determine the minimum number of additional stations required for all unassigned tasks. For every $b \in [0, 1/3 * C)$ the number of additional stations required for all tasks for which $t_i > b$ needs to be determined.
$$p(b) = \{ 0, \lceil \frac{\sum_{i \in W(b, C-b)} t_i - (|W(1/2 * C, C-b)| + \lceil 1/2 * K \rceil) * C}{C} \rceil \}$$

The computation of $p(b)$ rests on the fact that tasks i with $t_i > b$ cannot be assigned to workstations which already have an work content at least $C-b$ time units. For a given b a valid lower bound on the number of required workstations is

$$LB_2(b) = |W(1/2 * C, C)| + \lceil 1/2 * K \rceil + p(b)$$

Since this holds for every b , the required bound is given by

$$LB_2 = \max_{0 \leq b < 1/3 * C} \{LB_2(b)\}$$

The values of b are restricted with the interval $[0, 1/3 * C)$ since $p(b) = 0$ for $b \geq 1/3 * C$. When b values corresponding only to task times are examined, the lower bound can be computed in $O(n)$ if the task times are given in nondecreasing order of task times.



APPENDIX C

MODIFIED EUREKA ALGORITHM

The branch-and-bound procedure performed in our modified EUREKA algorithm can be summarized as follows:

1. Set $S=LB$. Compute theoretical minimal total slack= $LB * C - \sum_i t_i$
2. Set current trial station $s=1$.
3. Generate a feasible task assignment, i , that has not been considered previously for trial station $s=S$.
4. If there are no more feasible assignments for trial station s and $s=S$
 - a. Increase number of stations by one: $S \leftarrow S+1$
 - b. Increase theoretical minimal total slack by c .
 - c. Begin anew to determine if S stations are feasible. Go to Step 2.
5. If there are no more feasible assignments for trial station s and $s < S$, backtrack by decrementing the trial station: $S \leftarrow S-1$ and go to Step 3.
6. If step 3 generated a feasible assignment for station s , perform the following check the task generated in step 3 is an element of the fixed tasks set. Otherwise go to (c)
 - a. If there exists no other element of the fixed tasks set in s go to (c). Otherwise go to (b).
 - b. For tasks in $j \in FT$ that are assigned to station s and for the task i generated in step 3, if

$$\sum_{\substack{j \in FT \\ i \in s}} t_j + t_i + \sum_{k \in X} t_k \leq C$$

where $X = \{j \in FT \text{ and } j \in S: \text{ all tasks in paths between } j \text{ and } j \text{ on combined precedence diagram}\}$, go to (c). Otherwise fathom the node and go to Step 3.

- c. Compute the cumulative slack time on all completed stations.
7. If this cumulative slack exceeds the theoretical minimum total slack time fathom the node, and go to Step 3.
8. If this is trial station s a complete feasible solution with slack less than or equal to the theoretical minimum total slack is found, STOP.
9. Else, increment the trial station number $S \leftarrow S+1$, and go to Step 3.



APPENDIX E

THE MEAN PLOTS OF THE EXPERIMENTS

Some illustrative plots of the experiment for response variables average number of stations, maximum number of stations and balance delay are given in the following figures



