MIXED-MODEL TWO-SIDED ASSEMBLY LINE BALANCING

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

BY

EMRE UÇAR

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

JANUARY 2010

Approval of the thesis:

MIXED-MODEL TWO-SIDED ASSEMBLY LINE BALANCING

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

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

Prof. Dr. Nur Evin Özdemirel
Head of Department, **Industrial Engineering** _____

Prof. Dr. Ömer Kırca
Supervisor, **Industrial Engineering Dept., METU** _____

Asst. Prof. Dr. Sedef Meral
Co-Supervisor, **Industrial Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Meral Azizoğlu _____
Industrial Engineering Dept., METU

Prof. Dr. Ömer Kırca _____
Industrial Engineering Dept., METU

Asst. Prof. Dr. Sedef Meral _____
Industrial Engineering Dept., METU

Asst. Prof. Dr. Ferda Can Çetinkaya _____
Industrial Engineering Dept., Çankaya University

Hüseyin Güden _____
Industrial Engineering Dept., Başkent University

Date: _____

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

Name, Surname : Emre, UÇAR

Signature :

ABSTRACT

MIXED-MODEL TWO-SIDED ASSEMBLY LINE BALANCING

UÇAR, Emre

M.S., Department of Industrial Engineering

Supervisor: Prof. Dr. Ömer KIRCA

Co-Supervisor: Assist. Prof. Dr. Sedef MERAL

January 2010, 148 pages

In this study we focus on two-sided mixed-model assembly line balancing type-I problem. There is a production target for a fixed time horizon and the objective is to produce this amount with the minimum level of workforce. A mathematical model is developed to solve this problem in an optimal manner. For large scale problems, the mathematical model fails to give the optimal solution within reasonable computational times. Thus, a heuristic approach based on threshold accepting algorithm is presented. Both the mathematical model and the heuristic approach are executed to solve several example problems from the literature and a case study problem which is derived from the refrigerator production. Computational experiments are carried out using both approaches. It is observed that the heuristic procedure finds good solutions within very reasonable computational times.

Keywords: Mixed-Model, Two-Sided Assembly Line, Assembly Line Balancing, Threshold Accepting

ÖZ

KARIŞIK-MODELLİ ÇİFT-TARAFLI MONTAJ HATTI DENGELEMESİ

UÇAR, Emre

Yüksek Lisans, Endüstri Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Ömer KIRCA

Ortak Tez Yöneticisi: Yrd. Doç. Dr. Sedef MERAL

Ocak 2010, 148 sayfa

Bu çalışmada çift taraflı montaj bandında karışık modelli üretim tip-I dengelemesi üzerinde yoğunlaşılır. Belirli bir zaman dilimi için hedef bir üretim adedi mevcuttur ve amaç bu üretimi en az iş gücüyle gerçekleştirmektir. Çalışmada bu probleme optimal çözüm sağlayan matematiksel bir model geliştirilir. Ancak, büyük ölçekli problemlerde matematiksel model makul sürelerde optimal çözümü bulamamaktadır. Bu yüzden, büyük ölçekli problemler için "eşik kabulü" sezgiseline dayanan bir sezgisel yöntem geliştirilir. Hem matematiksel model, hem de sezgisel yöntem ile literatürden seçilen örnek problemler ile buzdolabı üretiminden seçilen bir vaka analizi problemi çözülür. Her iki yöntemin değişen problem parametrelerine göre oluşan performansı incelenir. Geliştirilen sezgisel yöntemin makul çözüm sürelerinde çok iyi sonuçlar sağladığı gözlemlenir.

Anahtar Kelimeler: Karışık-Model, Çift-Taraflı Montaj Hattı, Montaj Hattı Dengelemesi, Eşik Kabulü

To my family

ACKNOWLEDGEMENTS

I would like to express my most sincere gratitude to Prof. Ömer Kırca and Assist. Prof. Dr. Sedef Meral for their guidance throughout this study.

I again would like to express my deepest and greatest gratitude to my father, Mevlüt Uçar, to my mother, Demet Uçar, last but not the least, to my sister, Ebru Uçar for their continuous support and encouragement during this study. Without them, this study might not find its way to completion.

I would like to thank my former managers at Arçelik Eskişehir Refrigerator Plant for letting me attend lectures for four semesters and bearing the loss of work time. Without their support again this study would not end.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

The assembly line balancing problem (ALBP) has been studied for decades. The assembly lines are necessary for the mass production of many goods that are consumed in large amounts.

An assembly line consists of a number of stations arranged along a conveyor belt or a similar mechanical material handling equipment. *Stations* are places where *workers* and/or robots perform certain operations on the product so that the raw materials turn into a final product. Figure 1.1 depicts an illustration of an assembly line with three stations.

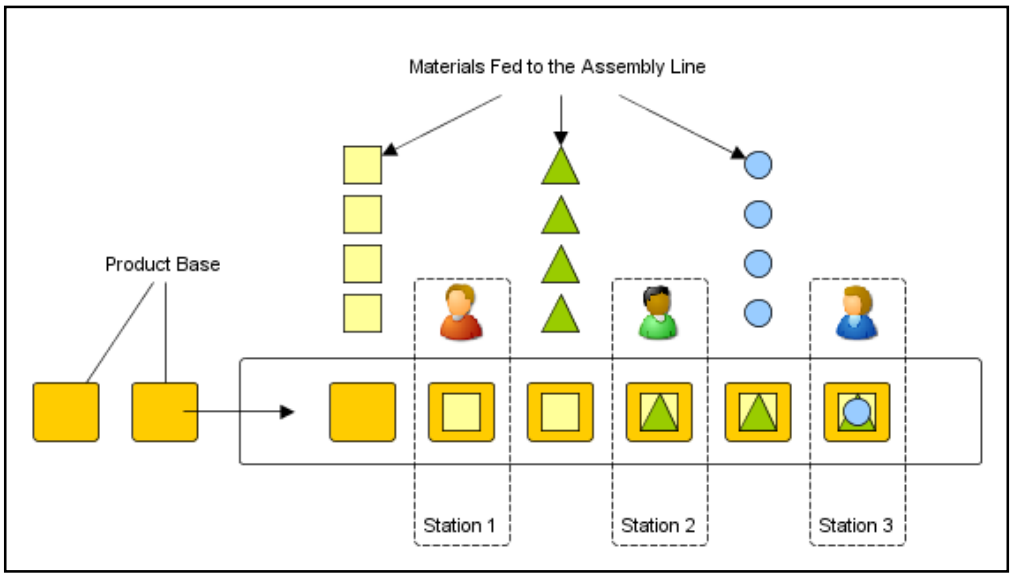

Figure 1.1 3-Station Assembly Line

A *task* is the smallest indivisible individual operation. The time required to perform a task is called as either *processing time* or *task time.* Every product flows down the stations along the assembly line and spends a certain amount of time at every station. This time is equivalent to the total processing time of tasks assigned to the station and called *cycle time* of the line. The production rate of the assembly line is dependent on the cycle time. The *production rate* is the number of final products assembled during a unit period of time. The assembly line balancing problem tries to assign tasks to stations in such a way that an objective function mostly based on cycle time or workforce level is optimized.

When a single product is assembled on the assembly line in large volumes, a single model assembly line balancing problem is considered. A multi-model assembly line is used when different models of a product are assembled on the line. The products are launched in lots and there may be a setup operation between two consecutive lots. In mixed-model assembly lines, different models of the same product are assembled in a mixed order. A mixed-model assembly line can be considered a multi-model assembly line with a lot size of one and without a setup operation required between any two consecutive lots. Figure 1.2 illustrates single model, multi-model and mixed-model assembly lines.

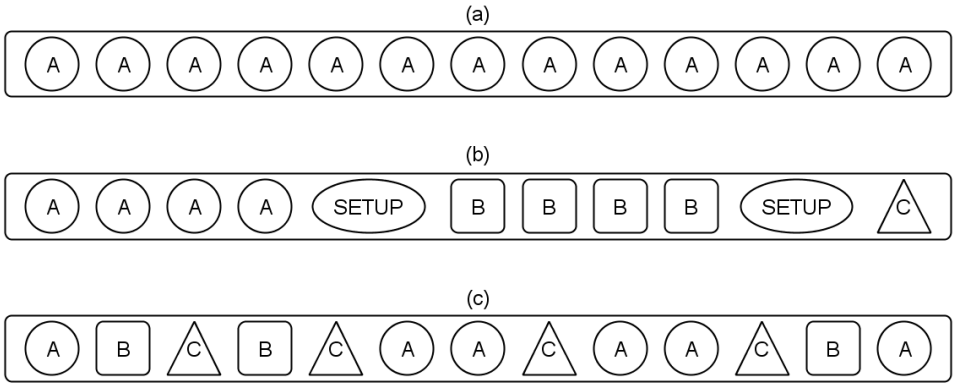

Figure 1.2 Assembly lines: (a) single model, (b) multi-model, (c) mixed-model

Most assembly lines are *serial* in which the stations of the line are arranged in a serial manner. There are also *U-shaped* assembly lines in which the beginning and the ending of the line are close to each other so as to form a narrow U-shape. A worker can work on two products, one of which is close to the beginning and the other is close to the ending of the line. This practice has been shown to improve the efficiency of the line.

The cycle time of a line may be smaller than some task times of a product produced on that line. Then parallel stations should be considered. *Parallelism* is the duplication of a station in which a task group is performed by more than one station simultaneously on different units of the product. There should be a material handling system for distributing the products between parallel stations and collecting them back to the main line.

The size of the product is another concern. If the product is large (like a truck, car, refrigerator), opposite side of the product may not be reachable by the operator at the other side of the line. Then a *two-sided assembly line* should be considered where both sides of the line have workers working on the same work piece simultaneously. Figure 1.3 depicts such a line configuration.

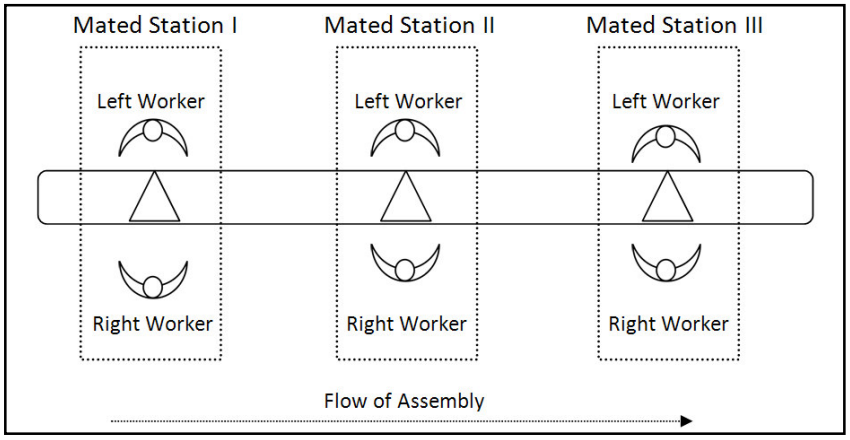

Figure 1.3 A Two-Sided Assembly Line with Three Mated Stations

In a two-sided assembly line, the location at which a right and left side worker work simultaneously on the same product is called a *mated station*. Individual places occupied by each worker will be called a *workstation* or *station* for the rest of the study.

The cycle time of the line may be constant or dependent on the performance of the workers on the line. At constant cycle time lines, the products follow to the next station at a constant speed; hence these lines are called as *paced assembly lines*. On the other hand, the line may be operated with the pace of actual performance of workers on the line. A product follows to the next station when workers at the current station finish their tasks on the product. This type of line is called an *unpaced assembly line*. Figure 1.4 presents taxonomy of the assembly lines based on the features discussed above.

The nature of the product may impose some restrictions on the line balancing. *Positive zoning constraints* require that some of the tasks should be assigned together to the same station, while *negative zoning constraints* require that some of the tasks cannot be assigned together to the same station.

There are *position-related restrictions*. Some tasks should be executed on different ends of the product. Assigning these tasks to the same station may not be efficient. There may be other restrictions related to the nature of the task, operator and station.

Assembly line balancing is assigning the tasks to stations in such a way that all restrictions are satisfied and an objective function is optimized. However, due to the technological constraints of the tasks of the product, some tasks cannot be processed before some other tasks are completed, i.e. some tasks precede some others. Such a relation between any two tasks is called a *precedence relation*. The task that should be processed earlier is called the *predecessor* of the latter task and the task that should be processed later is called the *successor* of the former

task. Precedence relationships among all tasks are to be taken into account for a feasible balancing.

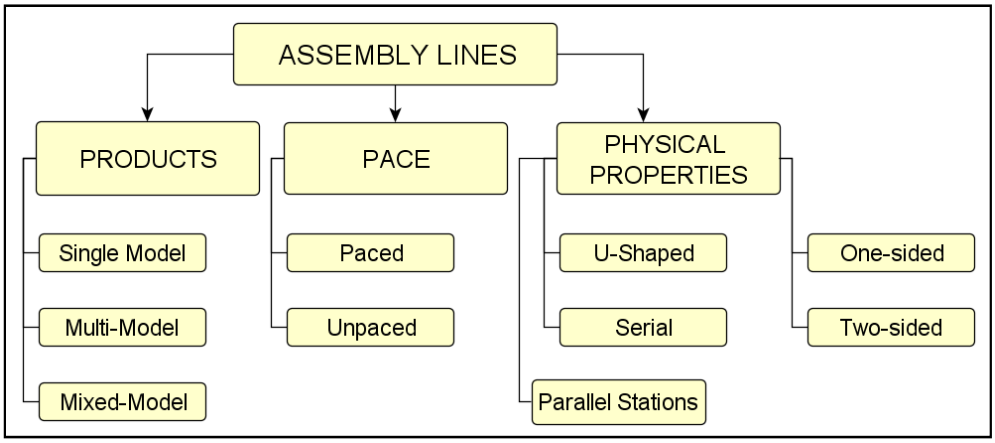

Figure 1.4 Assembly Line Taxonomy

The assembly line balancing may have different objectives. The four main types of objectives can be summarized as follows (Becker and Scholl, 2006).

Type-I problems: Cycle time (production rate) is given and the objective is to minimize the number of workers to achieve a given cycle time.

Type-II problems: The number of stations or number of workers is fixed and the objective is to minimize the cycle time; in other words, to maximize the production rate.

Type-E problems: Simultaneously minimize cycle time and the number of workstations considering their relation with the total idle time or the inefficiency of the line (Bautista and Pereira, 2006)

Type-F problems: The number of stations and the cycle time are given. The objective is to find a feasible balance.

Apart from these main objectives, there are several other objectives. A detailed classification of these objectives and problems is presented by Boysen, Fliedner and Scholl (2007).

In this study, we focus on the mixed-model two-sided assembly line balancing problem with type-I objective that can be considered a new problem domain as the number of studies is limited and only available recently. The problem is a mixed-model problem meaning that a number of versions of a main product - models- are produced on the same assembly line. These models have many tasks in common; however there are also model-specific tasks and the task times may differ among models. The assembly of these models requires the workforce to be distributed both on the right and left-hand sides of the workstations along the assembly line. These types of assembly line balancing problems arise in production environments such as white goods or automobile manufacturing where heavy and big products are produced.

In Chapter 2, literature survey is presented in three sections. First section includes the fundamental studies on simple assembly line balancing; second section includes studies on mixed-model assembly line balancing, and finally the last section includes the studies on two-sided assembly line balancing problem.

In Chapter 3, a mathematical model is presented for the mixed-model two-sided assembly line balancing problem with the objective of minimizing the total workforce. Experimental studies are conducted to test the validity and computational performance of the formulation. A case study problem which is derived from the refrigerator production facility is solved by means of the mathematical model. The production environment of the case study is the main motivation behind this study, where two-sided mixed-model production takes place.

In Chapter 4, a meta-heuristic procedure based on Threshold Accepting Algorithm is presented to solve large scale problems which could not be solved by the mathematical model within reasonable computational times. Test problems and case study problems are solved by means of the proposed heuristic procedure and experimental studies are conducted again.

Chapter 5 is the closing chapter of the study summarizing the work undertaken, the findings and some of the possible future research directions.

CHAPTER 2

LITERATURE REVIEW

In this chapter, literature survey is presented in three sections. The first section includes the survey of studies on the general assembly line balancing problem. The second section is devoted to the mixed-model assembly line balancing problem. In literature there are various studies on this problem. However, most of the studies concentrate on one-sided assembly lines. The third and final section includes the most recent and relevant studies on the two-sided assembly line balancing problem. As mentioned before, the two-sided assembly line balancing problem is a new area of concern and the studies are only a few.

2.1 General Assembly Line Balancing Problem

The very first example of assembly line balancing study can be attributed to Bryton (1954) who first describes and proposes an approach to the problem in his master's thesis work at Northwestern University in 1954. Salveson (1955) is the first person to analytically define the problem in his study. He formulates the problem as a simple assembly line balancing problem of type-1 (SALBP-1) allowing task divisibility. Task divisibility leads to partial assignments of a task to more than one station.

Jackson (1956) proposes an enumeration procedure for SALBP-1. First a procedure is explained on how to draw a precedence diagram using the information on task precedence relationships. The algorithm to solve SALBP relies on the generation of the feasible assignment sets. For the first station a set of combinations of tasks is generated. Every combination in the set consists of one or several tasks and each combination in the set can be a candidate for the

resulting work content of the first station. For the second station, a set of paired combinations of tasks is generated. The first part of every paired combination in the set is a possible assignment for station 1, while the second part is for station 2. This procedure is nested within the main algorithm for the balancing procedure. The balancing ends when the generation ends up including all the tasks that are not included in the previous generation cycle. The resulting combinations in the final set are the resulting work contents for all stations.

Bowman (1960) is the first to provide a "non-divisibility" constraint, by changing the LP formulation to one of integer programming (Baybars, 1986). Bowman formulates the problem in two ways. In the first formulation, the main variable is the amount of time dedicated to perform a task in a given station. The formulation guarantees non-divisibility and tries to pack the operations into the earlier stations on the line, i.e., leaving the latter stations free or in other words not requiring them. The second formulation introduces a new variable replacement, that is, the clock time when an operation starts. Now the objective is to minimize the maximum of the finishing time of tasks with no successors.

Helgeson and Birnie (1961) introduce a method called Ranked Positional Weight Method for the SABLP. This method assigns every task a positional weight rank value. This value for a specified task is equal to the summation of the specified task's own processing time and processing times of all other tasks that cannot be processed before the completion of the specified task. Then considering the cycle time and precedence relations, the task having the largest positional rank value is assigned next to the first available position. The method is easy to apply and it is also easy to understand its underlying principle. A task having larger positional rank value means that this task and its successors require higher processing time and therefore occupy a higher number of stations. Thus, it is sound to assign the high positional weight tasks to earlier stations to compress all tasks into a minimum total number of stations.

Hoffman (1963) introduces a heuristic algorithm for SALBP-I. The procedure is based on the precedence matrix. A precedence matrix is a square matrix where entry *ij* equals 1 if task *i* precedes task *j*. Using this matrix, Hoffman generates feasible task combinations, for the station under consideration. Among the feasible combinations, the one with the lowest slack time is chosen for the station. This procedure tries to achieve lower slack time only for the station under consideration, so the procedure has no idea about the future performance of the succeeding stations.

Gutjahr and Nemhauser (1964) apply shortest path concepts to the SABLP. The nodes in the network represent the states. A state is a combination of tasks that can be processed without prior completion of any task and in any order that complies with the precedence relationships. Two nodes *i* and *j* are connected with a directed arc *ij* if

$$C_i \subset C_j \quad \text{and} \quad t(C_j) - t(C_i) \leq c \tag{2.1}$$

C_i being the tasks grouped in state *i*, $t(C_j)$ being the total processing time of tasks in state *j* and c being the cycle time. The arc *ij* is assigned a value which is equal to $c[t(C_j) - t(C_i)]$. These definitions of nodes and arcs lead to the following results. There is a one-to-one correspondence between paths from source node to sink node and feasible assignment of tasks to stations. Moreover, the length of any path with *n* arcs from source to sink is equal to:

$$n.c - t(C_r) \tag{2.2}$$

where $t(C_r)$ is the total processing time of all tasks. By definition, this value is equal to the total idle time for the corresponding task assignments. If the network is constructed as stated in the paper, then any path leading from node 0 to the final node has the least number of arcs and hence the total idle time is minimized.

Patterson and Albrecht (1975) improve the 0-1 formulation of the Bowman's model and introduce earliest and latest station concepts. Earliest (latest) station for a task is the station where that task cannot be assigned to a station earlier (later) than that station by the virtue of precedence relations. The introduction of these concepts increases the pre-processing of the problem while significantly reducing the 0-1 variables in the problem. The objective function also has a new concept. A *dummy task d* is created with zero processing time. This dummy task should be preceded by all tasks in the problem. The objective function is in the following form:

$$z = \sum_{j=E_d}^{M} (\ M + 1 - j) \tag{2.3}$$

A lower limit M' for M is calculated and this value is augmented by the following way until a feasible solution is obtained:

$$M_k = M^{'} - 1 + \ F_k \tag{2.4}$$

where F_k is the Fibonacci number.

Talbot and Patterson (1984) introduce an integer programming algorithm. They use an integer variable A_i rather than a binary variable which is equal to the index number of the station task *i* is assigned to. This reduces the total number of assignment variables from a much greater number to the number of tasks in the problem. They propose an augmentation procedure based on integer programming and enumerates all possible task assignments. Network cuts are introduced to the procedure to speed up the process.

Johnson (1988) introduces FABLE (Fast Algorithm for Balancing Lines Effectively) as a branch-and-bound procedure to find an optimal solution to the large scale SALBP-I. The procedure is capable of generating a feasible solution and even the optimal solution in a reasonable time. FABLE reaches a proven optimal solution

by enumerating, either explicitly or implicitly, a tree of feasible solutions which is proven by Jackson (1956) to include an optimal solution.

Hoffman (1992) introduces an exact branch and bound method for SALBP-I. In this method, every level in the tree refers to the station under consideration for task assignment. For every station, every possible task combination is generated and moved to the next station. At every level, total slack generated for every branch of the tree is calculated and the branches having total slack time higher than the theoretical minimum slack time is fathomed. The theoretical minimum slack time is calculated as follows:

$$\left\lceil \sum_i t_i / C \right\rceil * C - \sum_i t_i \qquad (2.5)$$

where t_i is the task time of task *i* and C is the cycle time. This search and fathom rule may end up with the totally fathomed tree. This means that there is no solution with this theoretical minimum total slack time, which is then incremented by an amount equal to the cycle time. The search continues with the previously generated tree that has the fathomed branches becoming unfathomed. This procedure is repeated until an optimal solution is found.

Boctor (1995) proposes a four-rule priority based heuristic approach to the problem. The task with the highest priority among the *schedulable tasks* is assigned to the current station until the last task is assigned. Before the priority rules are given, two definitions should be made. *Severe task* is a task having a processing time greater than or equal to one half of the cycle time. Secondly, a task is said to be a *subsequent candidate of task i* if it remains or becomes schedulable after assigning task *i* to the current workstation. If assigning *i* reduces the remaining time at the current workstation to zero, a task is said to be a subsequent candidate of *i* if it is schedulable for the next workstation. During balancing, the highest priority is given to:

- the task having processing time equal to the remaining time of the station under consideration. If there is no such task, the next rule is used. To break ties, the task with the largest number of 'subsequent candidates' is assigned.
- the 'severe' task having the largest number of 'subsequent candidates'. If there are no severe tasks, the next rule is used. In the case of a tie, the task with the longest processing time is chosen.
- the combination of two tasks having a duration equal to the remaining time. If there is no such combination, the next rule is used. As a tie breaking rule, the largest number of 'subsequent candidates' is used.
- the task having the largest number of 'subsequent candidates'. To break ties the task having the greatest number of 'severe' immediate successors is selected and if the tie persists, the task with the longest processing time is assigned.

The performance of the method is compared to fifteen other priority based heuristic approaches using two different problem sets.

Lapierre and Ruiz (2004) develop MS Access based software for a real world problem of SALBP-I type for the appliance manufacturing industry. The production of the appliance necessitates execution of tasks at two different heights at both sides of the assembly line. Furthermore, there are some subassemblies that are assembled off-line until the last task. The last tasks can be performed by the workstation on the assembly line and the subassembly is assembled to the main product. A test should be performed on the subassembly while it is assembled on the main product before the main product leaves the line. Due to rework considerations, subassembly line is faster than the main assembly line. Under these conditions, a software program is developed. Different heuristics are tried in order to assign the tasks to the available workstations. The results are presented and compared with the actual balancing already applied in the assembly line.

There are some survey papers on simple assembly line balancing literature (Baybars, 1986; Scholl and Becker, 2004).The studies mentioned above are generic in the problem domain, but recent studies are more specific in terms of the problem type. The studies on the mixed-model assembly line balancing problem (MALBP) and two-sided assembly line balancing problem (TALBP) are more relevant to our study. Hence, the literature on these areas will be presented in more detail.

2.2 Mixed-Model Assembly Line Balancing Problem

Thomopoulus (1967) is among the first to propose a method for the MALBP. The procedure is an extension of the heuristics developed for SALBP by Kilbridge and Wester (1961). Thomopoulus forms a combined task set and a combined precedence diagram over all models produced. Then, for every task of this set, the task's unit processing time for a model is multiplied by the total production quantity of the model per shift. Then this calculation is made for all other models and these figures are summed up. At the end, the result is equal to the total time required to complete this task for all models' total production quantity per shift. The heuristics of Kilbridge and Wester assign tasks based on the unit processing time of individual tasks, while in the modified version for MALBP, tasks are assigned based on the cumulative processing time for total production of all models per shift and the total available operating time per shift. The balancing procedure is the same as the original heuristics.

Thomopoulus (1970) proposes a technique to improve the balancing from the previous procedures with respect to the smoothness of the result. A mathematical term is formulated to measure the smoothness of a specific balancing. This formulation measures the difference between the average work content for a model on station and the actually assigned work content to a station. The revised balancing procedure is designed to minimize this term. Tasks are assigned to the stations in a serial manner. A finite number of iterations is

made to find the acceptable assignment combinations for each station. The smoothness term is calculated for each combination. Searching ends for a station when a certain iteration number is reached or the smoothness factor becomes zero for the current combination.

Macaskill (1972) is the first to make a formal definition of the combined precedence diagram. The balancing strategy is the one presented by Thomopoulus. The tasks are assigned to the stations based on the total labor hour requirement of the task per shift rather than the unit processing time and cycle time of the model. The focus of the paper is not on this procedure, but on the algorithm used to select the next task from the available group for the next assignment. The ranked positional weight method is selected and a computer program is developed to balance an assembly line with the method of Thomopoulus.

Gökçen and Erel (1996) propose a preemptive goal programming approach for MALBP. The program uses a combined precedence diagram. A task can have different processing times for different models, but it is assigned to the same station for all models. This is achieved through an assignment variable without a model index. The first goal of the program is keeping the total number of stations under a certain level. Second goal is achieving an upper bound for the cycle time for all models. The least priority goal is achieving to comply with a zoning constraint between two tasks.

Erel and Gökçen (1998) use ideas presented by the work of Gutjahr and Nemhauser (1964) to address MALBP. The conditions which are used to generate nodes and arcs are basically the same. Precedence relations are consolidated into a single combined diagram which is then used in the algorithm. The procedure now leads to a minimum total level of idleness summed over all stations and over all models in the problem.

Sarker and Pan (2001) present a study on open station, continuous line MALBP. In an open station, the operator is allowed to move outside of his station up to some certain limits without interfering with adjacent station's operator. A mathematical model is formulated to minimize total idle time and utility time. Utility time occurs when a worker cannot finish his job before the work unit leaves his station, thus additional workforce is needed to complete the job. The model includes decision variables on the physical properties of individual workstations and assembly line itself like launch interval between two consecutive jobs and length of a station.

Pastor et al. (2002) solve a real world problem of four-model MALBP for a white goods producer. The number of stations is fixed thus the problem is MALBP type II. The study includes a tabu search algorithm trying to minimize the cycle time for each model (maximize total output) and minimizing task dispersion, i.e., the same tasks of different models are tried to be assigned to the same station. Before the balancing procedure, a pre-processing stage is undertaken. The tasks which are to be executed together are merged, while the tasks with special requirements necessities are assigned to the stations having those tools. Then, the four previously defined heuristic rules are used to balance the line. The results are used to initiate the first tabu search with the aim of minimizing the cycle times. Two hundred feasible solutions are tested. These solutions are obtained from the combinations of the heuristics used, pre-assembly procedure and the order of models, etc. Nine results maximizing the total output are selected and further used for the next tabu search aiming to minimize the task dispersion. The most effective solution is chosen for the implementation and the company has achieved a %25 productivity gain with its new balanced production.

Simaria and Vilarinho (2004) present a detailed study on MALBP-II. In the first part of the paper, they formulate a 0-1 mathematical programming for the problem. The objective function is a summation of two values. The first one is

cycle time. The second figure is a factor taking values between 0 and 1 which shows how evenly the tasks are distributed among the available workstations. Zero value for the second factor shows that the total idleness is distributed perfectly among the available workstations. The mathematical program is not solved due to its nature and size. The second part of the paper presents a genetic algorithm (GA) based procedure for MALBP-II. Initially a lower bound is calculated for the cycle time. Before proceeding to GA, a simple constructive heuristic approach is applied for MABLP-I for the calculated lower bound and the resulting work force is compared to the available work force. If the solution is not feasible, the lower bound for the cycle time is increased by one and heuristics is applied again until a feasible solution is found. Next, GA-part is applied to the solution obtained from the constructive heuristics. This part decreases cycle time at each iteration by one to minimize the cycle time with the available work force. When a solution is reached, GA-II is initiated to smooth the assignments.

For further and detailed literature in mixed-model assembly line balancing problem review, the reader is advised to see the papers by Fokkert and Kok (1996) and Boysen et al. (2007).

2.3 Two-sided Assembly Line Balancing Problem

Bartholdi (1993) presents one of the first studies to address TALBP. A computer software program is developed which can balance one-sided and two-sided assembly lines. The program's balancing procedure is based on a modified "First-fit" rule. A list of schedulable tasks is formed. The tasks in this list are in the order which is the same as the order that the tasks are introduced to the software environment. The first task in the list is selected and assigned according to the modified first-fit rule. The software is developed to accommodate intensive user interaction. The user can change the order list of schedulable tasks, enforce zoning constraints and modify allowed work content for specific workstations

among other capabilities. These features are added to generate alternative balances with user intervention.

Kim et al. (2000) apply GA techniques to TALBP. The objective is to minimize the number of workstations. The steps of the genetic algorithm are presented with an encoding and decoding procedure of a solution to balancing problem. The procedure has an emphasis on positional constraints and designed to handle such situations. Experimental studies are made on small scale problems and the results are compared with the optimum solutions found with mathematical modeling.

Lee et al. (2001) propose a heuristic algorithm for TALBP. First they formulate two new performance measures for a line balance. The *work relatedness* measure is based on Agrawal's (1985) formulation. The figure measures how much the tasks assigned to a station are interrelated. If a task is reachable from another task through the precedence relations, then these two tasks are interrelated and assigning these two tasks to a single station is preferable with respect to the work relatedness measure defined. The *work slackness* is the other measure defined. It is asserted that slack time between the finishing time of a task and the starting time of its successor should be as much as possible. In case the preceding task delays, the succeeding task will not be affected if there is enough slack time. Increasing slack time can be achieved by rescheduling the tasks assigned to every station. After defining these new performance measures, the paper presents the details of the balancing heuristic procedure that starts by forming task groups. This is necessary since the procedure assigns tasks to stations in groups, not individually. Task grouping is again based on previous work of Agrawal. After group formation, an assignment strategy is developed. Finally these two procedures are incorporated into a single procedure for TALBP. In order to measure the performance of the group assignment procedure, a comparison heuristic procedure is formed based on the existing one-sided ALB heuristics. First, type-I formulation is checked for varying cycle times for three different

problems sets with 65, 148, 205 tasks, respectively. The results present the number of stations, the number of mated-stations and the performance measures defined by the authors. Second, type-II formulation is checked for varying numbers of stations. This time, the resulting cycle time and the resulting performance measure levels are presented for the three problems sets. The results show that the group assignment procedure increases the work relatedness and work slackness of the balance with little or no loss in cycle time or the number of work stations.

Kim et al. (2007) present a mathematical formulation and a genetic algorithm for TALBP. The mathematical formulation minimizes the cycle time over a given workforce. One of the decision variables is the assignment variable. Other decision variable is the finishing time of each task which is equal to the processing time of each task and the other tasks' processing times that are assigned to the same worker and finished before that task. The solution of the mathematical formulation conveys the information about the assignment of tasks and their sequences at the stations. The cycle time of each station, which the program seeks to minimize, equals the finishing time of the last task performed at that station. In the second part of the paper, a GA is presented. The algorithm uses a localized evolution that authors expect can promote the population diversity and the search efficiency. An initial population is generated and the member of the population is presented by a two-dimensional grid. A single member and its surrounding eight neighbors form the subject of the GA. An evaluation function is developed to measure the fitness of potential solutions. The algorithm creates better-fit generations based on the initial population of the nine members and the genetic factors formulated. The performance of the genetic algorithm is compared to the MIP formulation presented for three small-scale problems. Three large scale problems are compared to two different methods; one is another GA mostly based on the GA proposed by Kim et al. (2000) and the other one is the first-fit rule proposed by Bartholdi (1993). The results show that the

proposed GA is superior to both GA and the first-fit heuristic in terms of the solution quality and the convergence speed.

Simaria and Vilarinho (2007) introduce the ant colony optimization (ACO) technique into the mixed-model TALBP. First, they describe the problem via a mathematical modeling for type-I TALBP. Then ACO procedures are introduced. An initial colony of ants is divided into sub-colonies. Every pair of ants belonging to these sub-colonies produces a solution for TALBP. After every pair of a sub-colony produces an instance of line balancing, the best solution among these solutions is recorded and the pheromone levels are updated. The pheromone is a measure of attractiveness between two tasks. At the beginning of the first balance, a certain amount of pheromone is introduced between every task pair. Then, at every balancing iteration, the initially introduced pheromone level is vaporized by some percent and some additional pheromone is introduced if two tasks are assigned one after another for that individual balancing solution. The ants use this pheromone level information and heuristics information to assign tasks to stations. Heuristics information refers to the priority levels of tasks depending on the general priority rules like positional ranked weight or the number of successors of a task. The final sub-colony uses the cumulative pheromone levels and the heuristics information to find the best solution.

Baykasoğlu and Dereli (2008) also use ACO for TALBP. The objective is to minimize the number of workstations for a given cycle time and secondarily maximize work relatedness where possible. Work relatedness is measured again by Agrawal's formulation. The study has special emphasis on zoning constraints and the algorithm is developed to handle such constraints.

Hu et al. (2008) proposes a heuristic procedure depending on the concepts of *earliest start time* and *latest start time*. Since a task cannot be started before all of its predecessors are finished, earliest start time is determined by the finishing time of the predecessors of a task. Again, a task's latest start time is dependent

on its successors start time and its own processing time. These concepts are used to develop a heuristic to assign tasks. At one time, only a position is considered. Position is defined as a mated-station. The procedure may lead to infeasible solutions with respect to the precedence relations. A backtrack mechanism is suggested to remove the infeasible assignments. For type-I problems, a lower bound for the number of workers is formulated. The heuristic is used to solve type-I problems with varying cycle times and the results are compared to the lower bound values.

Özcan and Toklu (2008) propose a mixed integer goal programming model for TALBP. The objective is to minimize the deviations from the three different target values in a lexicographic order: number of mated stations, cycle time and number of tasks assigned to a workstation. Aspiration levels are to be defined by users. In the second part of the paper, fuzziness is introduced into the problem. In the fuzzy environment, objective function is to maximize the weighted addition of fuzzy goals.

Özcan (2008) presents a study on mixed model two-sided assembly line balancing. The study includes a mathematical model and a simulated annealing algorithm. Mathematical model objective function tries to minimize the line length as the primary objective and tries to minimize the number of workstations as the secondary objective. The model includes positive and negative zoning constraints, fixed location constraints and synchronous task constraints. Simulated annealing approach has an objective function value composed of two parts. First part is the weighted line efficiency and the second part is the weighted smoothness index. Both terms are defined such that the line efficiency is maximized and work load is distributed evenly among existing stations. Moreover, the number of stations opened is minimized due to the nature of these definitions.

CHAPTER 3

MATHEMATICAL MODEL FOR THE MIXED-MODEL TWO-SIDED TYPE-I ASSEMBLY LINE BALANCING PROBLEM

In this chapter a mathematical model is developed to solve TALB type-I problem. The objective of the model is to minimize the workforce necessary to assemble the models on an existing assembly line with a fixed length, given a production rate, i.e., tact time or a target cycle time. The mathematical model is developed to assign each repetition of every task to the same side of the same mated station for all models. Synchronous tasks are allowed in the model, but neither positive nor negative zoning constraint exists. There is no parallel station; therefore, the cycle time of the line has to be always greater than the maximum task time among all tasks. First we introduce necessary terminology, assumptions and notation. Then we present the model together with the validation and the parametric testing of its performance. Four different parameters are selected and the performance of the model is tested against these parametric changes.

3.1 Terminology, Assumptions and Notation

3.1.1 Terminology

Task: Indivisible work element.

Left (Right) Task: Task that should be performed at the left (right) side of the line.

Either Task: Task that can be performed at either side of the line.

Synchronous tasks: A right task and a left task that should be started at the same time at the same mated station.

Model: A product or one of its variations, i.e., a product which has a distinctive and unique blend of features making it different from other products.

Station: Location on the assembly line where some certain tasks are done repeatedly.

Task Time: Duration necessary to perform a task.

Cycle Time: Available time of a worker to perform tasks assigned to him/her.

Precedence Relations: Relations among tasks restricting the order of execution of tasks, defined by the nature of the tasks and the products assembled.

Precedence Graph: A network based representation of precedence relations. The tasks are represented by nodes and an arc between two nodes represents the precedence relation between those tasks. Further information on tasks can also be included in the graph. An example of a precedence graph for 6 tasks is given in Figure 3.1. This graph illustrates both the task time and side preference of each task (Right, Left, Either) given in parenthesis above each task's node.

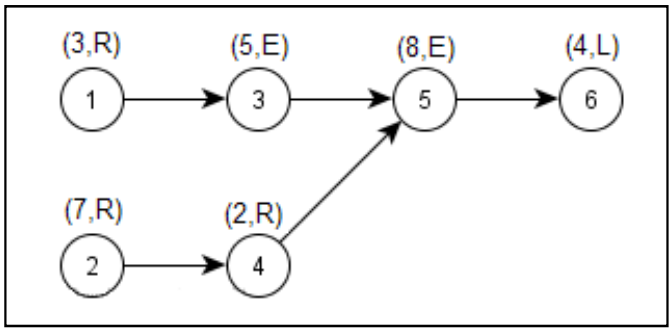

Figure 3.1 Sample Precedence Graph for TALBP

Combined Precedence Graph: Precedence graph that includes all tasks of all models and relations among all tasks.

Precedence Matrix: Precedence matrix is the illustration of the precedence relations in a matrix format. For a problem with *N* tasks, an *N***N* upper right matrix is formed. Entry (*i*, *j*) is 1, if task *i* precedes task *j*, 0 otherwise. A sample precedence matrix is given in Table 3.1 for the precedence graph in Figure 3.1.

Table 3.1 Sample Precedence Matrix for TALBP

Task	1	2	3	4	5	6
1	-	0	1	0	0	0
2		-	0	1	0	0
3			-	0	1	0
4				-	1	0
5					-	1
6						-

Predecessors of a task: Set of tasks that must be completed before starting time of the task. For example, the predecessors set of task 5 is {1, 2, 3, 4}.

Immediate Predecessors of a task: Set of tasks that must be completed immediately before the task starts. For example, the set of immediate predecessors of task 5 is {3, 4}.

Successors of a task: Set of tasks that cannot be processed before the completion time the task. For example, the successors set of task 2 is {4, 5, 6}.

Immediate Successors of a task: Set of tasks that cannot be processed immediately before the completion time of the task. For example, the immediate successors set of task 2 is {4}.

3.1.2 Assumptions

These assumptions are the common assumptions for the mathematical models and algorithms for TALB. Further assumptions will be introduced when required.

- Task times are deterministic. The same task may have different processing times for different models.
- All stations are equally skilled with respect to the work force.
- The assembly line produces several models of a certain product. The models are derivations of a basic model and models have many common tasks.
- The precedence graphs for all models are known and fixed. These individual precedence graphs can be combined into a single combined precedence graph.
- The tasks of all models can be incorporated into a single task group.
- The line length is fixed and the balancing is made for an existing assembly line with given number of mated stations.

3.1.3 Notation

The following notation is used throughout the text. Further notation will be introduced when required.