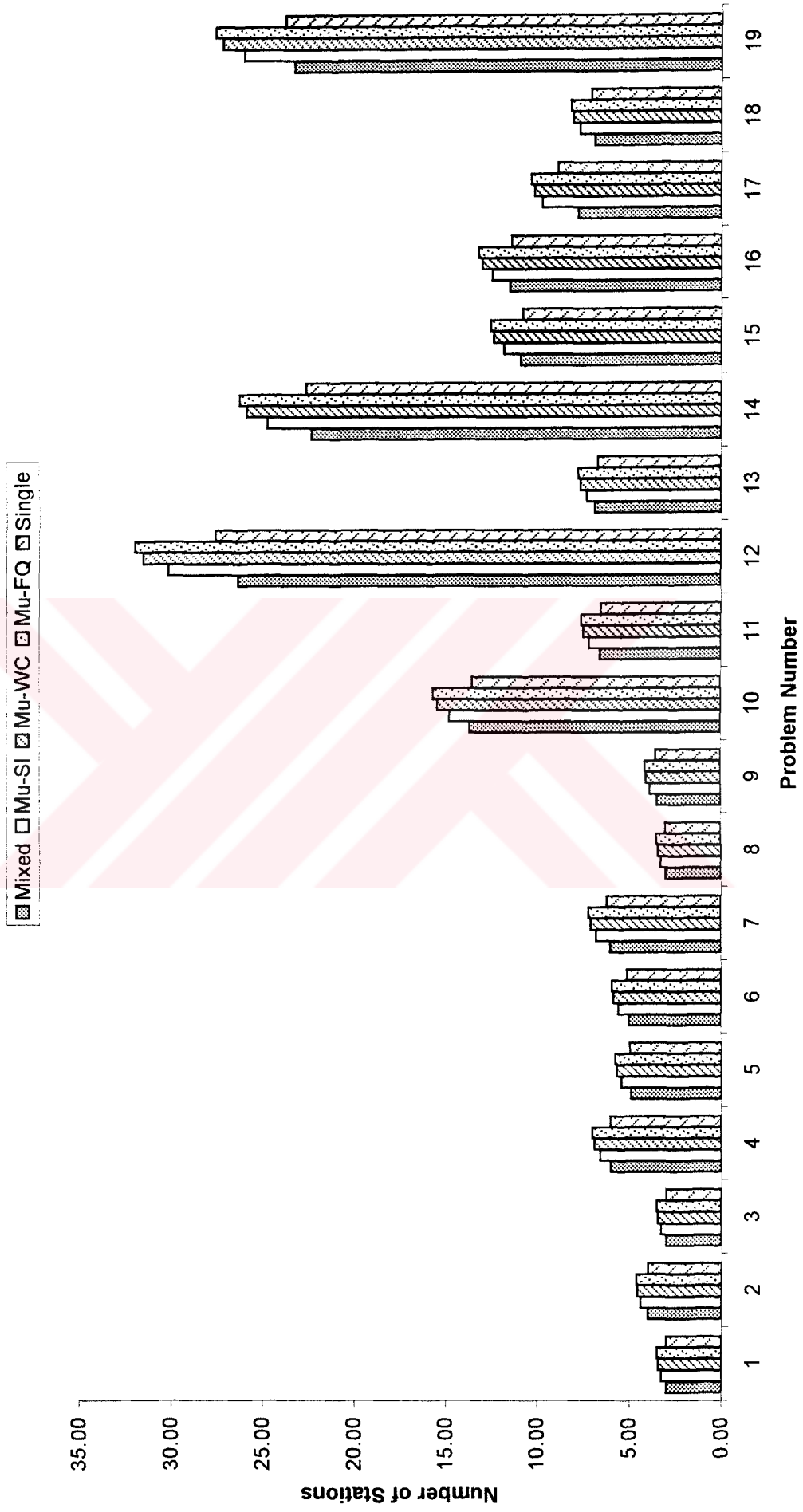


Figure E.1 Average Number of Stations with Maximum Cycle Time

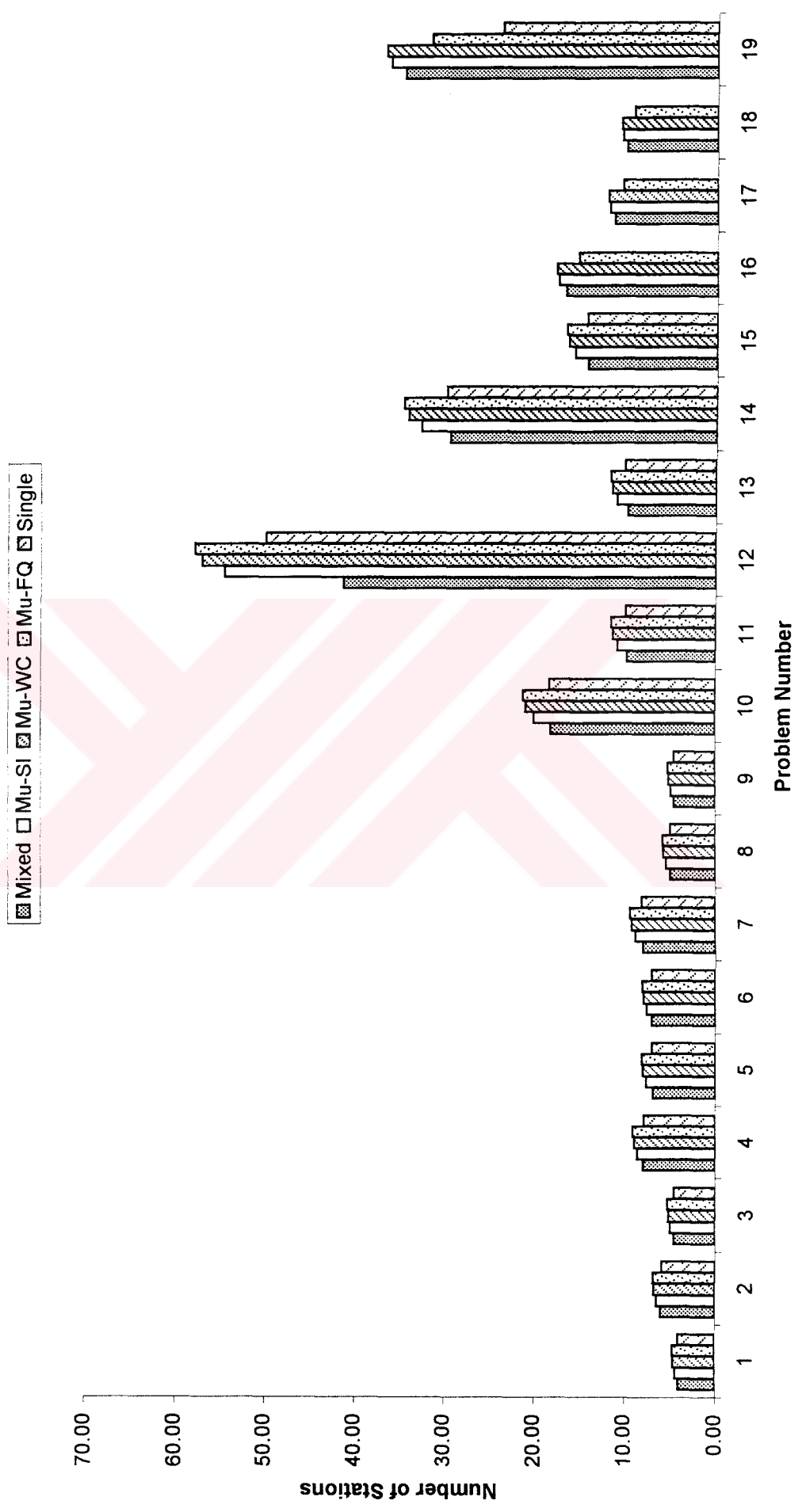