Indices:

i, j : Task indices $i = 1, \dots, N\ ;\ j = 1, \dots, N$

k : Mated station index $k = 1, \dots, K$

d : Side index $left = 1, right = 2$

m : Model index $m = 1, \dots, M$

Sets:

U : Set of tasks over all models

I : Set of mated stations

O : Set of models

L : Set of left tasks over all models

R : Set of right tasks over all models

E : Either type tasks over all models

P_i : Set of immediate predecessors of task i

S_i : Set of immediate successors of task i

SA_i: Set of successors of task i

SYN : Set of pairs of synchronous tasks over all models

Parameters:

C : Common cycle time for all models; a function of tact time

t_{im} : Task time of task i for model m

G : Big positive number

Decision Variables:

Y_{ik1}: $\begin{cases} 1, & \text{if task } i \text{ is assigned to left side of station } k \\ 0, & \text{otherwise} \end{cases}$

Y_{ik2}: $\begin{cases} 1, & \text{if task } i \text{ is assigned to right side of station } k \\ 0, & \text{otherwise} \end{cases}$

Z_{k1}: $\begin{cases} 1, & \text{if a worker is assigned to the left side of station } k \\ 0, & \text{otherwise} \end{cases}$

Z_{k2}: $\begin{cases} 1, & \text{if a worker is assigned to the right side of station } k \\ 0, & \text{otherwise} \end{cases}$

X_i: Records the station number to which job i is assigned

W_i: Records the side of the station to which job i is assigned, equals 1 if job i is assigned to the left side, 2 otherwise

T_{im}: Starting time of task i for model m

v_{ij}: Binary variable deciding the order of execution of tasks assigned to the same worker and do not have precedence relation in betweeen

3.2 Mathematical Modeling

A mathematical model is developed to obtain a balancing which seeks to minimize the number of workers for a fixed line length with *K* mated stations.

Objective

$$min \quad \sum_{k=1}^{K} (Z_{k1} + \ Z_{k2}) \tag{3.1}$$

Constraints

$$\sum_{d=1}^{2} \sum_{k=1}^{K} Y_{ikd} = 1 \quad for \ \forall i \ \in U \tag{3.2}$$

$$\sum_{d=1}^{2} \sum_{k=1}^{K} k * Y_{ikd} \ = X_i \quad for \ \forall i \ \in U \tag{3.3}$$

$$\sum_{d=1}^{2} \sum_{k=1}^{K} d * Y_{ikd} = W_i \quad for \ \forall i \ \in U \tag{3.4}$$

$$\sum_{k=1}^{K} Y_{ik2} = 0 \quad for\ \forall i\ \in\ L \tag{3.5}$$

$$\sum_{k=1}^{K} Y_{ik1} = 0 \quad for\ \forall i\ \in\ R \tag{3.6}$$

$$X_i * C +\ T_{im} +\ t_{im} -\ X_j * C -\ T_{jm}\ \leq 0 \tag{3.7}$$

$$for\ \forall i,j\ \in U:\ S_i \neq \emptyset\ ,j\ \in\ S_i\ \ ;\ for\ \forall m\ \in O$$

$$(2 * X_i - W_i) * C +\ T_{im} +\ t_{im} -\ (2 * X_j - W_j\ ) * C -\ T_{jm} \leq G.v_{ij} \tag{3.8}$$

$$for\ \forall i,j \in U:\ (i,j) \notin SYN\ ,j \notin\ SA_i\ ,i < j\ ;\ for\ \forall m \in O$$

$$(2 * X_j - W_j)C +\ T_{jm} +\ t_{jm} - (2 * X_i - W_i\ )C -\ T_{im} \leq G * \left(1 - v_{ij}\right) \tag{3.9}$$

$$for\ \forall i,j \in U:\ (i,j) \notin SYN\ ,j \notin\ SA_i\ ,i < j\ ;\ for\ \forall m \in O$$

$$X_i * C +\ T_{im} -\ X_j * C -\ T_{jm}\ = 0 \quad for\ (i,j)\ \in SYN\ ;\ for\ \forall m \in O \tag{3.10}$$

$$\sum_{i=1}^{N} Y_{ik1} * t_{im} \leq\ Z_{k1}\ *\ C \quad for\ \forall k \in I\ \ ;\ for\ \forall m \in O \tag{3.11}$$

$$\sum_{i=1}^{N} Y_{ik2} * t_{im} \leq\ Z_{k2}\ *\ C \quad for\ \forall k \in I\ \ ;\ for\ \forall m \in O \tag{3.12}$$

$$T_{im} + t_{im} \leq C \qquad for\ \forall i \in U\ ;\ for\ \forall m \in O \tag{3.13}$$

$$T_{im} \geq 0 \qquad for\ \forall i \in U\ ;\ for\ \forall m \in O \tag{3.14}$$

The objective (3.1) minimizes the number workers assigned to the sides of mated stations. Constraint (3.2) ensures that every task is assigned to one side of a station. Constraint (3.3) records the station number to which task *i* is assigned. Constraint (3.4) records the side of the station to which task *i* is assigned. Constraint (3.5) and (3.6) ensure side compatibility of the tasks assigned to the stations. Constraint (3.7) states that starting time of a succeeding task should be equal to or greater than the finishing time of the preceding task. Constraint (3.8) and (3.9) schedule the starting and finishing times of the tasks assigned to the same worker and have no precedence relation in between. Constraint (3.10) ensures that synchronous tasks are assigned to the same mated station and their respective starting times are equal to each other. Constraints (3.11) and (3.12) ensure that if at least one task is assigned to a work station, then a worker should be present at that work station. Constraint (3.13) ensures that every task is finished before the end of the cycle time. Constraint (3.14) ensures that starting time of each task for each model can take only non-negative values.

All other variables should also take non-negative values, but it is not necessary to force this by explicitly constructing a constraint since they can only take on non-negative values by definition.

At this point, it is necessary to elaborate on the *interference* phenomenon of TALBP. When balancing a one-sided assembly line, the tasks are distributed among stations according to the precedence relationships satisfying the cycle time constraint. The worker can perform the tasks without any delay one after

another respecting the precedence relationships. The finishing time of the last task assigned to a worker is equal to the sum of the processing times of all tasks assigned to that worker. But this situation is not valid for a two-sided assembly line balancing. The phenomenon will be described through an example. The right side of the mated station depicted in Figure 3.2 is assigned task *i* and the left side is assigned task *j*. Assume that task *i* precedes task *j* and all other tasks that are assigned to left side succeed task *j*. Then the worker at the left side should wait until task *i* is completed by the right-side worker. The first four time units are lost as idle time due to the interference between tasks *i* and *j*. It is evident that one cannot be sure of satisfying the cycle time constraint by simply adding the processing times of tasks assigned to the left-side worker as four units of time were already lost.

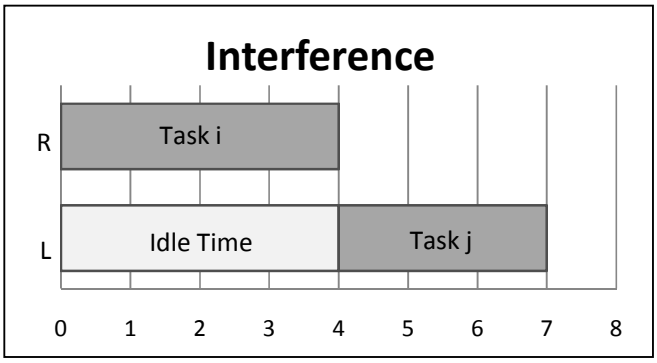

Figure 3.2 Illustration of Interference at a workstation

This phenomenon is the main distinction between one-sided assembly line balancing problem (OALBP) and TABLP. The tasks should be scheduled within the cycle time of every worker. The formulation of the mathematical model should include the capability of scheduling the task within the cycle time of each work station. The starting and finishing times of every task should be recorded to check

that duration of every task pair does not overlap. The constraints (3.7), (3.8) and (3.9) are introduced to the formulation with this aim. Constraint (3.7) is for tasks that have precedence relationships among each other. For such tasks, if they are both assigned to the same mated station, then the durations of these tasks should not overlap whether they are assigned to the same side of the mated station or not. Constraint (3.8) and (3.9) are for tasks that have no precedence relation. If such a pair of tasks is assigned to the same side of the same mated station, one should follow the other, and hence they should not overlap. These constraints decide the order of execution of tasks assigned to the same worker. One should note that only one of constraints (3.8) and (3.9) is active at a time for any two tasks that are assigned to the same worker depending on the order of tasks. Both constraints should be inactive for any two tasks that are assigned to different work stations. The use of these two constraints is achieved through the presence of parameter *G*. Assume a task is assigned to the left side of mated station 1 and another task is assigned to the mated station *K* right side. The level of *G* should be determined such that both constraints will be redundant for this couple of tasks. The following formulation is devised to determine the minimum level of G for given other problem parameters.

$$G \geq 2 * K * C \tag{3.15}$$

Before proceeding to the next section, we should mention that this mathematical formulation is not the first of its kind, at least in general approach to the problem. There are other mathematical formulations proposed by various studies for the mixed-model TALBP. The formulations are based on the binary formulation as our mathematical model. Although there is a similarity in general, the structure of constraints, especially the constraints that schedule tasks and avoids interference, is unique to this study. Any mathematical model may have pros and cons compared to the other formulations presented in literature. However, we

are not interested in making performance comparison with the other models. Rather, the performance of the model is searched in several problem environments.

3.3 Verification of the Mathematical Model

In this section, four literature problems are solved using the proposed mathematical model. These problems are widely studied in the TALBP literature. First problem is adopted from the study of Kim et al. (2007) which is a 24-task problem. The second problem and the fourth problems are obtained from Lee et al. (2001) with 65 tasks and 205 tasks, respectively. The third problem is adopted from the study of Bartholdi (1993) which includes 148 tasks. Table 3.2 summarizes the basic data on the test bed problems.

Table 3.2 Test Bed Problems

Test-Bed Problems	24-Task	65-Task	148-task	205-Task
Total Work Content (time units)	140	5099	5124	23363
Cycle Time (time units)	15	500	350	1500
Min. Line Length (mated stations)	5	6	8	8

Cycle times are set to approximately two times of the maximum duration task time among all tasks for each problem set. Total work content is the summation of processing times of all tasks.

The mathematical program needs an initial line length value in order to solve the problem. The sample problems do not specify a line length. The formulation, which is given in (3.16), is used to calculate an approximate value for the minimum line length.

Note that this value is an approximation for the minimum required line length to execute all tasks on the same assembly line. However, this formulation does not take into account the effect of interference, thus a longer line may be required. If there is infeasibility, the line length is increased gradually until feasibility is obtained.

$$LB_{LL} = \left\lceil \left\lceil \left(max_{\ m} \left\{ \sum_{i=1}^{N} t_{im} \right\} \right) / \ C \ \right\rceil /2 \right\rceil \tag{3.16}$$

Problems are formulated with GAMS 22.2 and solved with CPLEX 10.0 solver. Solution time is set to 10,800 seconds. Gap percentage is set to 0%. Gap percentage is the percentage difference between the incumbent integer solution found so far and the current level of lower bound on optimal solution calculated by the branch and cut procedure of CPLEX algorithm. CPLEX tries to increase lower bound and find better integer solutions simultaneously. The procedure ends when percentage difference between these two figures is proved to be zero or time limit is reached. Table 3.3 summarizes both the problem statistics and the solution statistics for all test-bed problems.

24-task problem is first solved with a line length of 5 mated stations; however an infeasible solution is obtained. Line length is increased by one. A line length of 6 mated stations for 24-task problem has turned an integer solution. The 65-task problem is solved again with a line length of 6 mated stations and an integer solution is found. 148-task problem and 205-task problem are solved with a line length of 8 mated stations. However, an integer solution is not found for both problems due to the problem size.

Table 3.3 Balancing Results of Test Bed Problems

	24-Task		65-Task	148-Task	205-Task
Line Length	5	6	6	8	8
# of Total Variables	597	647	2976	13534	24733
# of Discrete Variables	500	550	2780	13089	24117
Elapsed Time(seconds)	Infeas.	35.15	97.78	10800	10800
Objective Function	-	11	11	-	-
Absolute Gap	-	0	0	-	-
Relative Gap	-	0%	0%	-	-

Solutions to the test-bed problems verify that the mathematical formulation is correct and turns sound solutions to the problem. However, for larger problems it takes longer time to find an integer solution, as the number of variables increases exponentially with the increasing number of tasks. Processing times and precedence relations data of the test bed problems are presented in Appendix A while solution details are given in Appendix B.

3.4 Case Study

The mathematical model is also used to solve a real life problem. Problem is derived from a refrigerator production plant. Before the problem introduction, the main production processes of a refrigerator are explained.

The production processes mainly consist of the assembly of cooling system components and food storage components to the plastic body. It should be noted that the production processes may alter from one product to another, if products belong to different families of products.

Every refrigerator has a plastic inner cabinet. These cabinets are produced from plastic sheets. Plastics sheets are fed to a machine called thermoform machine. This machine is a big machine in terms of its size and cost. Every production line has two thermoform machines. These machines need molds to shape the plastic

sheets. For every product design, a mold is made. When a model is to be produced, the mold of the previously produced model is disassembled from the thermoform machine and the mold of the next product is mounted. This mold change is a machine setup and may last up to 45 minutes.

After plastic sheets are shaped into an inner cabinet with thermoform process, the cabinets follow the pre-polyurethane preparation line where the cable group and other components are assembled. If the product is a conventional one, its cooling system is assembled also on the pre-polyurethane preparation line. Along the pre-polyurethane line, products are transferred horizontally. Plastic cabinets of refrigerators are physically large and a worker standing on one side of the line cannot reach the other side of the product. Thus, workers are distributed at both sides of the line. Moreover, assembly of some parts requires two workers working simultaneously at the left and the right sides of the line. Hence, pre-polyurethane line is a two-sided assembly line.

As the side and back walls of the product are assembled at the end of the pre-polyurethane preparation line, the products enter the polyurethane plant where the inner volume between inner cabinet and walls are filled with polyurethane. This material is fluid, but it expands and solidifies in open air. This process is called vulcanization and lasts for 350 seconds on the average. The products leaving the polyurethane process enter the assembly line where it receives its cooling system (if it is a no-frost type) and the parts necessary for food storage (shelves, crispers etc.). The cooling gas is fed to the cooling system and the system is closed with welding process. The product goes through some quality control checks both visually and electronically. Its cooling performance is checked. Then, the doors of the product are assembled and the products leave the assembly line for packaging line.

Although the main production steps are the same for all products, there are quite different procedures for different products. First, if the product is conventional,

the cooling system components are assembled on pre-polyurethane preparation line. If it is a no-frost product, the cooling components are assembled on the assembly line.

Apart from the former classification, the whole product range can be grouped into two groups depending on whether it is an electronic product or not. Electronic products have a main board with electronic sensors and controllers with LCD displays. On the other hand, the mechanical ones have mechanical sensors and controls with no LCD display. On average, an electronic model has more tasks to be completed both on the pre-polyurethane preparation and the assembly line.

There are six different production lines each designed and equipped so that one or more product groups are produced. A product group consists of a main design and its varieties with respect to product size, color, accessories, etc. The main designs and their varieties are called “models”. At any time two models are assembled on the production line simultaneously. Although models are launched in lot sizes of 30-40 units on average, models intermix due to the repair loops and off-line processes on the line which accept products in lots but return products to the main line randomly. Thus, a mixed model line is present.

At this point, it is necessary to make clear why a mixed-model line is preferred although mixed-model lines have several disadvantages. First disadvantage it that it is harder to monitor the production flow in mixed production. As products use different components, more raw materials should be fed to the line at least in terms of material variety. Since different product models may require different workforce distribution along the production line, there may be more idleness compared to the case of a single model production. However, amid these disadvantages, more and more companies adopt the mixed-model production.

Considering this specific case study, if single model production is adopted, at least 2 thermoform molds and 16 polyurethane molds should be purchased, because the required production rate can be met only with this level of investment. However, if mixed-model production is adopted, then the production rate is halved for each model; thus one thermoform mold and 8 polyurethane molds are sufficient. By this way, the initial investment for each model is halved. Moreover, in single model production, the level of inventory for one model increases sharply in a short period of time, while other products that are not produced at that moment may suffer from shortage. Mixed-model production feed inventory levels of more products at a more moderate rate.

In the case study problem context, four models are chosen from the conventional models set. Two of these models have electronic cooling control systems, while the other two models are installed with mechanical cooling control systems. Thus, the models with electronic cooling control systems have higher work content than the other two models. All these models are produced on Production Line 4. The whole production process is not considered; only the pre-polyurethane steps of the production process are considered in our case study, as only this part of the production process features two-sidedness. Line 4 has 13 mated stations along the pre-polyurethane line. All stations are assumed to be equally equipped with respect to the production machinery and labour force. Although this is not the case in real life especially with respect to the machinery, this part of the production line does not feature high-cost machinery, so all the mated stations can be arranged according to a new balance of workload with only a small investment and effort. The workers mostly do manual tasks and only use machinery like glue guns, rivet guns, screwdrivers, etc. all of which are easily obtained in case of necessity and portable from one station to another.

As cited before, the pre-polyurethane line is a two-sided line and workers perform tasks at both sides of the station. A station is 2.1 meters long and two

workers may work at the same side of the station on the same product, thus four workers can work on the same product: two on the left side and two on the right side.

In its basic explanation, the pre-polyurethane line consists of conveyor belt units, each measuring 2.1 meters in length and all conveyor belts are connected to each other. A conveyor belt can contain one and only one product at a time. A conveyor belt is called “buffer” if it is not a location where tasks are performed on the product. Every conveyor belt has a key box. At this key box there are two options: manual pass and automatic pass. If the conveyor belt is chosen as buffer, then key is switched to automatic pass. The conveyor belts wait until the next location on the line empties, then passes the product on itself to posterior location on the line and receives one product from the anterior location if any product exists. If a conveyor belt is chosen as a station, then the key is set to manual pass. Then there should be another button on this belt. When workers finish their work on the product, they press the button and the belt gets the signal that the product is ready to go down the line. Since the stations are two-sided, there are two buttons on each side and for the transfer of product each button should send signals. It is obvious that the actual transfer of the product takes place when posterior location on the line is empty. These aspects of the line make it a stop-and-go line as pace of the line is determined by the pace of the workers. Workers are assigned tasks such that a certain cycle time is achieved and a pre-determined output is realized; however a stop-and-go line does not have a physically set cycle time and real cycle time is determined by the pace of the workers given all the other factors; for example, raw material feeding system does not affect the performance of the line. Two different products are fed to the line at the same time, thus mixed-model production is present. Since pre-polyurethane line is two-sided and mixed-model production is adopted, this production environment is chosen as our case study problem for mixed-model two-sided assembly line balancing.

3.4.1 Problem Data

Task times and combined precedence relations of the case study problem are presented on Table 3.4 and Table 3.5. Note that task times are in seconds. There are 74 tasks in the combined set of tasks among four models. Maximum task time is 24 seconds.

The production of any refrigerator model requires some basic common steps. Thus, most tasks are identical for all models and the precedence relations of models are mostly identical to each other. The differences among models' production processes come from the variability of task times for a common task, and additional tasks of a specific model due to additional features of the model, compared to basic models.

As mentioned before some tasks have different tasks times for different models. For example, the cooling parts are checked for cracks, since even an invisible crack on the cooling part may lead to leakage of cooling liquid circulating through the whole cooling system. The control is achieved through a machine that circulates an inert gas through the part and checks the pressure values. During this check, the machine sends a fixed amount of gas per unit time into the cooling part. These parts have inner volumes changing from model to model. Thus, the time that the machine spends to achieve a certain pressure value changes. Then, this quality control task has changing task times for different models.

Another variability source in the production process is the addition of some extra features to some products. As mentioned before, the case study includes four models produced on the same line. Two of these models have electronic cooling control systems, while the other two models are conventional products with mechanical cooling controls. Thus, electronic models and mechanical models have some tasks that are unique to these models.

Table 3.4 Task times of the Case Study Problem

Task	M1	M2	M3	M4	Task	M1	M2	M3	M4
1	10.69	6.83	9.75	13.61	38	0.00	0.00	0.00	4.72
2	4.28	4.28	4.28	4.28	39	0.00	0.00	0.00	3.24
3	8.11	9.84	9.84	9.84	40	0.00	0.00	0.00	4.12
4	3.48	3.59	3.59	3.59	41	0.00	0.00	0.00	2.40
5	1.80	1.80	1.80	1.80	42	0.00	0.00	0.00	4.81
6	0.00	0.00	0.00	4.32	43	3.48	3.48	3.48	3.48
7	3.55	3.55	0.00	0.00	44	3.68	3.68	3.68	3.68
8	11.26	0.00	9.60	9.60	45	3.68	3.68	3.68	3.68
9	2.49	4.33	4.33	4.33	46	6.13	6.13	6.13	6.13
10	2.93	2.93	2.93	2.93	47	2.93	2.93	2.93	2.93
11	2.93	2.93	2.93	2.93	48	4.30	4.30	4.30	4.30
12	16.46	16.46	16.46	16.46	49	2.16	2.16	2.16	2.16
13	3.95	3.95	3.95	3.95	50	2.57	2.57	2.57	2.57
14	24.00	24.00	24.00	24.00	51	4.50	4.50	4.50	0.00
15	24.00	24.00	24.00	24.00	52	2.15	2.15	2.15	2.15
16	4.34	4.34	4.34	4.34	53	0.00	0.00	0.00	2.46
17	5.56	5.56	5.56	5.56	54	4.50	4.50	4.50	4.50
18	2.85	2.85	2.85	2.85	55	2.24	2.24	2.24	2.24
19	6.29	6.29	6.29	6.29	56	2.40	2.40	2.40	2.40
20	21.25	21.25	21.25	21.25	57	2.88	2.88	2.88	2.88
21	3.38	0.00	0.00	0.00	58	3.09	3.09	3.09	3.09
22	6.76	8.42	8.42	8.42	59	6.36	6.36	6.36	6.36
23	3.03	0.00	0.00	0.00	60	3.29	3.29	3.29	3.29
24	0.00	3.68	3.68	3.68	61	2.24	2.24	2.24	2.24
25	0.00	0.00	0.00	3.76	62	7.54	7.54	7.54	7.54
26	2.74	3.43	4.15	4.41	63	2.32	2.32	2.32	2.32
27	9.99	9.99	9.99	9.99	64	1.71	1.71	1.71	1.71
28	3.19	3.19	3.19	3.19	65	1.61	1.61	1.61	1.61
29	1.09	1.09	1.09	1.09	66	1.93	1.93	1.93	1.93
30	2.70	2.70	2.70	2.70	67	3.46	3.46	3.46	3.46
31	2.57	2.57	2.57	2.57	68	7.54	7.54	7.54	7.54
32	22.20	22.20	22.80	22.20	69	6.67	6.67	6.67	6.67
33	22.80	22.80	22.80	22.80	70	2.93	2.93	2.93	2.93
34	3.31	3.31	3.31	0.00	71	3.46	3.46	3.46	3.46
35	4.12	4.12	4.12	0.00	72	5.41	5.41	5.41	5.41
36	2.40	2.40	2.40	0.00	73	2.40	2.40	2.40	2.40
37	4.81	4.81	4.81	0.00	74	2.55	2.55	2.55	2.55

Table 3.5 Precedence Relations of the Case Study Problem

Task No	Immediate Predecessors	Task No	Immediate Predecessors
1	-	38	-
2	-	39	32,33
3	-	40	16
4	3	41	40
5	3	42	14,15
6	1,3	43	14,15
7	1	44	25
8	1	45	25
9	-	46	1,22
10	-	47	43
11	-	48	14,15
12	-	49	43,48
13	-	50	43,48
14	12,13	51	50
15	12,13	52	51
16	14,15	53	52
17	14,15	54	52
18	14,15	55	53
19	17	56	52
20	19	57	52
21	3	58	52
22	20	59	52
23	22	60	59
24	22	61	59
25	22	62	60
26	22	63	62
27	22	64	62
28	27	65	62
29	21	66	62
30	23	67	63,64,65,66
31	30	68	67
32	20	69	68
33	20	70	48
34	29	71	63,64,65,66
35	16	72	69
36	35	73	69
37	35	74	69

3.4.2 Solution of the Case Study Problem

Every product group has its own production target for specific production lines. Some product groups can be produced by more than one production line. Then, production targets may change from one line to another, since the equipment of the lines are not identical. For four models that are included in the case study and considering that the assembly takes place on Line 4, the management sets the production target at 850 pieces (units) per shift.

A shift lasts for 480 minutes. However, 55 minutes are spared as allowance for lunch, tea time etc...Thus, production continues for 425 minutes. The management also sets Overall Equipment Efficiency (OEE) level at 85%. This means 15% of working time is considered to be the lost time. This further reduces production time to 361.25 minutes. These figures are used to calculate the cycle time by the following formulation.

$$Cycle\ Time = \frac{(Shift\ time - Allowance) * OEE}{Target\ Production\ Level} \tag{3.17}$$

This formulation leads to a cycle time of 25.5 seconds. This cycle time results in a total production of 850 units per shift. Cycle time is set to 25.5 seconds. There is an existing line with 13 mated stations. Thus, *K* is equal to 13. With these parameters and problem data, the case study problem is solved by the mathematical model and the following results are obtained.

Objective function: 17 workers
Solution time: 10800 seconds
Absolute Gap: 2.01
Relative Gap: 11.86%

The mathematical model returns an integer solution with a positive gap percentage therefore it is not the proven optimal solution to the case study

problem. This integer solution represents a line balance that can achieve the desired level of production with 17 workers among the existing 26 workstations. There are 9 workstations that are left empty. The resulting line configuration is depicted on Figure 3.3.

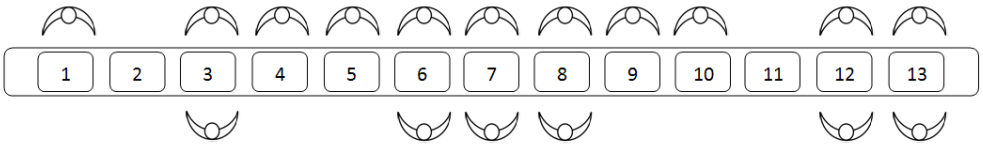

Figure 3.3 Workforce Distribution for LL=13 mated stations

Note that the solution has a positive gap percentage value and the solution process interrupts after 10,800 seconds. This means that the solution is not optimal and there is still room for improvement that can be achieved by relaxing the time trap and let the solution process proceed after 3 hours. However, this is not desirable. In real life, balancing is not made frequently. Rebalancing is made when demand patterns change or a new model is introduced. Thus, one may think that the management can wait for a long time for a solution to come up. However, it is not the case. Although balancing is not made frequently, management always expects a prompt solution.

There is another possibility of improvement for this case study problem. There are two unused mated stations at the existing assembly line. A shorter line may suffice. It is worth trying a shorter line length and solving the case study problem again, since the existing solution has a positive gap value and a better solution may be obtained. Line length is set to 10 and the case study problem is solved again. The results are as follows:

Objective function: 16 workers

Solution time: 10800 seconds

Absolute Gap: 1.01

Relative Gap: 6.36%

The solution again has a positive gap percentage value and solution procedure could not achieve zero gap % value within the allowed time limit. There is still room for improvement, but as cited before, a prompt solution is sought. Line lengths shorter than 10 mated stations turn no solutions within the time interval of 10,800 seconds. The new line configuration is presented on Figure 3.4. There are 6 right-side stations and 10 left-side stations. The details of the solution are given in Appendix B.

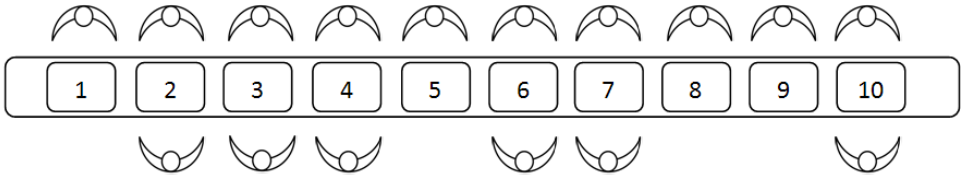

Figure 3.4 Workforce Distribution for LL=10 mated stations

Shorter line results in a better objective function value, since the number of integer variables, i.e. problem size decreases. However, a higher objective value may be obtained at the end of this change. As explained before, for a shorter line, occurrence of interference is more likely, especially for high order strength (OS) problems, i.e. more tasks are interrelated and this may lead to a higher objective function value. At this instance, the case study problem has a very low OS value, thus, an increase is not observed in the interference occurrences among tasks assigned to the same mated station.

3.5 Computational Experiments

In this section, the performance of the formulation is tested against several problem parameters.

3.5.1 Experimental Parameters

There are various factors that may have effect on the performance of the formulation and the resulting solution. However, only the following parameters are studied.

Number of tasks: The numbers of tasks are selected as 20, 30, 40, and 50. It should be noted that the problem may have multiple models and the number of tasks are the union of all tasks among all models in the problem.

Cycle Time: Cycle times for each model in the problem are taken as being equal to 2.5 times and 3 times the maximum task time of the model. Task times will be uniformly distributed between 10 and 100 seconds. Thus, cycle times are set to 250 seconds and 300 seconds respectively.

Order Strength (OS): OS is a measure of the complexity of precedence relations. Mathematically, it equals to the summation of number of successors of all tasks divided by $N*(N-1)/2$. In this experimental study, four levels of OS are studied for each task group which are 0.2, 0.4, 0.6 and 0.8.

Side Preference Freedom (SPF): This parameter has never mentioned in any study. It shows the number of either-side tasks as a ratio of total number of tasks. For experimental runs, the selected SPF levels are 0.3, 0.5, and 0.7. For each problem, the tasks that are not either side are distributed evenly among right side and left side.

By changing the experimental parameters, 96 test problems are generated. The experimental parameters are presented on Table 3.6.

Table 3.6 Experimental Parameters

	# of Tasks	Cycle Time	OS	SPF
Levels	20	250	0.2	0.3
	30	300	0.4	0.5
	40		0.6	0.7
	50		0.8	

The number of models in the problem is considered as an experimental parameter. However, the number of models only affects the task times, since a task may have different processing times for different models. The number of models in all experiment problems is set to three models.

Line length is another parameter that has an effect on the performance measures. However, it is not directly studied. As cycle time changes in the problem set, line length is altered. Moreover, as the number of tasks is increased, the line length is increased accordingly. Table 3.7 summarizes the line length values.

Table 3.7 Line Length Values

	Line Lengths	
# of Tasks	CT=250 seconds	CT=300 seconds
20	5	4
30	6	5
40	8	6
50	10	8

Line length values are determined such that all problem sets have feasible solutions. Lower OS value and higher SPF value problems need shorter lines, however high OS and low SPF value problems need longer lines. Thus, line length

is determined according to the longest line necessary to have feasible solutions for all the problems generated.

3.5.2 Performance Measures

The performance measures evaluated are defined below.

Number of workers: the objective function value

Solution Time: the elapsed time of the GAMS Cplex solver until an integer solution is returned

Gap percentage: The non-negative difference between lower bound on objective function value calculated by the branch and cut procedure of CPLEX solver and the integer solution obtained from the mathematical formulation; calculated as a percentage of the lower bound

3.5.3 Experimental Results

Before going through the results, it should be noted that all problems are formulated with GAMS 22.2 and solved with Cplex 10.0 solver run on a computer with Core2Duo 1.83 GHz processor with 2 GB RAM. The Cplex solver has a branch and cut algorithm and needs at least one limit for termination of the procedure, otherwise it may take a very long time to come up with the global optimal solution. For the experimental studies two limits are set during execution. The first limit is the gap percentage limit. When the gap percentage reaches 0%, then branching stops and the best integer solution found so far is returned as the final solution. If gap % does not drop to 0%, then the search continues for 10,800 seconds. At the end of three hours, best integer solution found so far is returned as the final solution. Solution details of all problems can be found in Appendix B.

3.5.3.1 Effect of Parameters on the Number of Workers

The values of the number of workers are plotted against the parameter values on Figure 3.5. There are two problem instances for 50-task group with lowest OS and

highest SPF level where the proposed mathematical model fails to find any integer solution within given computational time limit.

The most apparent effect seems to be due to the number of tasks. As the number of tasks increases, the objective function value increases, which is an expected and obvious result. A greater number of tasks necessitates a greater workforce with the same cycle time, thus with the same production rate.

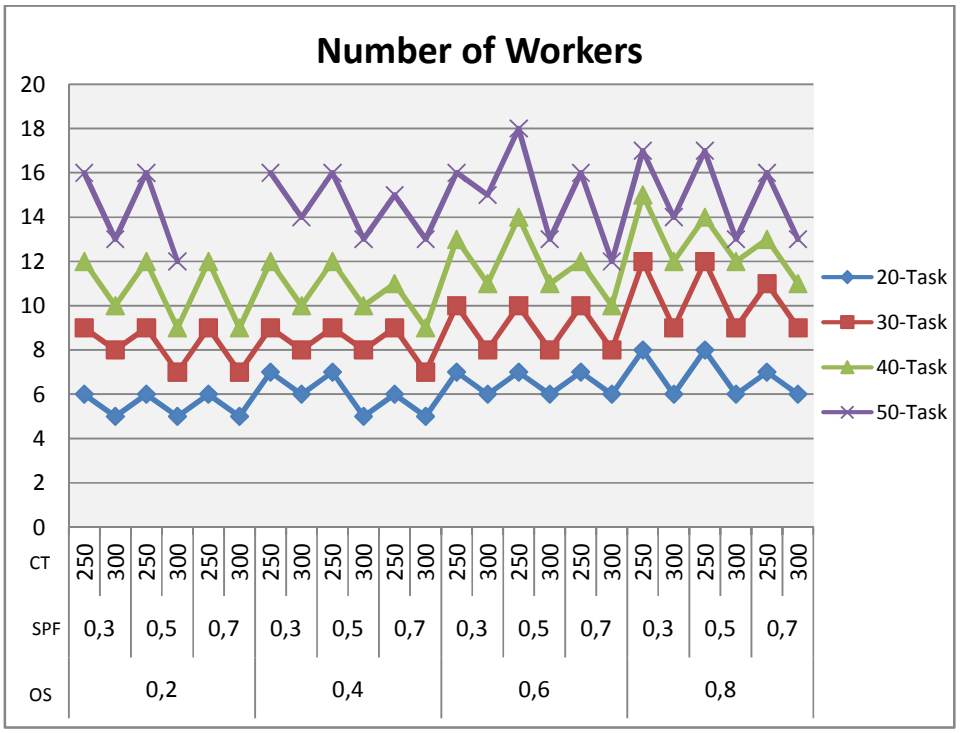

Figure 3.5 Objective Function Value against Changing Parameter Levels

The effect of cycle time is also obvious when the reaction of objective function value against changing cycle time level is considered. Higher cycle time level will allow workers to execute more tasks thus lower objective function values are expected for higher cycle time problems. This anticipated reaction is readily

observed in the results obtained from the experimental study problems. The number of workers and the cycle time parameters are studied to observe the effect on solution time and gap percentage values rather than the reaction of objective function value.

As mentioned before, OS is measure of complexity of precedence relations of tasks. The OS of a problem is higher when more tasks have precedence relations in between. If OS of any particular problem is equal to one, this means that there is always only one task that can be selected for the next assignment thus any efforts to optimize a type-I objective function are futile. On the other extreme, when OS is equal to zero of a particular problem, the balancing problem collapses into a bin-packing problem. The OS levels of all real life-problems are positioned between these two extremes. For two-sided lines, higher OS problems will lead to more idle time both due to restrictions posed by complex precedence relations when selecting the task for the next assignment and the higher number of instances of interference. Thus, higher OS problems are expected to have higher objective function values given all other parameters being equal. This effect is observed in the results obtained from experimental problems.

SPF has a diminishing effect on the objective function value. As SPF increases, i.e., either-side type tasks increases among all tasks, the objective function value decreases. Either type tasks can be used to reduce the idle time as they can fit in at both sides. Moreover, as they can be assigned to both sides, interference between tasks diminishes and again less idle time is created. However, as SPF value increases, the line becomes more of one-sided line. For a SPF value of '1', the line can be totally one-sided, but the resulting line length becomes much longer.

3.5.3.2 Effect of Parameters on the Solution Times

Solution time is another performance measure. Note that solution times are between 0 and 10,800 seconds, as the solution procedure terminates after

10,800 seconds if 0% gap is not attained yet. Figure 3.6 plots solution times against changing parameters values.

The problem sets with higher number of tasks have longer line lengths. Both line length parameter and number of tasks in the problem set will result in an increase in the number of binary variables in the problem. Thus, the scale of the problems gets larger for higher number of tasks in the problem. This leads to the expectation that larger scale problem sets will have solution times longer compared to small scale problems. The results obtained from experimental study confirm this expectation. 20-task problems have average solution time of 48.9 seconds while 50-task problems have 10800 seconds.

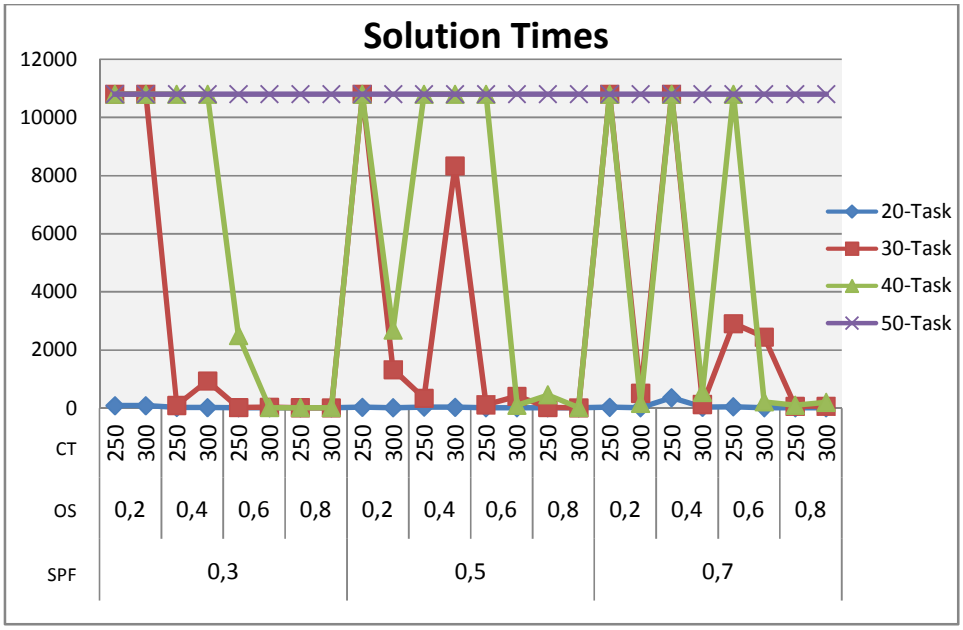

Figure 3.6 Solution Times against Changing Parameter Levels

There is a strong tendency towards longer solution times as the scale of the problem gets larger. However, the levels of other parameters result in high

variability in solution times. For instance, in 30-task problem set there are individual problems that have solution times as low as 13.26 seconds and as high as 10800 seconds.

OS has a diminishing effect on the solution times. High OS problems include less constraint (3.8) type and constraint (3.9) type since the order of execution of tasks are determined by the precedence relations in a greater extent for high OS problems and the mathematical model should make a smaller number of decisions on the scheduling of tasks assigned to the same work station. Thus the number of binary variable v_{ij} is less for high OS problems. Less binary variable results in a smaller search space and the solution times for such problems are shorter.

SPF parameter has an increasing effect on solution times. The mathematical model has a less number of constraints of type 3.5 and type 3.6, thus a larger solution space exists for larger SPF value problems. CPLEX branch-and-cut algorithm has a larger tree, since the model should decide on which side to assign the either-side tasks. For all problem sets, higher SPF value problems have average solution times longer than lower SPF value problems.

Higher cycle time problems have less binary variables, as line lengths are shorter; thus shorter solutions times are attained for 250-second cycle time problems. One should note that as problem size increases, the effect of parameters on the solution times diminishes, since the solution procedure cannot reach 0% gap and continues until time limit is reached. Thus, the effect of parameters is not observable.

3.5.3.3 Effect of Parameters on the Gap Percentages

Gap percentages are presented against several problem parameters on Figure 3.7. It should be noted that the solution procedure ends when 0% gap percentage is achieved or an elapsed time of 10,800 seconds is reached. The resulting gap

percentages are recorded. Small scale problems achieve 0% gap percentage more frequently within the time trap compared to large scale problems. For 20-task problems the mathematical model always achieved to find the proven optimal solution where gap percentage equals to 0%. On the other hand, for 50-task problems, the procedure never achieved to find the proven optimal solution. For larger scale problems, the search space is larger, thus the branch and cut procedure fails to search through the whole tree within given computational time. At the end of the time trap, since a part of the search tree is left unexplored, the procedure fails to verify that the incumbent integer solution is global optimal and the solution process ends up with a positive gap percentage value.