Figure E.2 Average Number of Stations with Medium Cycle Time

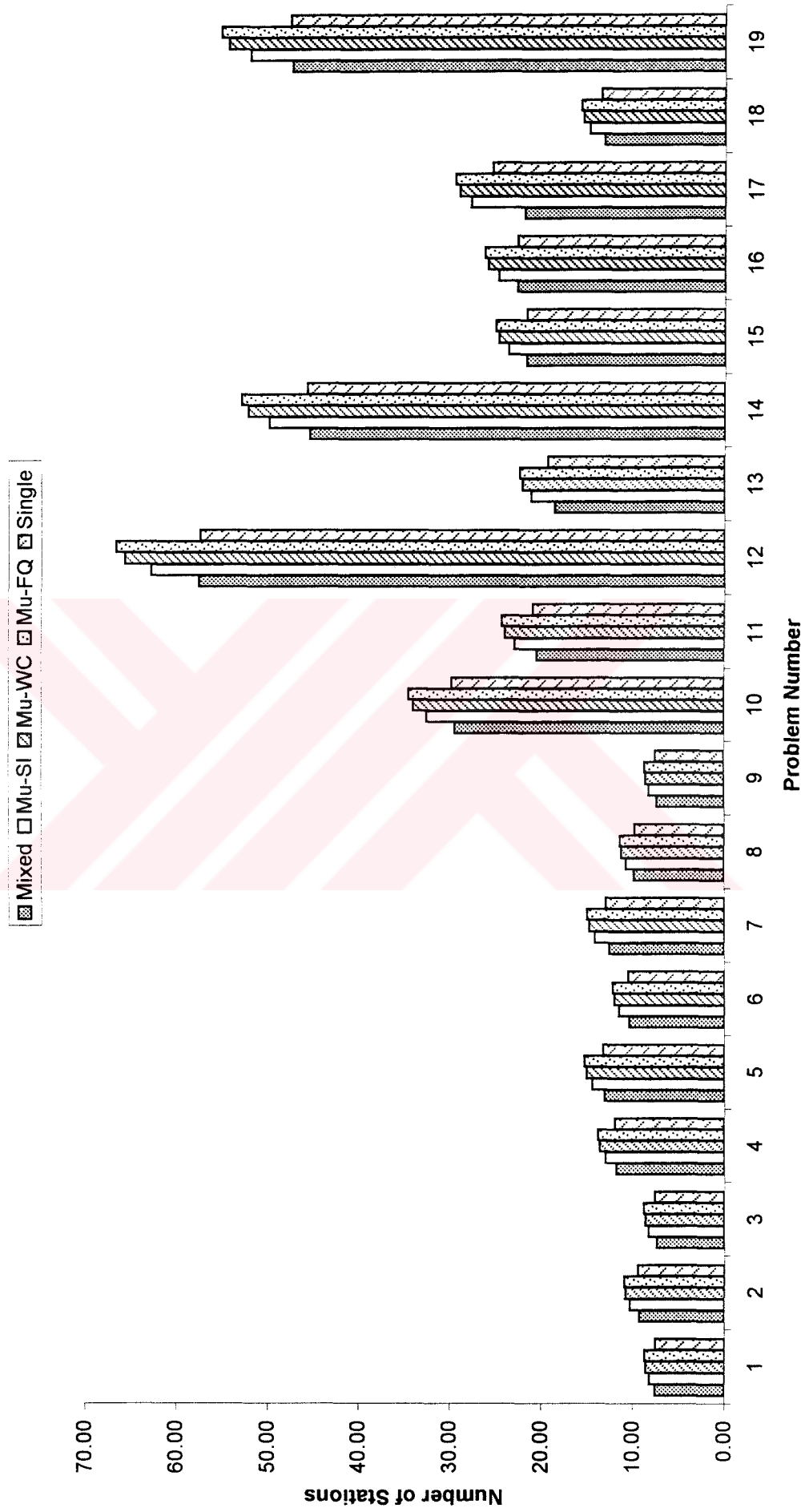