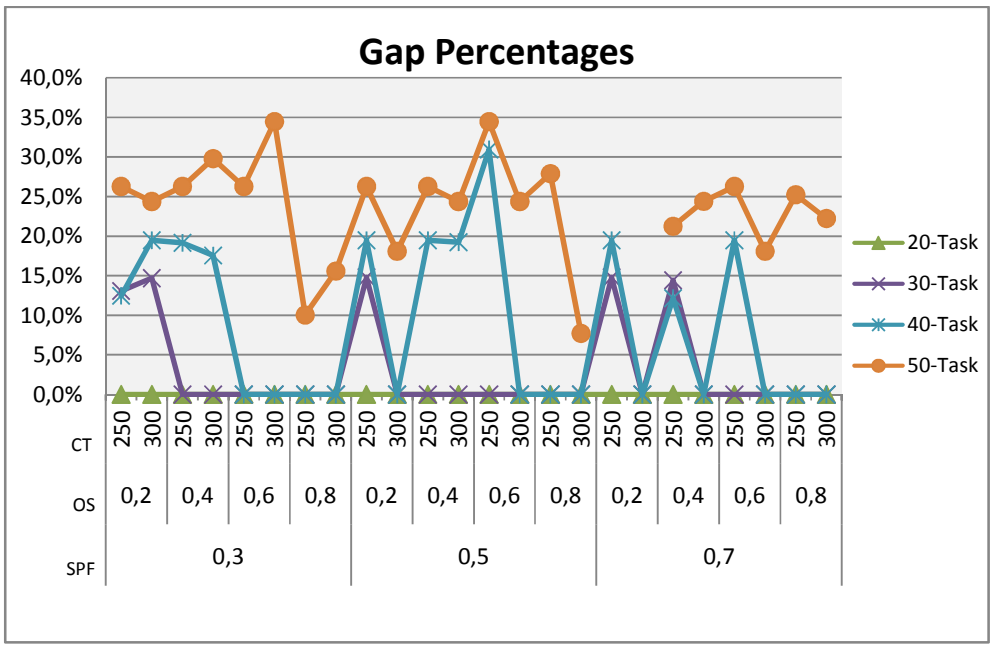

Figure 3.7 Gap Percentages against Changing Parameter Values

For higher cycle time problems, again the solution space is smaller since the problems with higher cycle time are solved with shorter line lengths. There are less binary variable Y_{iks} for these problems. Due to the reduction in binary variables in the problem, the search process is expected to end within shorter computational time. This expectation is verified with the results obtained from experimental study.

Higher OS problems have lower gap percentages. High OS problems results in less binary variable v_{ij} in the problem thus the corresponding search space is smaller. Thus, the procedure can explore the whole search space within given computational time for high OS problems and find the proven optimal solution.

Higher SPF problems should have a bigger search tree. Thus, solution process takes longer as shown in previous sections. Due to this fact, gap percentage which shows the distance between current solution and the best solution should be larger on average given fixed amount of computational time. However, high SPF values led to slightly smaller gap percentages. This effect may be attributed to the randomness present in the problem structure.

3.5.3.3 Overall Performance of the Mathematical Model

The effects of parameters on the performance measures of the mathematical model are studied in the previous sections. All parameters have significant effects on the performance of the mathematical model. All in all, the following conclusions can be drawn from the solutions of the 96 experimental problems' solutions:

- As the problem size gets larger, it takes more time to solve the problem. If the program software is given enough time, an integer solution with 0% gap percentage is obtained. If solution process is interrupted by a time trap, then a lower quality solution is obtained i.e., a solution with positive gap percentage.

- The number of binary variables is dependent number of tasks and line length. Line length is a function of task times and cycle time. Thus, scale of the problem is mostly dependent on number of tasks, task times and line length. As number of tasks and task time figures increase, the problem gets larger. On the other hand, higher cycle time leads to shorter line length and small size problems. Performance of the model is found out to be quite dependent on these parameters.
- Any constraint that prevents some binary variables becoming basic has a positive effect on solution time and solution quality given limited search opportunity. In our mathematical model there two groups of such constraints. The first group is precedence constraints. A complex precedence structure drops the possible number of integer assignment variables out of consideration during search process thus search process is more efficient and quick. The second type is task side preference constraints. As more tasks are non either-side tasks, these constraints again reduce the search space and more efficient solutions are possible.

It is important to elaborate on solution quality discussion. Our mathematical model will always find optimal solution given enough time. Optimal solution is the best quality solution and will have zero gap percentage value. On the other hand, in real life, we do not have infinite time thus an upper bound is set to limit the search process. The solution obtained when upper bound is reached is recorded as the best solution found so far. This solution has positive gap percentage thus there is a probability that the solution process ended up with a suboptimal solution. This solution is evaluated as a lower quality solution. Thus, solution quality is only considered when a time trap is set for search process.

CHAPTER 4

HEURISTIC APPROACH

In this chapter a heuristic approach is developed to solve especially the large scale problems. Note that as the problem gets larger, the mathematical model fails to return integer solutions within reasonable computational times. This heuristic approach is designed to provide a good solution within a reasonable computational time limit.

There are many studies in the literature proposing heuristic procedures for the two-sided assembly line balancing. Most studies propose meta-heuristic procedures. Recently population based algorithms are popular. There is an increasing number of population based algorithms which are inspired by instinctive behaviours of animals. Examples are Ant Colony Optimization (ACO) and Artificial Bee Colony Optimization (ABCO), the performances of which seem to be superior.

There are also single solution based algorithms. Most popular one is the Simulated Annealing (SA) approach. Threshold Accepting (TA) and General Hill Climbing (GHC) are the other single solution based algorithms. These three procedures share the basic rationale:

- All procedures start and work with a single solution.
- An initial solution is generated and this solution is recorded as the current solution S_C .
- Another solution is created by means of a perturbation made to the current solution. This new solution is recorded as the neighbour solution S_N . The objective function values are compared.

- If neighbour solution has an objective function value not worse than the current solution, then it is recorded as the new current solution.
- Otherwise, a decision criterion is checked. If this criterion is met, then the neighbour solution is again recorded as the new current solution. If not, the neighbour solution is discarded and a new neighbour solution is generated.
- The process is terminated when some kind of termination criterion is met and the best solution S_B is returned.

The basic idea that differentiates these algorithms from strictly greedy algorithms is that these algorithms allow transition to solutions with higher cost while searching through the solution space. This strategy avoids being trapped with the local optimal solutions.

In this study, single solution based procedures are selected to work with. First these procedures are easy to code and apply. The rationale behind these algorithms is easy to understand especially from an end user point of view. They depend on user intervention less than the other meta-heuristics, as few parameters are to be determined by the user.

The algorithm we propose in this study is based on Threshold Accepting (TA) which is first proposed by Dueck and Scheuer (1990). The basic TA algorithm is given in Figure 4.1. As explained before, the process starts and works with a single solution. The initial solution is recorded as S_C. A neighbour solution is generated and recorded as S_N. If the objective function value of neighbour solution, N is better than the objective function value of the current solution,C or worse within a predefined deterministic interval, T_C then the neighbour solution is accepted as the new current solution. The algorithm ends when some kind of termination criterion is met and returns the best solution, S_B found so far.

Note that the threshold can be altered during the course of the process. Most of the studies opt to decrease the threshold value incrementally every time the process generates a predetermined number of iterations and the process is terminated when the threshold value becomes zero.

1. Generate an initial solution and record it as S_C
2. WHILE Stopping criterion not met
   - 2.1 Generate a neighbour solution S_N
   - 2.2 IF $(N - C) < T_C$ then
   - 2.3 Set $S_C \leftarrow S_N$
   - 2.4 $C = N$
   - 2.5 EndIF
   - 2.6 IF $(C < B)$ then
   - 2.7 Set $S_B \leftarrow S_C$
   - 2.8 $B = C$
   - 2.9 EndIF
   - 2.10 Update T_C
3. End WHILE
4. Return S_B

Figure 4.1 Basic Threshold Accepting Algorithm

The advantage of TA can be observed when the solution process is trapped in a local solution surrounded by very high cost solutions. A series of uphill transitions should be made in order to escape from such a local solution. Since transition to worse solutions is probabilistic in both SA and GHC, many trials should be made to achieve enough uphill transitions. However, in TA the transition is deterministic. The number of trials required to escape from such local point in TA

is likely to be less. Thus, TA is chosen over other single solution based algorithms in this study. Another advantage of TA compared to all other meta-heuristics is that it requires less user intervention, as it requires determination of a few parameters. As it will be described in the following sections, the proposed procedure is developed such that it requires only a few user interventions.

Most studies that use TA algorithm in solving combinatorial optimization problems apply some kind of modification to the basic structure of the algorithm to achieve better convergence. Back Tracking TA (BATA) algorithm is one of those TA based algorithms and widely used in the literature. In this algorithm, a backtracking mechanism is devised. The algorithm is composed of repeated inner loops and an outer loop. Inner loops perform basic TA algorithm, while outer loop represents the threshold adjustment control mechanism. At the end of each inner loop, outer loop is executed. It is checked whether the solution process accepts an inferior solution at least once during the execution of the inner loop. If this check is satisfied, the threshold value is decreased by a predetermined increment. Otherwise, the threshold value is increased and the process continues with the next inner loop. This mechanism allows the algorithm to escape from local optimal solutions where no downhill movement is possible and no uphill movement is allowed by the current level of threshold.

Another modification is embedding a greedy local search sub-procedure into basic TA algorithm structure. A mechanism is devised such that the procedure can detect that there is a probability of consecutive downhill movements starting from the current point in the solution space. When such detection occurs, the procedure switches from TA algorithm to a strictly greedy algorithm for a given number of iterations to exploit the perceived opportunity of reaching better solutions starting from the current point.

As it is mentioned before, this will be the first application of a TA approach in TALBP. Thus, we decide to use the algorithm in its basic form. Moreover, the

modified versions of the algorithm may require more user interventions which is not desirable in the scope of this study.

4.1 The Threshold Accepting Algorithm for TALBP

Threshold Accepting has never been used for solving TALBP. To the best of our knowledge this will be the first example in the literature. The procedure presented here and most other studies presented in the literature cannot work directly with the problem. A decision should be made how to represent a solution during the execution of the procedure and another sub-procedure is necessary to convert an algorithmic representation of a solution into an actual line balancing instance. First procedure is called encoding and the second procedure is called decoding. Before presenting the details of the main procedure, these sub-procedures are presented.

4.1.1 The Decoding and Encoding Procedures

We should decide how to present a solution instance such that the proposed procedure can work with it. In this study, any solution is presented as a priority list. A priority list is a list that contains the unique priority values of the tasks and is ordered in increasing order with respect to the task number. An example is given in Figure 4.2.

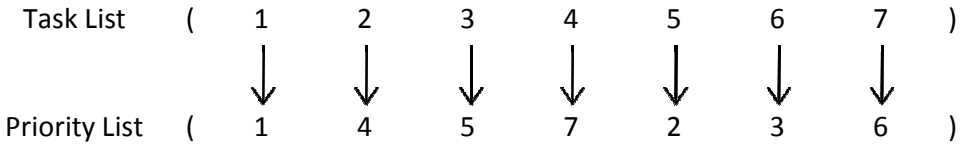

Figure 4.2 Priority List for 7-task Case

Every unique priority list corresponds to a unique solution. Note that unique solution does not necessarily mean a different objective function value. The

difference occurs in individual task assignments to individual stations or in the schedule of tasks within the cycle time of a workstation. This is encoding procedure for the proposed heuristic. The main procedure should also feature a decoding procedure. The decoding procedure is a procedure that transforms a priority list into an actual balancing instance for TALBP.

At this point, it should be noted that we focus on mixed-model assembly line balancing. However, in this study, the proposed procedure is designed to work with single task times. Hence, a mixed-model problem should be reduced to a single model problem. There are different approaches in reducing the mixed-model problem into a single model problem. For instance, some studies use task times averaged by their relative weights of each model's production quantity in total production quantity. However, this approach may lead to stations whose total workload is greater than the cycle time.

Another approach is to relax the assumption that each task should be assigned to the same mated station for all models. This relaxation reduces a mixed-model problem with *M* models into *M* independent SALBP instances. However, the assignment of the same tasks to different stations is usually not desired due to additional facility requirements, loss of specialization effects, complicated production control, and setup inefficiencies. Only in the case of multi-model production, where batches of models are processed, this relaxation of MALBP may be useful (Becker and Scholl, 2006).

This procedure enforces that total workload should never exceed the cycle time for all models. Thus, averaging task times cannot be considered. Moreover, our mathematical model assigns each task to the same station for all models. The advantages gained by assigning each task to a single station are considered important. Thus, reduction to a single model is achieved by taking maximum processing time of each task among all models. Thus, cycle time is never

exceeded for any assignment and procedure assigns each task to a single station for all models.

This decision reducing the mixed-model problem to a single model problem is widely discussed with other techniques presented in the literature in section 4.4. Figure 4.3 depicts the decoding procedure. As mentioned before decoding procedure is a procedure which receives a priority list and decodes it into an actual line balance. The detailed description of the decoding procedure is presented in Appendix C.

The procedure starts by opening a mated station, choosing right side and passing to the schedulable task formation step. This initialization is not depicted in the figure. At “Form schedulable tasks set” step, task list is filtered for the unassigned tasks that are side compatible with the current side under consideration. These tasks can start either at the current time of the side they will be assigned to or at a time later than this value, due to interference with facing side tasks. The possible starting time of each task is calculated. It is possible that a task in the set may not fit in the remaining time of the workstation, i.e. possible starting time plus processing time may exceed the cycle time. These tasks are eliminated from the set. Among the remaining tasks, the tasks with minimum possible starting times are also filtered. These tasks are tasks that can be started earliest for this station. Among these tasks, the one with the minimum priority value is selected for the next assignment to the current work station.

If the selected task is synchronous, two tasks should be assigned to the respective sides of the same mated station and they should start at the same time. If both tasks can fit in the current mated station’s remaining time for each side, they are assigned to the current station. If not a new mated station is opened and tasks are assigned to this mated station. If task has no synchronous task, it is directly assigned to the station under consideration.

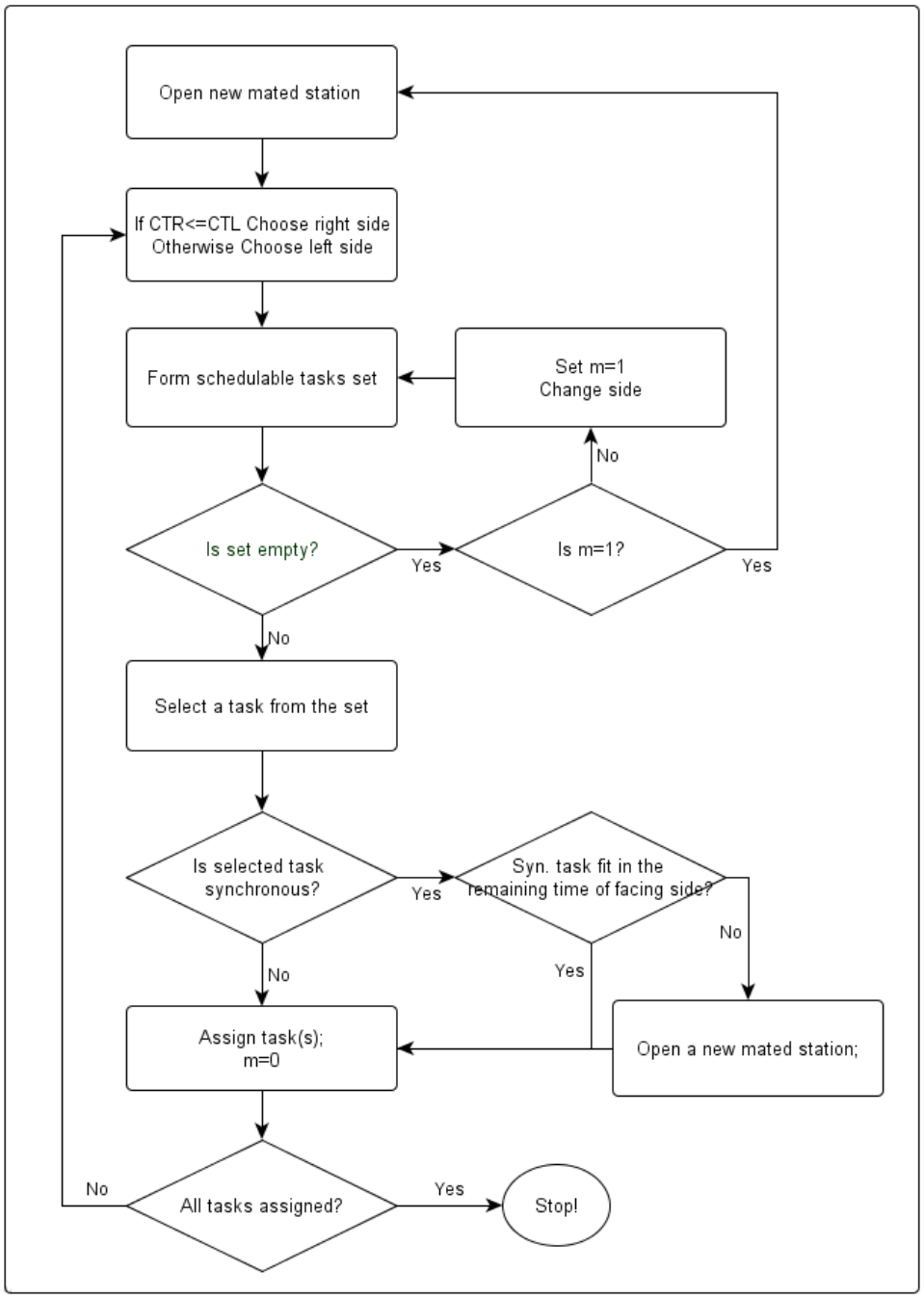

Figure 4.3 The Decoding Procedure

Note that no cycle time check is done at this stage, since the tasks that cannot fit in the remaining time are already eliminated at the previous stages. After an assignment is made, procedure continues to assign tasks to the side with the smaller current time. Note that CTR and CTL stand for the current times of right and left side of the same mated station.

At some point, the procedure fails to assign more tasks to the chosen side due to two reasons: i) no task is available due to precedence relations and side compatibility, ii) the available tasks cannot fit in the remaining time of the current side. At any case the procedure switches to the opposite side of the mated station and sets m=1. This variable records that the last assignment trial to the opposite side has failed. If procedure fails to assign any task to the new side, this means that a new mated station should be opened. However, if at least one task is assigned to the new side, then this means that two sides of the mated station may receive new tasks, because new tasks may become available for both sides. It is like the procedure shuttles between sides of the stations to make new assignments. If it fails for one side, it sets m=1; if it fails again when m=1, then a new mated station is necessary. Otherwise, it bounces between sides of the mated station to make new assignments. The creation of the new mated station is controlled in this fashion.

This is the outline of the decoding procedure. Note that the decoding procedure is a station based procedure rather than being a task based procedure. The task based procedures first choose the task according to some kind of criteria, then assigns the task to the suitable side of the station. On the other hand, our procedure first makes a side selection at the current mated station, then chooses the task with the minimum possible starting time, and among the tasks with the minimum possible starting time the one with the minimum priority. Even when a task has the minimum priority value, if its possible starting time is bigger than other tasks, then it is eliminated from the schedulable tasks set. This policy is

adopted to avoid interference related idle time. Since some smaller priority tasks lose their priority due to this policy, the priority values seem to play a secondary role in the selection of the next task for assignment. However, most of the time, there are more than one task with the minimum possible starting time, thus priority values still play a major role in task selection.

Given both encoding and decoding procedures, we can pass to the main Threshold Acceptance algorithm.

4.1.2 Main Algorithm

The algorithm presented in this study is based on basic Threshold Accepting. The procedure is composed of two phases: parameter setting phase and main phase.

Note that both in SA and TA, the parameters of these procedures are set such that at the beginning of the procedure, every generated solution is accepted regardless of the objective function value. This way a greater portion of solution space is visited. As procedure proceeds, the likelihood of accepting high cost solutions is decreased and eventually the procedure is expected to converge to a local optimal solution. This idea is applied also in this study. The initial threshold value should be set at such a value that at the beginning of the process every generated solution can be accepted. However, for any given problem, we do not have any idea about the topography of the solution space, thus the ideal initial level of the threshold value is unknown. Ideal initial level of the threshold should be high enough to accept every generated solution at the beginning of the procedure but not very high, such that the process lasts for a very long time. If we can estimate the maximum difference of objective function values between consecutively generated solutions, then the threshold parameter can be set by using the maximum difference level. The parameter setting phase is introduced to the procedure for this purpose.

The outline of the parameter setting phase is given in Figure 4.4. The procedure starts with a priority list where each priority value equals the task number. Starting from this solution, a series of neighboring solutions is generated for $oLIM$ times. The maximum absolute difference between consecutive solutions is recorded in variable max. Note that every generated solution is accepted as the new current solution regardless of its respective objective function value.

Parameter Setting Phase

1. $outer = 0$, $max = 0$
2. Generate a random solution and set as S_C
3. While $outer < oLIM$
4. $outer = outer + 1$
5. Generate a neighbour solution S_N starting from S_C
6. If $abs(N - C) >$ max then
7. $max = abs(N - C)$
8. End If
9. Set $S_C \leftarrow S_N$
10. End While
11. Return max

Figure 4.4 Parameter Setting Phase of TA Algorithm for TALBP

The absolute difference is recorded because of the reason that a reduction in the objective function value can be an increase in the objective function value, since current solution and neighbor solution are just definitions relative to where we start and where we are during the search. Thus, a movement from a current solution to a neighbor solution can be executed at opposite directions depending on where we start.