Figure E.3 Average Number of Stations with Minimum Cycle Time



Figure E.4 Maximum Number of Stations with Maximum Cycle Time

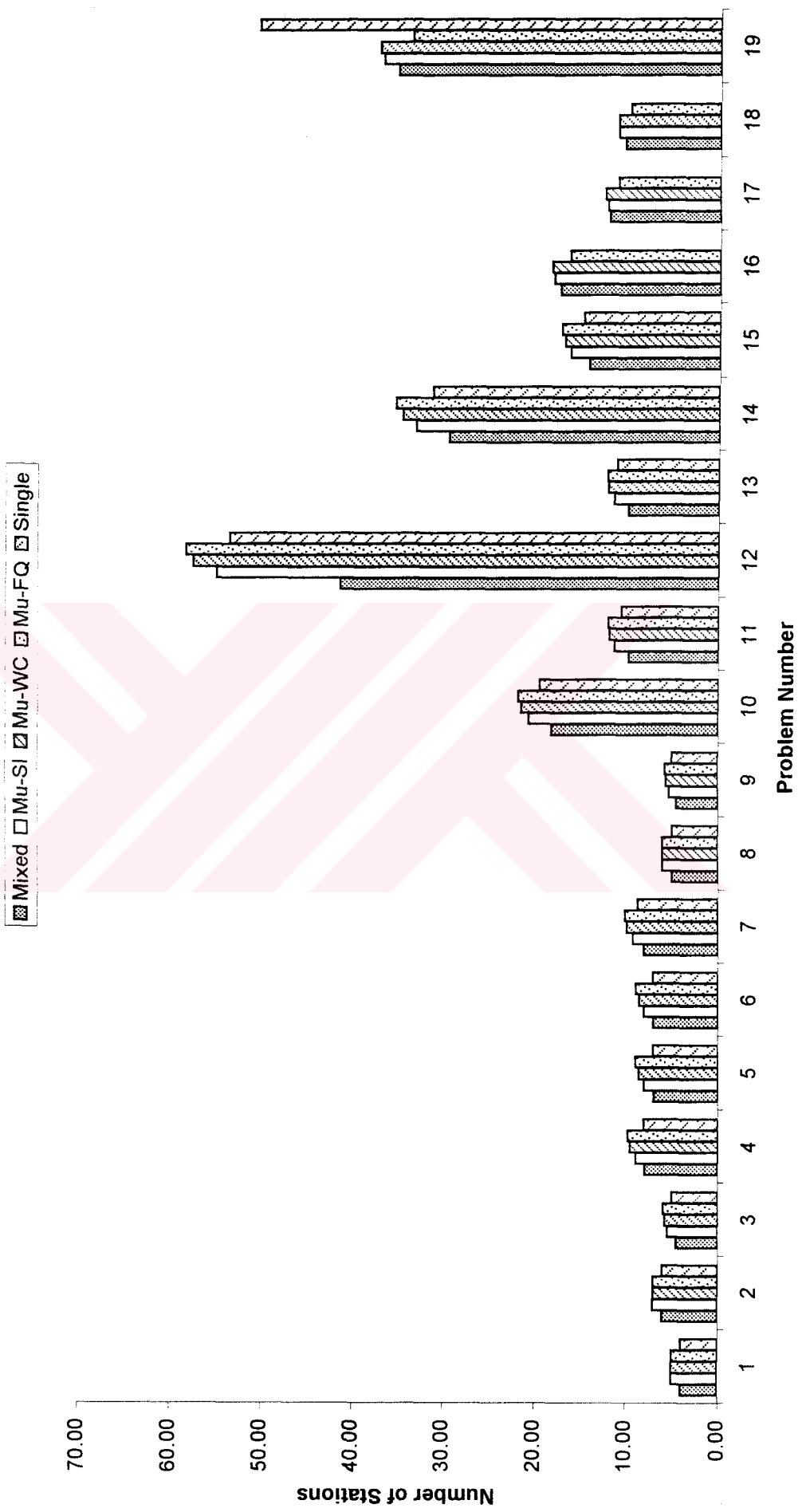


Figure E.5 Maximum Number of Stations with Medium Cycle Time

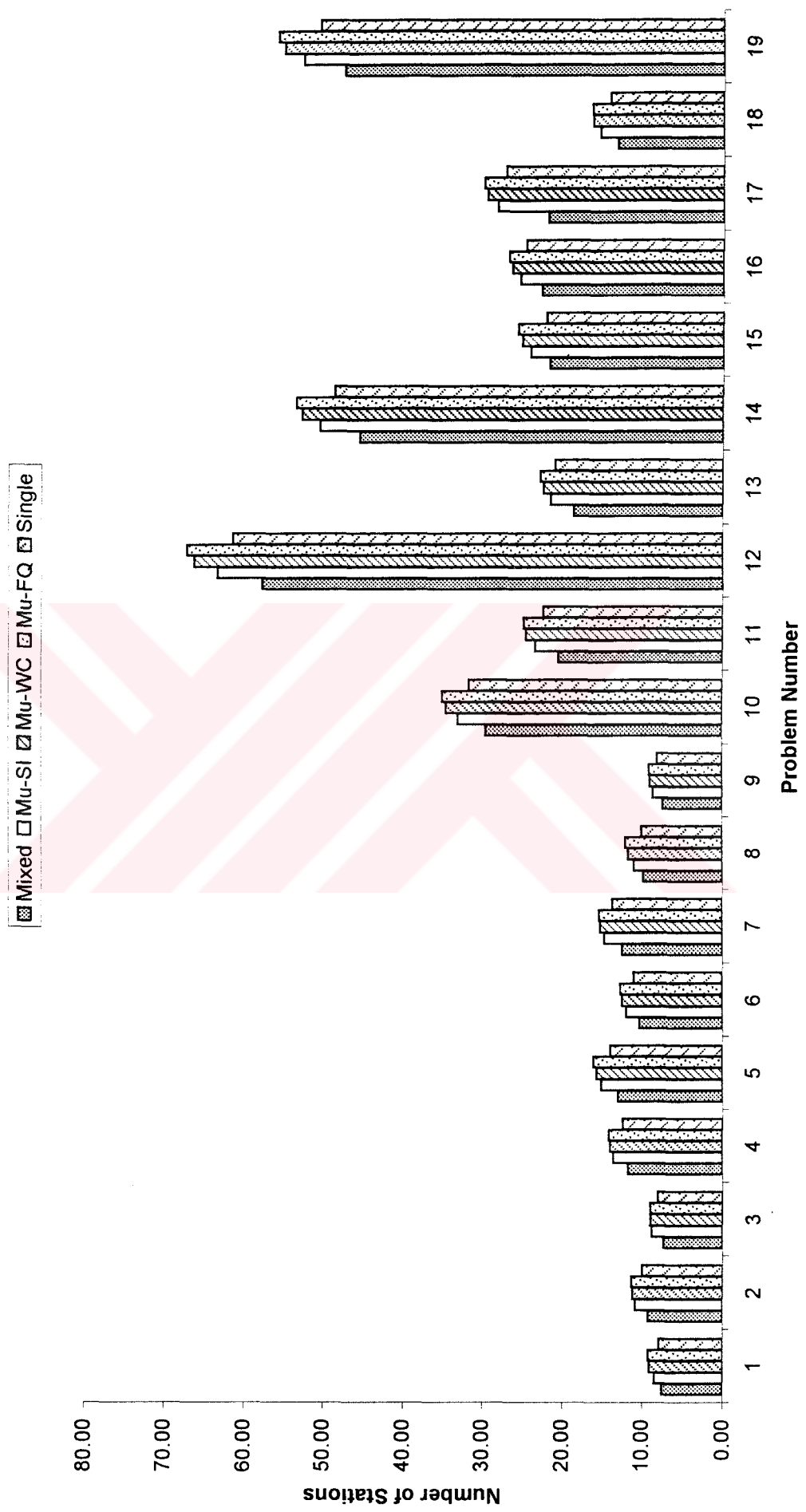


Figure E.6 Maximum Number of Stations with Minimum Cycle Time

▣ Mixed ▣ Mu-SI ▣ Mu-WC ▣ Mu-FQ ▣ Single

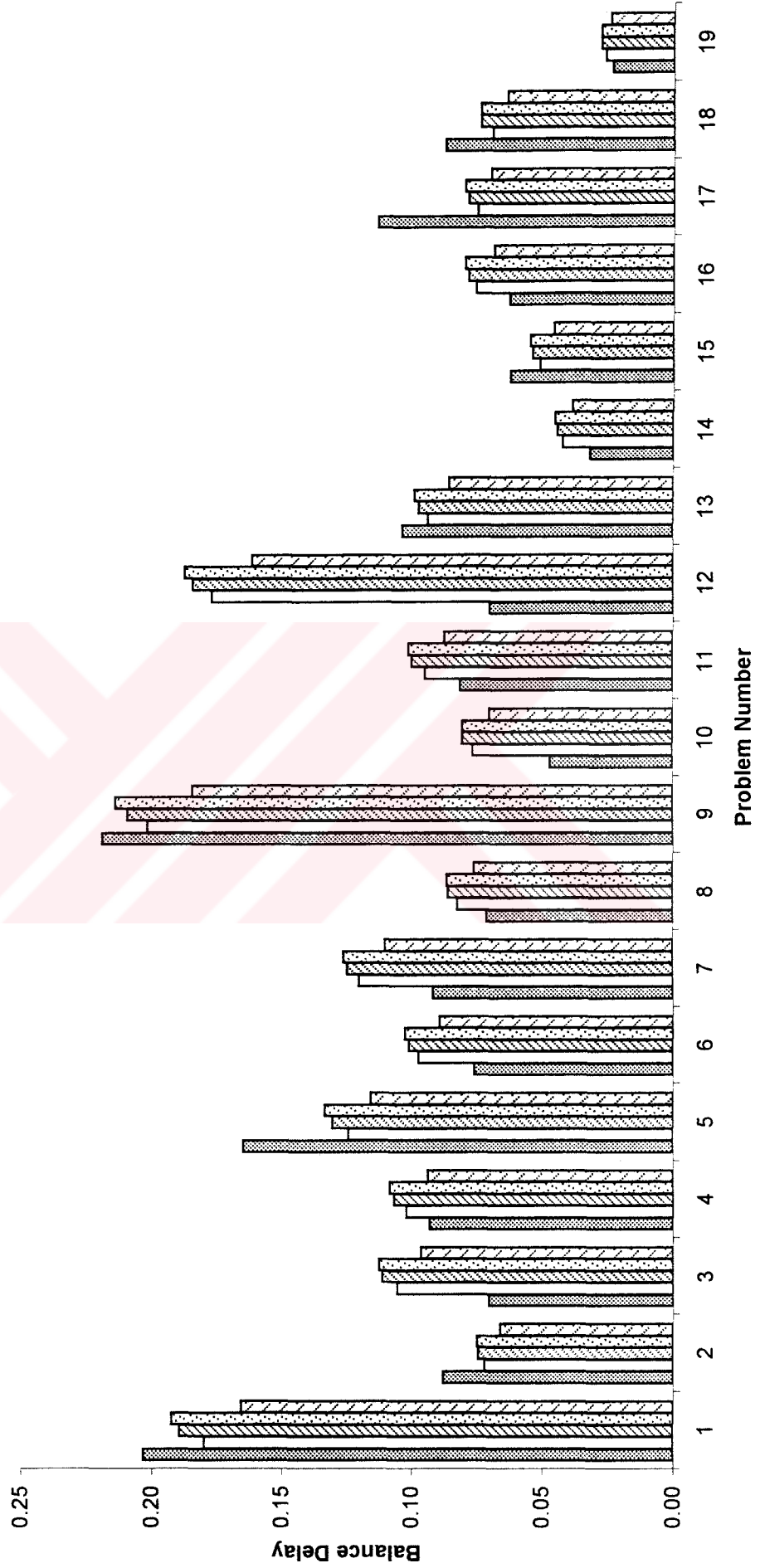


Figure E.7 Balance Delay with Maximum Cycle Time

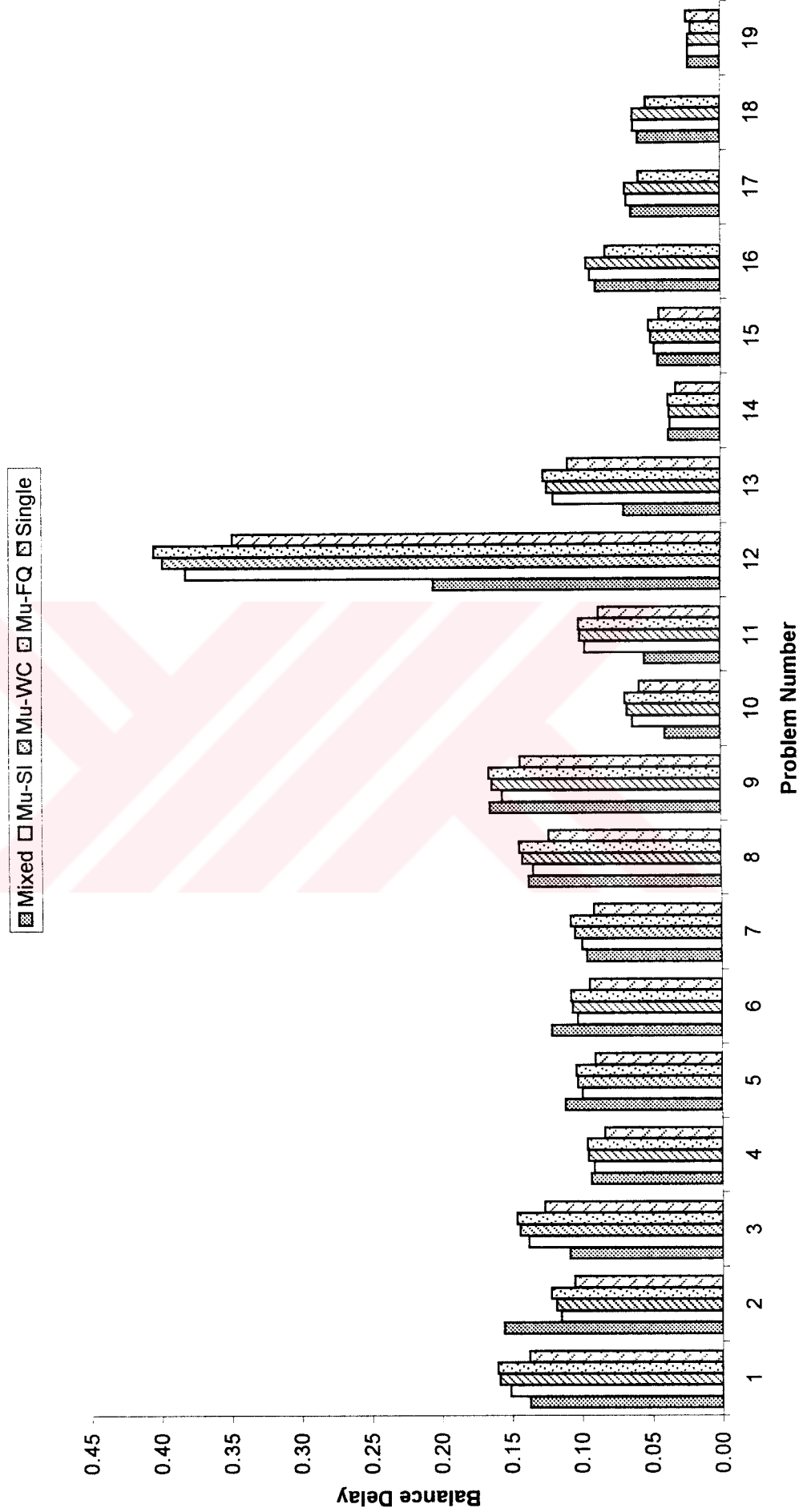