The main phase of the solution process is given in Figure 4.5. The procedure is in the form of basic TA. As explained before, we start with an initial solution and search through the solution space by means of perturbations made to the current solution. The initial solution for the main phase is the last solution generated by the parameter setting phase. A newly generated solution, which is called as neighbor solution, is accepted as the current solution if its cost is strictly smaller than the previous current solution cost plus current threshold value. The level of acceptance threshold is calculated by the formula given on line 4 in Figure 4.5. After the parameter setting phase, we have an estimation on the objective function value difference among neighboring solutions which is represented by max. Parameter LIM represents the limit of counter $inner$. When counter reaches this value, the threshold value should vanish and the process should terminate. The formula is formed to achieve this purpose. Suppose $LIM = 3000$ and $max = 3$, then the initial threshold value is 4 and it is decreased by one each time counter $inner$ is augmented by 750. If $LIM = 3000$ and $max = 4$, then the initial level of threshold is 5 and it is decreased by one for every 600 augmentation of counter $inner$. By this formulation, the threshold value is linearly reduced until it becomes 0, where the whole procedure ends. This is also the time when the total number of generated solutions equals the parameter LIM. Note that $\lfloor x \rfloor$ is equal to the next integer value smaller than x.

During the procedure, every current solution is compared with the best solution found so far and it is recorded as the new best solution, if its cost is strictly smaller than the previous best solution. Note that the user only needs to determine the level of parameter LIM, thus the performance of the algorithm is less dependent on user intervention.

Main Phase

1. Read max and S_C . $inner = 0$
2. While $T > 0$
3. $inner = inner + 1$
4. $T = (max + 1) - \lfloor inner/(LIM/(max + 1)) \rfloor$
5. Generate a neighbour solution
6. If $(N - C) < T$ then
7. $S_C \leftarrow S_N$
8. $C = N$
9. End If
10. If C $< B$ then
11. $S_B \leftarrow S_C$
12. $B = C$
13. End If
14. End While
15. Return S_B and B

Figure 4.5 Main Phase of TA Algorithm for TALBP

Up to this point neighbor solution generation is not specified. There are various neighborhood generation techniques. An effective generation scheme can significantly contribute to the performance of a heuristic procedure, as the solution space is investigated more efficiently. Tian et al. (1998) present a detailed study on different neighborhood generation mechanisms and their respective effect on solution quality. Example problems representing Travelling Salesman Problem (TSP), Flow-shop Scheduling Problem (FSP), and Quadratic Assignment Problem (QAP) are solved with SA and the performance of different schemes are compared. They propose six different mechanisms. These schemes are presented in Figure 4.6 through a 10-task problem. The figure is directly adopted from the mentioned study. The underlined numbers represent the individual numbers or subsequence of numbers affected by the perturbation.

In first scheme a random position is selected than its content is interchanged with adjacent position. In the second scheme, two different positions are selected and their content is exchanged. The third scheme chooses two positions. The content of the first position is then inserted left or right side to the second position. In forth scheme, two positions are selected and sequence between these two positions is moved. Note that the movement amount can be kept fixed for all movements or another random number can be generated to determine the amount of movement. In fifth scheme, two positions are selected and the sequence between these two positions is reversed. In sixth scheme, a sequence is selected. This sequence is both reversed and moved.

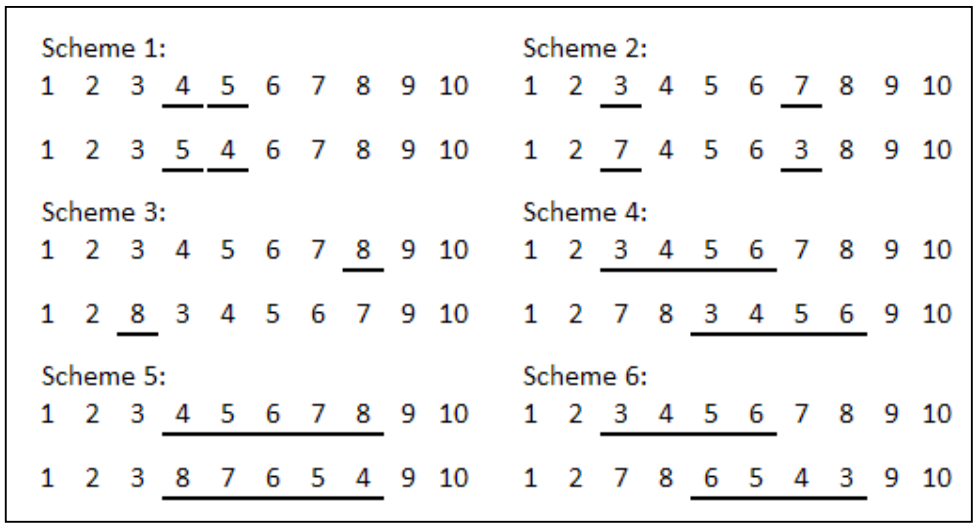

Figure 4.6 Neighbour Solution Generation Schemes

For our study we adopt the neighborhood generation scheme 5 which is proved to be a superior scheme in the mentioned study. In this scheme a subsequence of priority values is selected and the tasks which have these priority values are reversed. Note that the same neighborhood generation scheme is used for both the parameter setting phase and the main phase.

4.2 Verification and Performance Evaluation of Heuristic Approach

The mathematical model is used to solve the four test bed problems in order to verify the validity of the formulation. Case study problem is also solved with the mathematical model. These four test bed problems and case study problem are now solved with the proposed heuristic with the same cycle time for initial verification of the procedure.

For performance evaluation, more problem instances are solved for 65-task, 148-task and 205-task problems with different cycle times. These three problems are the widely studied problems in the literature. For performance comparison, the studies of Lee (2001), Baykasoğlu and Dereli (2006) and Simaria and Vilarinho (2007) are selected. Before presenting the solutions, a lower bound (LB) calculation method, which is introduced by Lee (2001), is presented. LB can be used to evaluate the performance of a procedure in case an optimal solution is not available for performance comparison.

$$LB = LB_R + LB_L + LB_E \tag{4.1}$$

$$LB_L = \left\lceil \left(\sum_{i \in L} \max_{\mathrm{m}} \{t_{im}\} \right) / C \right\rceil \tag{4.2}$$

$$LB_R = \left\lceil \left(\sum_{i \in R} \max_{\mathrm{m}} \{t_{im}\} \right) / C \right\rceil \tag{4.3}$$

$$LB_E = \left\lceil \left[\sum_{i \in E} \max_{\mathrm{m}} \{t_{im}\} - \left((LB_L + LB_R)C - \sum_{i \notin E} \max_{\mathrm{m}} \{t_{im}\} \right) \right] / C \right\rceil \tag{4.4}$$

Lower bound for any problem is composed of three parts: left, right and either. Lower bound for left side work stations is calculated by dividing the total work content of left-side tasks by the cycle time and taking the next integer larger than the calculated value. The same procedure is repeated for the right-side tasks. Either side tasks may fit in the remaining time in the right or left side stations after all right or left tasks are assigned to these stations. The minimum number of workstations to perform either-side tasks is calculated after slack time present in left or right side stations is deducted from the total work content of either-side tasks.

There is one single parameter that should be determined to start the procedure, which is parameter LIM. This parameter equals the number of iterations, i.e., the number of solutions generated during the procedure. Since problem size is a function of the number of tasks in the problem, the parameter can be determined as a function of the number of tasks as well. From this point of view, for all executions of the procedure, LIM parameter is determined by the following formula:

$$LIM = 50 * Number\ of\ Tasks \tag{4.5}$$

Parameter $oLIM$, the number of iterations made during parameter setting phase to estimate the initial threshold value, is determined as a function of the parameter LIM. It is set to 10% of parameter LIM for all executions of the procedure for the rest of the study.

There is a trade-off between solution quality and the solution time for the heuristic procedures like SA or TA. The procedure finds the global optimal solution as the search time converges to infinity. Of course, a balance should be established between solution quality and solution time. For this study, our priority is to find solutions as good as the best existing procedures provide for the

test bed problems and to keep the solution time at the minimum level. After determining the levels of parameters, the problems are solved and the results are presented on Table 4.1. The procedure is coded in Visual Basic 6.0 and run on a personal computer with 1,83 GHz Core2Duo® processor on 2 GB RAM.

For the 24-task problem, the procedure finds the optimal solution within an execution time of 2 seconds. For the 65-task problem, we cannot find the optimal solution, but the solution quality is comparable to the optimal solution. For the 148-task and 205-task problems the mathematical model failed to find any integer solution within time trap. However, the proposed heuristic procedure finds integer solutions for these large scale problems within reasonable solution times.

Table 4.1 Heuristic Solutions of Test Bed Problems

Test Problems	Optimal Solution	Optimal Solution Time (seconds)	Heuristic Solution	Heuristic Solution Time (seconds)
24-Task	11	35.15	11	2
65-Task	11	97.78	12	22
148-Task	-	10800	16	226
205-Task	-	10800	18	407
Case Study	16	10800	17	28

Note that, all test bed problems are single model problems, while the case study problem is a 4-model problem. The lower bound for the case study problem with maximum task times is 17 workers. Thus, the procedure finds the best possible solution for the case study problem. Moreover, the case study problem features two synchronous task couples. When actual task assignments are studied, it is observed that these tasks are successfully assigned to the opposite sides of the same mated station and their starting times are synchronous.

The validity of the heuristic procedure is verified with the results obtained with the four test bed problems and the case study problem. The details of the solutions are given in Appendix C. Now, performance of the proposed heuristic procedure is compared to the existing procedures. Lee (2001) proposes a procedure called Group Assignment Procedure while Baykasoğlu and Dereli (2006) and Simaria and Vilarinho (2007) both present procedures based on Ant Colony Optimization. Table 4.2 includes the results of our procedure and the results presented by the mentioned papers. GA stands for the results obtained with Group Assignment Procedure of Lee. ACO-I stands for the study of Simaria and Vilarinho, while ACO-II stands for the study of Baykasoğlu and Dereli. The results of GA and ACO-II are the results of multiple runs and the averaged objective function values are rounded down to the next integer value. ACO-I and TA, which are the proposed heuristics in this study, present the results of the 10 runs. The minimum, maximum and average objective function values are reported.

All procedures use random numbers during the neighborhood generation. Thus, the result of any run may depend on the series of random numbers generated. This is the reason why studies report results of multiple runs for each cycle time level of each problem. This way the performance of the proposed methods are freed from the effect of randomness introduced to the problem by the random number generation.

Our procedure has a performance comparable to the best existing procedure which is the ACO-based procedure of Simaria and Vilarinho (2007). Apart from the two cycle time levels at the 205-task problem, TA algorithm achieves to find solutions as good as the best literature-wide solution. For the 205-task problem and cycle time of 1322, we achieve to find the best literature-wide objective function value. Based on these results, we can conclude that the TA algorithm has proved its usefulness for TALBP.

Table 4.2 Performance Comparison of Heuristic Procedure

			Objective Function Values								Durations(s)	
			GA		ACO-I		ACO-II		TA			
	CT	LB	Avr.	Min.	Avr.	Max.	Avr.	Min.	Avr.	Max.	ACO-II	TA
A65	326	16	17	17	17	17	17	17	17.7	18	<1	22
	381	14	15	14	14.8	15	15	14	14.7	15	<1	22
	435	12	13	13	13	13	13	13	13	13	<1	21
	490	11	12	12	12	12	12	12	12	12	<1	22
	544	10	10	10	10.8	11	10	10	10	10	2.48	21
B148	204	26	27	26	26	26	26	26	26	26	4	249
	255	21	21	21	21	21	21	21	21	21	16	245
	306	17	18	18	18	18	18	18	18	18	51	259
	357	15	15	15	15.4	16	15	15	15	15	4	275
	408	13	14	14	14	14	14	14	14	14	2	278
	459	12	13	12	12	12	12	12	12	12	181	277
	510	11	11	11	11	11	11	11	11	11	15	282
A205	1133	21	23	22	22.4	23	24	22	22.2	23	451	384
	1322	18	20	20	20	20	22	19	19.4	20	449	399
	1510	16	20	17	17.2	18	18	18	18	18	288	376
	1699	14	16	15	15.8	16	18	15	15.5	16	448	402
	1888	13	16	13	13.8	14	15	14	14	14	178	415
	2077	12	14	12	12	12	14	12	12	12	7	425
	2266	11	13	12	12	12	12	12	12	12	131	418
	2454	10	12	10	10	10	12	10	10	10	7	412
	2643	9	12	10	10	10	11	10	10	10	69	411
	2832	9	10	10	10	10	10	10	10	10	304	416

The solution times are only reported by Baykasoğlu and Dereli. Thus, the only comparison is made with that study with respect to the solution times. The CPU times are reasonable enough and comparable with the existing procedures. From the practical perspective, the CPU times are quite competitive and reasonable for real life expectations. Note that CPU times are dependent on the parameter and the number of tasks in the problem.

4.3 Computational Experiments with Heuristic Procedure

The behavior of the mathematical model was studied with 96 randomly generated problems with changing problem parameter levels. These problems are also solved with proposed heuristics to see the behavior of proposed heuristic procedure. The number of workers values obtained with the proposed heuristic procedure is plotted against changing problem parameter levels in Figure 4.7. The individual results are given in Appendix C.

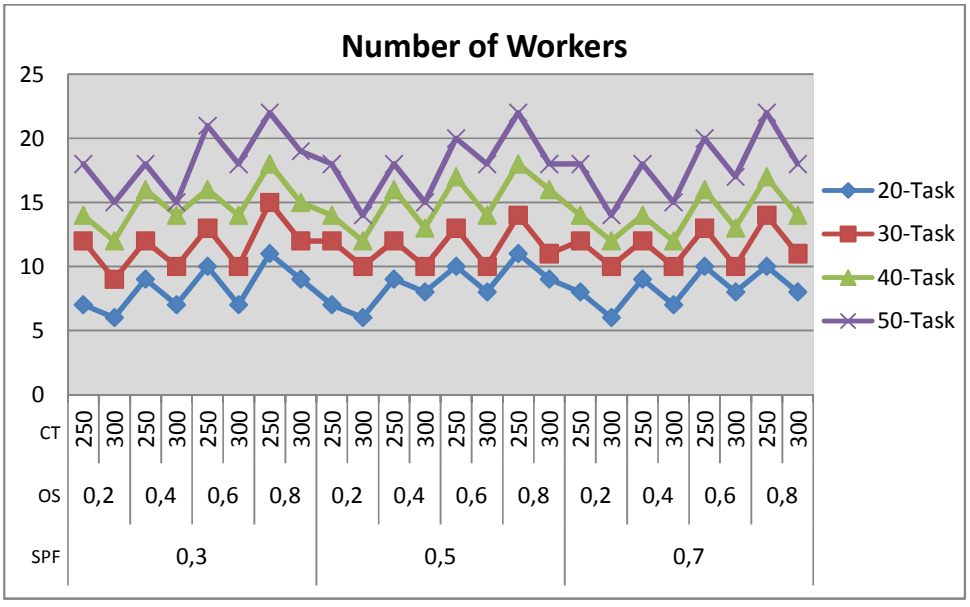

Figure 4.7 Number of Workers Figures obtained with Heuristic Procedure

It is easily observable that number of tasks in the problem and cycle time has the most apparent effect on the number of worker figures. OS has again significant effect on the resulting workforce figures. For every task group, higher OS problems resulted in higher number of workers figures.

SPF parameter is another parameter that has significant effect on objective function values and solution times for mathematical model. However, the effect of SPF is less apparent in the results obtained with heuristic procedure compared to the sensitivity observed in mathematical model solutions against changing levels of SPF parameter.

In this part, the effect of changing problem parameter levels is only studied with respect to the resulting number of worker figures. Solution length is not studied since the maximum solution time among all problem sets, which occurs for 50-task problem set, is only 8 seconds which is incomparably below the solution times observed with mathematical model.

4.4 Reduction of the Mixed-Model Problem to a Single Model Problem

The proposed heuristic procedure reduces mixed-model problems into single-model problems by taking maximum processing times for each task, and then solves the problem as a single model problem. Reducing mixed-model problem into a single model problem is a common practice in the literature. However, most studies opt to use the weighted average task times rather than the maximum task times. The weight of a product is determined based on the ratio of its production amount to the total production amount.

Averaged task times can lead to many practical problems when any solution obtained with these task times is put into application. For any given balancing procedure, while task assignments are made, the cycle time constraint is checked with the average task times and of course cycle time constraint is always satisfied

with the average task times. However, in the actual situation, the worker works with the individual task times of each model. It is likely that cycle time constraint is violated when exact task times are added up according to the assignments of the tasks. This cycle time violation is called overutilization or overloading. In other words, overutilization is the case where a worker is unable to finish processing all tasks assigned to the station during the cycle time. This situation is clarified with an example. Assuming that Task 1 and Task 2, whose task times are given on Table 4.3, are assigned to the same worker with given cycle time of 20 seconds. Note that the total of averaged task times satisfies cycle time constraint. On the other hand, for Model II, the tasks times of two tasks add up to 21 seconds. Cycle time violation i.e., overutilization, occurs for Model II.

Table 4.3 Example for Overutilization

Task Times (seconds)	Model I	Model II	Average
Task 1	4	6	5
Task 2	8	15	11.5
Total Workload	12	21	16.5

Different methods can be proposed to resolve the overutilization cases. First, a reserved workforce can be kept to resolve the overutilization instances. The reserved workforce is referred to when models leading to overutilization are produced on the line; and for the rest of their working time, they can be employed at some other parts of the production process. These workers may work on or off the line. If the line and station configuration allows extra workers to work simultaneously with the already existing workforce, then they are expected to work online. Otherwise unfinished products are transferred to a region off the line and the reserved workforce completes the unfinished (extra) tasks. The second case is undesirable due to handling costs.

The most important decision about the reserved workforce is to assign reserved workers to the right positions on the line and assigning the right jobs to these workers. There may be several overloaded stations on the line. A policy may be to assign a worker to each station where overutilization occurs. Two workers work on the same station and share the total work content. Since the station is two-sided, it is important to share tasks such that the other side of the station is not delayed. If consecutive stations are overloaded, then a single extra worker may work on these stations and perform extra tasks on these stations. However, this policy requires that the worker travels between these stations. If such a worker is assigned many tasks dispersed on many overloaded stations, some products processing may be split and delayed until the extra worker comes and finishes the respective tasks. This, in turn, may cause delay in the departure of the product from the station and hence the line performs below its aimed-at level. The most important aspect of this policy is to keep the tasks at the work station that they are assigned to at first; but a different worker performs the job.

A different policy may be to open a new workstation for every extra worker and assign extra tasks to these stations. An unfinished task may be delayed up to a station where a successor task is performed. Moreover, tasks from stations down the line can also be performed at this station, if a predecessor task is not in between this extra station and the overloaded station down the line. The location of the extra station can be chosen such that as many extra tasks as possible can be performed at this station. However, with this policy, the same task can be performed at several stations on the line.

Another approach to resolve overutilization may be to instruct workers to expedite their work for a certain period of time. This may work for minor overutilization cases. Every worker is assumed to be able to work at 100% pace and able to finish the assigned tasks in given cycle time. The 100% pace is the perceived performance level of an average worker under average working

conditions. Every worker may work at a higher pace, at least for a certain period of working time. Thus, minor overutilization cases may be ignored and assumed to be absorbed by the increased pace of the workers. Of course, a limit should be set. Assume that a limit is set and workers are assumed to be able to work at 110% pace. In this case, the worker can perform a work content of 28.05 seconds in 25.5 seconds. Overutilization cases up to this limit may be ignored.

Management may prefer doing nothing when overutilization occurs and let the line work with a slower pace. Overloaded stations then become the bottleneck sources of the assembly line and the whole line operates with a slower pace. The most overloaded station determines the pace of the line and all other stations, overloaded or not, stay idle. Note that balancing is done with the given cycle time at the beginning of the process; however, working with overloaded stations results in higher cycle times for some models. Therefore, idleness increases for such a line and the production output of the line turns out to be lower than anticipated.

Now, in order to clarify the practical problems of using average task times for line balancing problem, case study problem will be solved with averaged task times. In its original case, the total work content and individual task times of case study problem do not feature enough variability. In order to observe the effect of using averaged task times, the original task times of the case study problem is modified such that models have increasing work content from Model 1 to Model 4. Since comparison will be made with the line balancing obtained with maximum task times, the modification of task times are realized such that maximum task time for any given task does not differentiate from the original case study data.