Figure E.8 Balance Delay with Medium Cycle Time

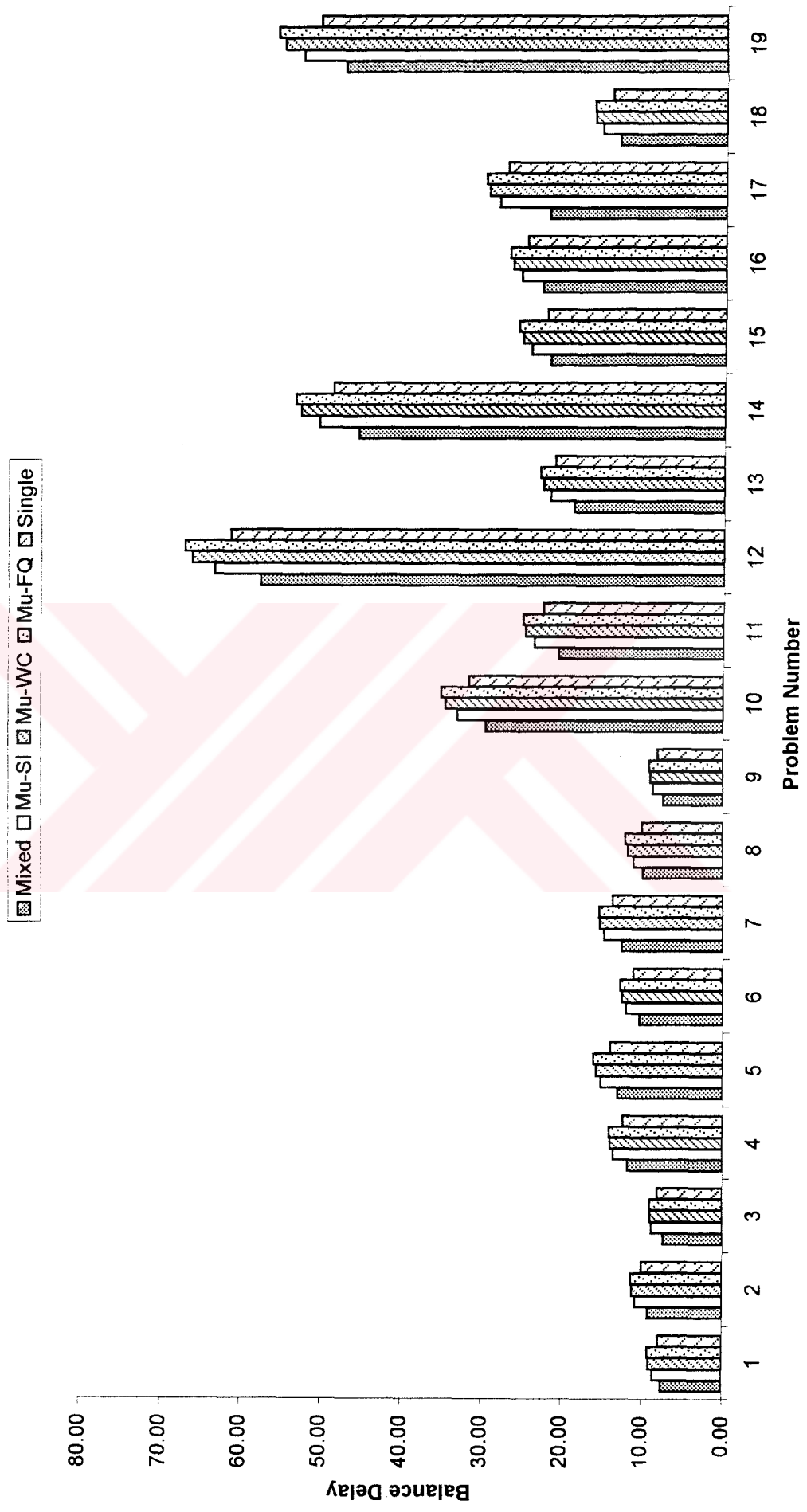


Figure E.9 Balance Delay with Minimum Cycle Time