4.4.1 Case Study

In its original form of the heuristic procedure, we solve the case study problem with the maximum task times; hence overutilization is not a possibility. When task assignments are made by the proposed heuristic procedure, cycle time

constraint is checked against the maximum task times. Thus, total work content of a worker is always equal to or smaller than the given cycle time. Table 4.4 gives the idle time for each model and each mated station with the final assignments made considering the maximum task times. The resulting solution employs 17 workers, while 9-mated stations are necessary.

Table 4.4 Idle Times for Heuristic Solution with Maximum Task Times

	Stations	1	2	3	4	5	6	7	8	9
M1	Right	5.1	1.5	4.7	4.0	16.6	5.9	14.0	10.4	
	Left	12.1	1.5	17.8	1.2	0.3	3.3	7.9	2.3	15.1
M2	Right	5.1	1.5	1.8	7.9	7.5	20.2	0.3	5.3	
	Left	1.7	1.5	9.4	1.2	0.3	3.3	0.4	2.3	8.5
M3	Right	0.4	1.5	9.6	4.7	0.3	6.3	0.3	5.3	
	Left	1.7	1.5	4.5	1.2	0.3	0.1	0.4	13.6	2.0
M4	Right	0.4	1.5	0.3	4.7	0.3	6.3	0.3	5.3	
	Left	1.7	1.5	4.5	1.2	0.3	0.1	0.4	1.1	2.0

Rather than overutilization, the opposite situation is observed when worked with the maximum task times, i.e., underutilization, where most stations experience high levels of idle times. Of course, all these idle times are not just the result of our choice of reducing the mixed-model problem into a single model problem. Model 4 has the task times closest to the maximum task times; however there are idle times for this model. These idle time figures may be the result of interference related idleness or inevitable idleness, because a worker's time cannot be fully used up since any combination of individual task times can be found such that a worker experiences absolutely no idle time.

The case study problem is solved with task times averaged according to Table 4.5. These ratios represent three different production scenarios. In the first scenario, models with more work content have more weights. In the second scenario all model weights are equal, while in the last scenario the models with less work content have more weights. From this point on, models with less work content which are model 1 and model 2, will be called *simple models*, while models with more work content, model 3 and model 4, will be called *difficult models*.

Table 4.5 Weights of Models

	Scenario 1	Scenario 2	Scenario 3
Model 1	15%	25%	35%
Model 2	15%	25%	35%
Model 3	35%	25%	15%
Model 4	35%	25%	15%

It should be noted that the models are assigned numbers in the order of their respective total work content. Thus, Model 1 has the minimum work content, while Model 4 has the maximum total work content. Model 4 has generally the highest task times among all models. However, this does not necessarily mean that maximum task time for each task is always with Model 4. Although this is the common pattern, there are cases where the maximum task time of a specific task occurs with Model 1 or another model.

The results of line balances with weighted averaged task times are given on Table 4.6. Note that the solution of our proposed heuristic procedure depends on the series of random numbers generated during the execution of the process. Thus, different results may be obtained with different random number series. For the rest of this part, we use the results with the highest occurrence for 10 different executions of the procedure.

Table 4.6 Results of Heuristic Procedure with the Averaged Task Times

	Max	Scenario 1	Scenario 2	Scenario 3
Number of Workers	17	16	15	15
Line Length	9	9	9	9

We already mentioned that line balancing obtained with the maximum task times always satisfy the cycle time feasibility. However, the line balances obtained by using the average task times only satisfy the cycle time feasibility with the average task times. It is highly likely that there are some stations on which the total processing time of the station exceeds the cycle time for some models. In order to determine these stations with cycle time violation, we need to calculate the actual finishing time of each task for every model. This process is not that much straightforward. First, actual starting of the task should be calculated. For each model, all predecessors of a task should be finished in order to start the task. Moreover, new instances of overutilization may arise which is not observed with the average task times. After calculating actual starting time of a task for each model, the actual task time for that specific model is added to the actual starting time to calculate the actual finishing time of the task for that model. Table 4.7 gives the initial assignments of tasks made by the proposed heuristic approach with the average task times of Scenario-1 and also the calculated actual finishing time of each task for every model for the first four mated stations. 'Heuristic Solution' part of the table shows the station and the side indices that tasks are assigned to. Finishing times are also presented. Note that no task's finishing time is higher than the cycle time. However, one should keep in mind that these finishing times are calculated with the averaged task times. The 'Actual Finishing Times' part of the table shows the actual finishing times if assignments are made according to the solution we obtain from the heuristic method. Note that the actual finishing time of each task changes from one model to another.

Shaded cells show where overutilization occurs. This means that the finishing time of some tasks exceed the cycle time.

There are many cases of overutilization for this assignment scheme obtained by the weighted average task times with weights of Scenario 1. For instance, right side worker at the mated station 3 seems to be perfectly loaded when the averaged task times are used. However, when actual finishing time of each task is calculated for each model, it is realized that the worker is overloaded for Models 1 and 4, while for Models 2 and 3, worker experiences underutilization. Nominal amount of work content of the worker seems to be 25.5 seconds with the averaged task times; however, actual work load ranges from 18.4 seconds to 28.0 seconds. The rest of the table for Scenario 1 and complete tables for Scenario 2 and Scenario 3 can be seen in Appendix C.

Table 4.7 Actual Finishing Times for Scenario 1

Heuristic Solution				Actual Finishing Times			
Mated Station	Side	Task	F(i)	Model I	Model II	Model III	Model IV
1	R	10	2.9	2.9	2.9	2.9	2.9
		11	5.4	2.9	5.9	5.9	5.9
		1	17.6	13.6	12.7	19.5	19.5
		7	18.7	17.2	16.2	19.5	19.5
		38	22	17.2	16.2	24.2	24.2
	L	13	4	4.0	4.0	4.0	4.0
		2	7.6	4.0	8.2	8.2	8.2
		12	24.1	20.4	24.7	24.7	24.7
		21	24.6	23.8	24.7	24.7	24.7
2	R	14	24	24.0	24.0	24.0	24.0
		29	25.1	25.1	25.1	25.1	25.1
	L	15	24	24.0	24.0	24.0	24.0
3	R	34	2.2	3.3	3.3	0.0	3.3
		8	10.6	14.6	3.3	9.6	12.9
		3	20.2	22.7	13.2	19.4	22.8
		43	23.7	26.2	16.6	22.9	26.2
		5	25.5	28.0	18.4	24.7	28.0
	L	16	4.3	4.3	4.3	4.3	4.3
		40	7.2	4.3	4.3	8.5	8.5
		41	8.9	4.3	4.3	10.9	10.9
		18	11.8	7.2	7.2	13.7	13.7
		17	16.5	7.2	12.8	19.3	19.3
		42	19.9	7.2	12.8	24.1	24.1
		48	24.5	27.0	17.5	23.7	27.1
4	R	50	2.6	2.6	2.6	2.6	2.6
		51	6.1	7.1	7.1	2.6	8.6
		52	8.3	9.2	9.2	4.7	10.7
		53	10	9.2	9.2	7.2	13.2
		55	12.2	11.5	11.5	9.4	15.4
		54	16.7	16.0	16.0	13.9	19.9
		70	19.6	18.9	18.9	16.9	22.9
		47	22.5	21.8	21.8	19.8	25.8
		49	24.7	24.0	24.0	21.9	27.9
	L	4	3.6	3.5	3.6	3.6	3.6
		6	6.6	3.5	3.6	7.9	7.9
		19	11.9	3.5	9.9	14.2	14.2
		56	14.3	5.9	12.3	16.6	16.6
		57	17.2	8.8	15.2	19.5	19.5
		58	20.3	11.9	18.3	22.6	22.6
		9	24	11.9	22.6	26.9	26.9

4.4.2 Resolving the Overutilization Cases

Given the actual finishing time of each task for every model, we see that there are many cases of overutilization. Table 4.8 presents the deviations from the given cycle time for each work station and each model. Positive deviations represent idle times and negative deviations, which are highlighted in the table, are the overutilization cases. Note that the assignments are made with the averaged task times with the weights of Scenario 1. In this scenario, difficult models have the majority in the total production amount.

The overutilization cases should be addressed such that the production process continues in a stable manner satisfying the predetermined cycle time of the line and thus achieving the given production target in the short and medium terms.

Table 4.8 Idle Times and Overutilization Cases for Scenario 1

	Mated Stations	1	2	3	4	5	6	7	8	9
M1	Right	8.3	0.4	-2.5	1.5		4.5	5.5	22.3	
	Left	1.7	1.5	-1.5	7.9	0.1	6.4	2.7	7.6	3.3
M2	Right	9.3	0.4	7.1	1.5		-3.1	0.3	12.3	
	Left	0.8	1.5	8.1	2.9	0.1	-1.1	2.7	0.9	3.3
M3	Right	1.3	0.4	0.8	3.6		-2.8	0.3	12.3	
	Left	0.8	1.5	1.8	-1.4	4.3	6.1	2.7	0.9	0.1
M4	Right	1.3	0.4	-2.5	-2.4		-4.3	0.3	12.3	
	Left	0.8	1.5	-1.6	-1.4	0.1	-2.3	2.7	0.9	0.1

As observed, with all models (M1, M2, M3, and M4) there are at least two overloaded workstations. Let us assume that we adopt a policy by which no reserved worker is assigned, when there appears no overloaded workstation with negative deviation more than 10% of the cycle time. Given this policy, we assign no reserved worker for the first model. For all other models, at least one worker

should be assigned. Assume that by one reserved worker only, all overutilization cases are resolved for all three models. Table 4.9 gives total workforce values for this scenario. Note that the base workforce for this scenario is determined as '16' with the proposed heuristic procedure.

Table 4.9 Workforce Figures for Scenario 1

Scenario 1	Model 1	Model 2	Model 3	Model 4
Base Workforce	16	16	16	16
Reserved Workforce	0	1	1	1
Total Workforce	16	17	17	17

This case study problem is adopted from a refrigeration production facility. Since mixed-model production takes place, there are two different products on the line at the same time. Thus, model 1 is always in mixed production with another model. As a result, although model 1 requires 16 workers, 17 workers should be kept on the line all the time, since all other models require that workforce level. As a result, we end up with the same workforce figure we obtain by line balancing with the maximum task time.

First scenario represented a production scenario where difficult models make up the majority of the total production. In Scenario 2 all models have an equal share. The resulting base workforce is 15 workers. Table 4.10 presents the deviation amounts from the given cycle time.

Note that the occurrence of overutilization increased for difficult models. This result is expected because as the weights of difficult models decrease, the gap between average task times and maximum task times increase. Since workers are assigned tasks with average task times, there will be more deviation between

theoretical total time of tasks assigned and the actual time it takes to complete the tasks assigned.

Table 4.10 Idle Times and Overutilization Cases for Scenario 2

	Mated Stations	1	2	3	4	5	6	7	8	9
M1	Right	-0.9	1.5	4.6	4.5	6.7	10.9			
	Left	-1.8	1.5	9.6	1.9	7.6	10.0	2.0	0.1	3.3
M2	Right	11.3	1.5	-1.6	4.5	-1.1	0.9			
	Left	1.5	1.5	-0.8	1.9	0.1	-2.4	2.0	0.1	3.3
M3	Right	-1.7	1.5	-1.6	2.2	-2.0	0.9			
	Left	0.4	1.5	-0.8	1.9	-3.7	-2.4	6.8	0.1	0.1
M4	Right	-5.0	1.5	-1.6	-3.8	-2.0	0.9			
	Left	0.4	1.5	-0.8	1.9	-3.7	-2.4	-5.7	0.1	0.1

The overutilization cases with less than 10% cap are ignored. Thus, no reserved workforce is necessary for models 1 and 2. For model 3, a reserved worker is assigned between the mated workstations 5 and 6 to the left side. Task 25 is performed by this worker for model 3. For model 4 the reserved worker is kept at the same place. It is observed that this worker can receive task 8 from mated station 1 right side, task 6 from mated station 4 right side, task 25 from mated station 5 left side, and tasks 16 and 40 from mated station 7 left side. These actions resolve all overutilization cases above 10% cap. The new reserve worker's total work content equals to 26.14 seconds. Ignoring this new occurrence of overutilization and other existing overutilization cases which are under 10% cap, we can maintain the production amounts of models 3 and 4 with the addition of one reserved worker. The resulting workforce figures are presented in Table 4.11. The maximum workforce is found to be 16 workers. Thus, any two-model mix can be produced with 16 workers. For this instance, using the averaged task times

lead to a solution better than the solution using the maximum task times. Assume models 3 and 4 are mixed. Note that both models have a base workforce figure of 15 workers. One additional worker is assigned to the same location on the line. Thus, total workforce is composed of 16 workers.

Table 4.11 Workforce Figures for Scenario 2

Scenario 2	Model 1	Model 2	Model 3	Model 4
Base Workforce	15	15	15	15
Reserved Workforce	0	0	1	1
Total Workforce	15	15	16	16

There may be cases where this total workforce exceeds 16 workers. This can occur if the reserved worker for models 3 and 4 cannot be located at the same location. In that case, two reserved workers should be kept on the line; one for model 3 and one for model 4, which makes the total workforce value as 17 workers. Assume all four models need a reserved worker and they should be located at different locations. Then, any two-model mix production needs a workforce composed of 17 workers which is equal to the workforce figure obtained with the maximum time balancing. This situation is exemplified in Figure 4.8. In this case, there are 6 mated stations and 12 workers. This is the base workforce and represented by white circles in the figure. Assume that one reserved worker, which is represented by the first black circle in the figure, is necessary for the first model and it should be placed between mated stations 2 and 3. For the other model, again one reserved worker is necessary. However, it should be placed between mated stations 5 and 6. Thus, 13 workers are required to produce each model. When they are in mixed production, however, the workers should be placed like in Figure 4.7. This mixed production requires not 13 workers, but 14 workers due to two extra workers placed on the line at different locations. Note that the first reserved worker is totally idle when the first model

comes to this workstation and the second reserved worker is totally idle when the second model comes to this second extra workstation.

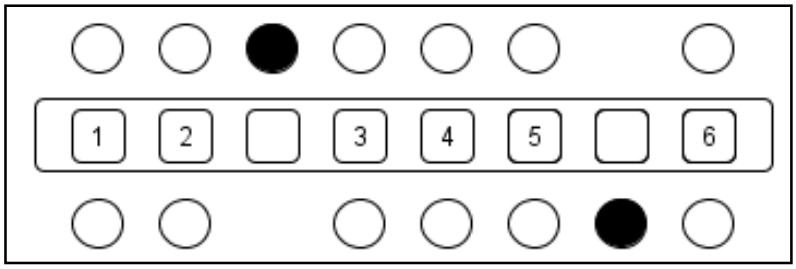

Figure 4.8 Reserve Workers at Different Locations

In Scenario 2 we are able to place the reserved worker at the same location for models 3 and 4. Thus, only one reserved worker is added to the base workforce. Now, the third production scenario is studied. In this scenario, simple models make up the majority of the total production amount. Thus, the averaged task times are the smallest among all scenarios. Table 4.12 presents the deviations experienced by each workstation on the line and for each model. The base workforce is composed of 15 workers for this alternative.

The deviation amounts from the cycle time increase again compared to the previous scenario. When the overutilization cases are studied, additional one reserved worker for model 2 and two reserved workers for models 3 and 4 resolve all overutilization cases. Since the overloaded stations mostly coincide for each model, the reserved workers can be placed in the same location on the line for all models for which reserved workers are required.

Table 4.12 Idle Times and Overutilization Cases for Scenario 3

	Mated Stations	1	2	3	4	5	6	7	8	9
M1	Right	5.0	1.5	4.0	7.5		13.9	7.5		
	Left	4.8	1.5	0.8	8.5	1.0	7.3	5.4	0.1	3.3
M2	Right	0.7	1.5	4.0	15.8		-3.9	0.3		
	Left	2.0	1.5	4.7	-3.3	1.0	-0.3	-1.3	0.1	3.3
M3	Right	-4.0	1.5	6.1	6.2		-4.9	0.3		
	Left	2.0	-3.3	0.4	-3.3	1.0	-4.0	3.5	0.1	0.1
M4	Right	-4.0	1.5	0.1	6.2		-4.9	0.3		
	Left	2.0	-3.3	-2.9	-3.3	1.0	-10.5	-2.5	0.1	0.1

Table 4.13 gives the total workforce values with the addition of reserved workers. In the production environment from which the case study problem is derived, any two models may be present on the line during mixed production. For instance, if model 1 is produced together with model 2, the total workforce should be 16. If model 1 is produced with model 2 there should be 17 workers on the line. Then, during production of model 1 and model 2 there will be 16 workers on the line and during production of model 3 and model 4 there will be 17 workers on the line. Depending on which models are mixed with each other, the resulting workforce figure can change from 16 to 17 given that reserve workers can coincide in each model of the mixed production.

Table 4.13 Workforce Figures for Scenario 3

Scenario 3	Model 1	Model 2	Model 3	Model 4
Base Workforce	15	15	15	15
Reserved Workforce	0	1	2	2
Total Workforce	15	16	17	17

With all these production scenarios, it is shown that that assembly line balancing can be made with the averaged task times and can actually lead to better solutions for some scenarios under certain conditions and assumptions. If production is mainly composed of models with more work content, then it is better to balance the line with the maximum task times. When the averaged task times are used in this case, the base workforce figure does not drop significantly, and when reserved workforce is added to the base workforce, the resulting workforce figure is not significantly different from the workforce obtained for line balancing with the maximum task times. In the opposite case, where difficult models are produced in very small amounts and simple models constitute the majority of the production, using the maximum task times leads to a solution where workers stay idle for long times, as most of the working hours are spent for the production of simple models where the workforce level is determined based on the higher task times. In this case, using average task times for line balancing significantly reduces the base workforce level. Even though a significant amount of reserved workforce should be added to the base workforce when difficult models are on the line, the resulting total labor cost is reduced.

These are the advantages of the line balancing with the average task times. However, it should be noted that the results are specific to this problem. We ignore many minor overutilization cases. Tasks are relocated from their originally assigned workstations to the newly created reserved workstations. Thus, some repetitions of the same task are performed at different locations. This is contrary to the basic objective of this study which is to assign each repetition of every task to the same workstation. When the same task is performed at different locations on the line for different models, the material distribution system has to distribute the materials to different location on the line. This increases the workload of material distribution system. If the task requires some kind of equipment, then the investment in this equipment is doubled when the same task is performed at two different locations on the line. We assume that when reserved workers are

free, i.e., they are not required on the line, they are considered to be working somewhere else in the total manufacturing system. However, in real life, the workforce cannot be re-sized that liberally. Mostly, an average production scenario is considered and the corresponding workforce to this scenario is hired and kept fixed. In the presence of all these real life conditions, it can be said that assembly line balancing with the average task times is not a straightforward and easy process. Nonetheless, it has been shown that it may improve the balancing of the line compared to the line balances obtained with the maximum task times.

CHAPTER 5

CONCLUSIONS

In this study, we focus on the problem of balancing a two-sided mixed-model assembly line. The objective is to minimize the number of workers to achieve the given cycle time. We first develop a mathematical model to solve the problems optimally and solve some example problems from the literature. We generate a problem set in which each problem has its own parameter settings. These problems are solved with the mathematical model so as to observe the performance of the mathematical model. One important observation obtained from this experimental study is that as problems get larger, i.e., the number of tasks in the problem increases, the computational time required reaching an optimal solution by the mathematical model exponentially increases.

The studies in the literature show that the low performance of the mathematical model on the computational speed lead people to search for faster methods at the expense of solution quality most of the time. These methods are meta-heuristic methods. There are an increasing number of studies on meta-heuristic procedures mostly inspired by the behavioral models of living things. These methods find good quality suboptimal solutions within reasonable computational times. We also apply a meta-heuristic in this study. Threshold Accepting is an easy to apply meta-heuristic procedure. It can be viewed as a deterministic version of the meta-heuristic -Simulated Annealing. We use this meta-heuristic procedure to solve the same problems as the ones solved by the mathematical model. This is the first application of the Threshold Accepting in this problem domain. It is observed that the procedure finds good solutions within very reasonable computational time.

The proposed procedure is also compared with existing meta-heuristic procedures presented in the literature by solving three different problems with different cycle time levels. It is again observed that our procedure is as good as the best existing meta-heuristic procedure.

To sum up, in this study, we present a mathematical model and a genuine meta-heuristic based procedure to solve the two-sided mixed-model assembly line balancing problem. The literature in this problem is still limited. For future research direction, more effort should be spent on developing faster exact solutions. As cited above, there is an increasing tendency towards avoiding the utilization of exact methods for large scale problems. This is not without its cost. Most of the time, it is sacrificed from the solution quality. Developing faster and exact methods can be a solution.

Moreover, the proposed mathematical model and the heuristic model both exclude some real life constraints on task assignments such as positive/negative zoning constraints. These models can be extended to cover more realistic cases of line balancing problem.

REFERENCES

AGRAWAL, P.K., The related activity concept in assembly line balancing, International Journal of Production Research, 23 (1985), 403–421.

BARTHOLDI, J.J., Balancing two-sided assembly lines: A case study, International Journal of Production Research, 31 (1993), 2447–2461.

BAYBARS, I., A survey of exact algorithms for the simple assembly line balancing problem, Management Science, 32 (1986a), 909–932.

BAYBARS, I., An efficient heuristic method for the simple assembly line balancing problem, International Journal of Production Research, 24 (1986b), 149–166.

BAYKASOGLU, A., DERELI, T., Two-sided assembly line balancing using an ant-colony based heuristic, International Journal of Advanced Manufacturing Technology, (2008), 36(5–6), 582–8.

BECKER, C., SCHOLL, A., A survey on problems and methods in generalized assembly line balancing, European Journal of Operational Research, 168 (2006), 694–715.

BOCTOR, F.F., A multiple-rule heuristic for assembly line balancing, Journal of the Operational Research Society, 46 (1995), 62–69.

BOWMAN, E.H., Assembly-line balancing by linear programming, Operational Research, 8 (May-June 1960), 385-389.

BOYSEN N., FLIEDNER, M., SCHOLL, A., A classification of assembly line balancing problems, European Journal of Operational Research, 183 (2007), 674–693.

BRYTON, B., Balancing of a continuous production line, Unpublished M.S. Thesis, Northwestern University, Evanston, ILL., (1954).

DUECK, G., SCHEUER, T., Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing, Journal of Computational Physics, 90 (1990), 161-175.

EREL, E., GOKCEN, H., Shortest-route formulation of mixed-model assembly line balancing problem, European Journal of Operational Research, 116 (1999), 194-204.

GOKCEN, H., EREL, E., A goal programming approach to mixed-model assembly line balancing problem, International Journal of Production Economics, 48 (1997), 177-185.

GUTHJAR, A.L., NEMHAUSER, G.L., An algorithm for the line balancing problem, Management Science, 11(1964), 308–315.

HELGESON, W.P., BIRNIE, D.P., Assembly line balancing using the Ranked Positional Weight technique," Journal of Industrial Engineering, 12, 6 (November-December 1961), 394-398.

HOFFMANN, T.R., Assembly line balancing with a precedence matrix, Management Science, 9, 4 (July 1963), 551-562.

HOFFMANN, T.R., EUREKA: A hybrid system for assembly line balancing, Management Science 38 (1992), 39–47.

JACKSON, J.R., A computing procedure for a line balancing problem, Management Science, 2, 3 (April 1956), 261-272.

JOHNSON, R.V., Optimally balancing large assembly lines with ‘‘FABLE’’, Management Science, 34 (1988), 240–253.

KILBRIDGE, M.D., WESTER, L., “The balance delay problem,” Management Science, 8 (1961), 69–84.

KIM Y.K., KIM Y., KIM Y.J., Two-sided assembly line balancing: a genetic algorithm approach, Production Planning and Control, (2000), 11(1), 44–53.

KIM Y.K., et al., A mathematical model and a genetic algorithm for two-sided assembly line balancing, Computers and Operations Research, (2007).

LAPIERRE, S.D., RUIZ, A.B., Balancing assembly lines: An industrial case study, The Journal of the Operational Research Society, 55, 6 (Jun., 2004), 589-597.

LEE, T.O., KIM, Y., KIM, Y.K., Two-sided assembly line balancing to maximize work relatedness and slackness, Computers and Industrial Engineering, 40 (2001) , 273–292.

MACASKILL, J.L.C., Production-line balances for mixed-model lines, Management Science, 19 (1972), 423–434.

OZCAN, U., Balancing of mixed-model two-sided assembly lines, Computers & Industrial Engineering, (2008), doi: 10.1016/j.cie.2008.11.012

OZCAN, U., TOKLU, B., Multiple-criteria decision-making in two-sided assembly line balancing: programming and a fuzzy goal programming models, Computers & Operations Research, (2008).

PASTOR, R., ANDRES, C., DURAN, A., PEREZ, M., Tabu search algorithms for an industrial multi-product and multi-objective assembly line balancing problem, with reduction of the task dispersion, Journal of the Operational Research Society, 53 (2002) , 1317–1323.

PATTERSON, J.H., ALBRACHT, J.J., Assembly line balancing: 0-1 programming with Fibonacci Search, Operational Research, 23 (1975), 166-174.

SALVESON, M.E., The assembly line balancing problem, Journal of Industrial Engineering, 6 (1955), 18-25.

SARKER, B.R., PAN, H., Designing a mixed-model, open-station assembly line using mixed-integer programming, Journal of Operational Research Society, 52 (2001), 545–558.

SCHOLL, A., BECKER, C., State-of-the-art exact and heuristic solution procedures for simple assembly line balancing, European Journal of Operations Research, 168 (2006), 694–715.

SIMARIA, A. S., VILARINHO, P. M., 2-ANTBAL: An ant colony optimization algorithm for balancing two-sided assembly lines, Computers & Industrial Engineering, (2007).

SIMARIA, A.S., VILARINHO P.M., A genetic algorithm based approach to the mixed-model assembly line balancing problem of type II, Computers & Industrial Engineering, 47 (2004), 391–407.

TALBOT, F.B., PATTERSON, J.H., An integer programming algorithm with network cuts for solving the assembly line balancing problem, Management Science, 30, 1 (January 1984), 85-99.

THOMOPOULOS, N.T., Line balancing-sequencing for mixed-model assembly, Management Science, 14 (1967), 59–75.

THOMOPOULOS, N.T., Mixed-model line balancing with smoothed station assignments, Management Science, 16 (1970), 593–603.

TIAN, P., MA, J., ZHANG, D-M., Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: An investigation of generation mechanism, European Journal of Operational Research, 118 (1999), 81-94.

XIAOFENG, H., ERFEI W., YE, J., A station-oriented enumerative algorithm for two-sided assembly line balancing, European Journal of Operational Research, 186 (2008), 435–440.

APPENDIX A

TEST BED PROBLEMS DATA

Table A.1 Data of 24-Task Test Bed Problem

Task No.	Side	Task Time	Immediate Predecessors
1	L	3	-
2	L	7	-
3	R	7	-
4	R	5	-
5	L	4	2
6	E	3	2,3
7	R	4	3
8	E	3	5
9	E	6	6
10	E	4	7
11	L	4	1
12	L	3	8,9
13	E	3	9
14	R	9	9,10
15	R	5	4
16	L	9	11
17	E	2	12
18	E	7	13
19	E	9	13,14
20	R	9	15
21	L	8	16,17
22	E	8	18
23	R	9	19,20
24	E	9	20

Table A.2 Data of 65-Task Test Bed Problem

Task No.	Side	Task Time	Immediate Successors
1	E	49	3
2	E	49	3
3	E	71	4-23
4	E	26	5-6-7-9-11-12-25-26-27-41-45-49
5	E	42	14
6	E	30	14
7	R	167	8
8	R	91	14
9	L	52	10
10	L	153	14
11	E	68	14
12	E	52	14
13	E	135	14
14	E	54	15-18-20-22
15	E	57	16
16	L	151	17
17	L	39	31
18	R	194	19
19	R	35	21
20	E	119	21
21	E	34	31
22	E	38	31
23	E	104	24
24	R	84	31
25	L	113	31
26	L	72	31
27	L	62	28
28	R	272	50
29	L	89	50
30	L	49	50
31	E	11	32-36
32	E	45	33
33	E	54	34
34	E	106	35
35	R	132	50
36	E	52	37
37	E	157	38
38	E	109	39-40

Table A.2 (continued)

Task No.	Side	Task Time	Immediate Successors
39	L	32	50
40	R	32	50
41	E	52	42
42	E	193	43
43	E	34	62
44	R	34	46
45	L	97	46
46	E	37	47
47	L	25	48
48	L	89	50
49	E	27	50
50	E	50	65
51	R	46	65
52	E	46	65
53	L	55	65
54	E	118	65
55	R	47	65
56	E	164	57
57	E	113	65
58	L	69	65
59	R	30	65
60	E	25	65
61	R	106	65
62	E	23	63
63	L	118	64
64	L	155	65
65	E	65	-

Table A.3 Data of 148-Task Test Bed Problem

Task	Side	Task Time	Immediate Successors
1	E	16	5,6,7,8
2	E	30	3
3	E	7	4,5,6,7
4	E	47	8
5	E	29	14
6	E	8	9
7	E	39	14
8	E	37	10
9	E	32	14
10	E	29	14
11	E	17	12
12	E	11	13
13	E	32	-
14	E	15	15,16
15	L	53	17
16	R	53	17
17	E	8	18,19
18	L	24	20
19	R	24	20
20	E	8	21,22,23,24
21	R	7	25,26,27,28
22	L	8	25,26,27,28
23	L	14	25,26,27,28
24	R	13	25,26,27,28
25	R	10	29
26	R	25	29
27	L	11	29
28	L	25	29
29	E	11	31
30	R	29	-
31	E	25	36
32	L	10	34
33	R	14	35
34	L	41	36
35	R	42	36
36	R	47	37
37	R	7	38,45

Table A.3 (continued)

Task	Side	Task Time	Immediate Successors
38	R	80	39
39	R	7	40
40	R	41	41,48,54
41	R	47	-
42	L	16	43
43	L	32	44
44	L	66	-
45	L	80	46
46	L	7	47
55	R	7	133
56	E	28	73
57	L	12	82
58	L	52	86,88
59	E	14	75,89
60	E	3	-
61	E	3	62
62	E	8	63
63	E	16	67
64	R	33	65,71,72
65	E	8	66,99
66	E	18	67
67	E	10	68
68	E	14	95,98
69	R	28	79
70	R	11	71
71	R	118	-
72	R	25	134
73	E	40	86,88,89,90,96
74	E	40	75
75	E	101	90,97
76	E	5	77
77	E	28	78
78	E	8	82
79	E	111	80
80	E	7	81
81	E	26	82
82	E	10	83,84

Table A.3 (continued)

Task	Side	Task Time	Immediate Successors
83	E	21	-
84	E	26	106
85	E	20	-
86	E	21	87
87	E	47	-
88	E	23	-
95	E	20	101
96	E	31	104
97	E	19	-
98	E	34	101
99	E	51	100
100	E	39	101
101	E	30	102,103
102	E	26	127
103	E	13	127
104	E	45	-
105	E	58	119
106	E	28	107
107	E	8	108
108	E	43	109
109	E	40	110
110	E	34	-
111	E	23	112
112	L	162	113
113	L	11	114,116,120,123,128
114	E	19	115
115	E	14	125
116	E	31	117
117	E	32	118
118	E	26	126
119	E	55	-
120	E	31	121
121	E	32	122
122	E	26	126
123	E	19	124
124	E	14	125
125	E	19	-
126	E	48	-
127	E	55	-
128	L	8	129
129	L	11	130
130	L	27	131,137

Table A.3 (continued)

Task	Side	Task Time	Immediate Successors
131	L	18	-
132	E	36	135
133	L	23	135
134	R	20	135
135	E	46	136
136	E	64	-
137	L	22	-
138	E	15	139
139	E	34	140
140	E	22	-
141	L	151	142
142	R	148	143,146,147,148
143	L	64	-
144	L	170	145
145	R	137	147,148
146	R	64	-
147	L	78	-
148	R	78	-

Table A.4 Data of 205-Task Test Bed Problem

Task No.	Side	Task Time	Immediate Successors
1	E	692	36
2	E	42	3,4
3	R	261	5
4	L	261	5
5	E	157	7,13
6	E	90	36
7	R	54	8
8	R	67	9
9	R	30	10
10	R	106	11
11	R	32	12
12	L	62	36
13	L	54	14
14	L	67	15
15	L	30	16
16	L	106	17
17	L	32	18
18	L	62	36
19	E	56	36
20	E	67	22
21	E	86	22
22	E	37	23
23	E	41	24,34
24	E	72	26,27,28
25	R	86	28
26	L	16	35
27	R	51	35
28	R	66	29
29	R	41	30,33
30	R	72	31,32
31	R	51	35
32	R	16	35
33	R	15	35
34	L	15	35
35	E	85	36
36	E	59	37,40,41,42,62,69,72,75,83,110,111,112
37	L	23	38
38	L	13	39

Table A.4 (continued)

Task No.	Side	Task Time	Immediate Successors
39	L	19	45
40	E	108	43,54
41	E	214	92
42	E	80	43,54
43	L	37	44
44	L	84	45
45	L	18	46,48,51,53
46	L	12	47
47	L	29	92
48	L	37	49
49	L	13	50
50	L	70	92
51	L	217	52
52	L	72	92
53	L	85	92
54	R	43	55
55	R	97	56,59,61
56	R	37	57
57	R	13	58
58	R	35	92
59	R	217	60
60	R	72	92
61	R	85	92
62	E	43	63
63	E	37	64
64	E	37	65,68
65	E	103	66
66	E	140	67
67	E	49	80
68	E	35	80
69	E	51	70
70	E	88	71
71	E	53	73
72	E	144	73
73	E	337	74
74	E	107	76
75	E	371	92
76	E	97	77,78,79

Table A.4 (continued)

Task No.	Side	Task Time	Immediate Successors
77	E	166	80,82
78	L	92	80
79	R	92	80
80	E	106	81
81	E	49	84
82	E	92	92
83	E	371	92
84	E	87	85
85	E	162	86,88,90
86	E	96	87
87	E	79	92
88	E	96	89
89	E	42	91
90	R	88	92
91	R	90	93,94,95,96,97,98,99
92	R	97	135
93	R	270	135
94	E	452	113
95	R	48	113
96	E	338	100
97	E	34	100
98	E	65	100
99	E	50	101,103,105,109,130,131,134
100	E	112	102
101	E	48	113
102	E	117	104
103	E	50	113
104	R	68	106,107
105	L	232	108
106	L	122	108
107	E	151	113
108	L	31	113
109	E	97	113
110	R	308	113
111	L	116	113
112	R	312	114,115,116,117,118,119,120,121,122,123,124,161,162,163,169,171,174

Table A.4 (continued)

Task No.	Side	Task Time	Immediate Successors
113	E	34	203,204,205
114	L	128	160
115	E	54	160
116	R	175	160
117	E	55	160
118	E	306	126
119	E	59	126
120	E	59	126
121	E	66	126
122	E	66	126
123	E	23	126
124	E	244	125
125	E	54	126
126	R	294	127,128,129
127	E	84	135
128	E	61	135
129	E	57	135
130	R	38	136
131	E	944	132
132	R	511	133
133	R	625	189
134	R	445	189
135	L	68	136,137,138,139,140,141,142,144,145,148,149,150,151,153,158
136	L	53	189
137	E	49	160
138	E	92	160
139	E	236	160
140	L	116	143
141	L	265	143
142	L	149	143
143	L	74	160
144	E	332	160
145	E	324	146
146	L	104	160
147	L	51	160
148	R	58	160

Table A.4 (continued)

Task No.	Side	Task Time	Immediate Successors
149	R	67	160
150	R	49	160
151	E	107	160
152	L	38	160
153	L	27	154
154	E	68	155
155	E	207	156
156	E	202	157
157	E	83	189
158	R	35	159
159	R	58	189
160	E	42	164,170,178,179,184
161	R	68	167
162	R	68	165
163	R	68	164
164	R	103	165
165	R	103	166
166	R	103	167
167	R	103	168
168	R	103	177
169	L	68	170
170	L	103	172
171	L	68	172
172	L	103	175
173	L	103	175
174	L	68	175
175	L	103	176
176	L	103	177
177	E	10	185,186,187,188,194,195
178	E	187	180
179	L	134	180
180	L	89	181,183
181	L	58	182
182	L	49	-
183	L	134	-
184	L	53	-
185	E	334	189
186	R	24	189
187	R	76	189

TABLE A.4 (continued)

Task No.	Side	Task Time	Immediate Successors
188	L	76	189
189	E	192	190,191,193
190	E	98	-
191	R	258	192
192	E	165	-
193	R	38	-
194	E	115	197
195	L	83	196
196	R	56	197
197	R	29	198,199,201
198	R	303	-
199	R	18	200
200	R	29	-
201	L	154	202
202	L	90	-
203	L	93	-
204	E	94	-
205	E	165	-

APPENDIX B

DETAILS OF SOLUTIONS OBTAINED WITH MATHEMATICAL MODEL

Table B.1 Solution Details of 24-Task Test Bed Problem

Task No	Mated Station No.	Left	Right	Starting Time
1	2	1		6
2	1	1		0
3	1		1	0
4	2		1	0
5	4	1		0
6	1	1		12
7	1		1	7
8	5		1	12
9	2	1		0
10	1		1	11
11	3	1		0
12	6	1		0
13	4		1	0
14	2		1	6
15	3		1	0
16	3	1		6
17	6	1		3
18	5	1		0
19	4	1		6
20	3		1	6
21	6	1		7
22	5	1		7
23	5		1	0
24	4		1	6

Table B.2 Solution Details of 65-Task Test Bed Problem

Task No.	Mated Station No.	Left	Right	Starting Time
1	1		1	49
2	1		1	0
3	1		1	98
4	1	1		169
5	1	1		317
6	1	1		287
7	1		1	333
8	2		1	0
9	2	1		0
10	2	1		168
11	2		1	91
12	1	1		359
13	2		1	192
14	2		1	327
15	2	1		443
16	3	1		0
17	3	1		151
18	3		1	0
19	3		1	266
20	2		1	381
21	3		1	302
22	2	1		405
23	2	1		52
24	2	1		321
25	3	1		194
26	3		1	194
27	1		1	219
28	5		1	71
29	1	1		411
30	3	1		451
31	3		1	336
32	3	1		406
33	4		1	52
34	5	1		124
35	5		1	343
36	4		1	0
37	4		1	106
38	4		1	359

Table B.2 (continued)

Task No.	Mated Station No.	Left	Right	Starting Time
39	4	1		468
40	6		1	164
41	1		1	281
42	4	1		0
43	4		1	263
44	5		1	0
45	3	1		309
46	5		1	34
47	5	1		231
48	5	1		256
49	6		1	196
50	6		1	226
51	4		1	313
52	6		1	276
53	5	1		69
54	4	1		193
55	3		1	453
56	6		1	0
57	6		1	322
58	5	1		0
59	4		1	470
60	5		1	475
61	3		1	347
62	4	1		327
63	4	1		350
64	5	1		345
65	6		1	435

Table B.3 Solution Details of Case Study Problem

Task No.	Mated Station No.	Left	Right	S.T. Model I	S.T. Model II	S.T. Model III	S.T. Model IV
1	2	1		3.95	3.95	3.95	3.95
2	1	1		21.22	21.22	21.22	21.22
3	1	1		0.00	0.00	0.00	0.00
4	2	1		22.02	21.91	21.91	17.59
5	1	1		11.49	9.84	9.84	9.84
6	2	1		25.50	25.50	25.50	21.18
7	6		1	8.91	10.57	10.57	14.33
8	10	1		14.24	25.50	15.90	15.90
9	1	1		18.73	16.89	16.89	16.89
10	2		1	22.57	22.57	22.57	22.57
11	4		1	3.48	3.48	3.48	3.48
12	2		1	0.00	0.00	0.00	0.00
13	2	1		0.00	0.00	0.00	0.00
14	3		1	0.00	0.00	0.00	0.00
15	3	1		0.00	0.00	0.00	0.00
16	4	1		10.75	10.75	10.75	10.75
17	4	1		4.30	4.30	4.30	4.30
18	5	1		22.65	22.65	22.65	22.65
19	4	1		15.09	15.09	15.09	15.09
20	5	1		0.00	0.00	0.00	0.00
21	1	1		8.11	9.84	9.84	9.84
22	6		1	2.15	2.15	2.15	2.15
23	7		1	18.22	21.25	21.25	21.25
24	7		1	0.00	0.00	0.00	0.00
25	6		1	8.91	10.57	10.57	10.57
26	10		1	7.96	7.96	7.96	7.96
27	7		1	6.13	9.81	9.81	9.81
28	10		1	19.74	19.74	19.74	19.74
29	3		1	24.41	24.41	24.41	24.41
30	10	1		11.54	20.23	13.20	10.80
31	10		1	22.93	22.93	22.93	22.93
32	9	1		3.30	3.30	2.70	0.06
33	7	1		2.70	2.70	2.70	2.70
34	4		1	19.25	19.26	19.26	22.57
35	4	1		21.38	21.38	21.38	21.38
36	9	1		0.00	0.00	0.00	0.00
37	6	1		0.00	0.00	0.00	0.00

Table B.3 (continued)

Task No.	Mated Station No.	Left	Right	S.T. Model I	S.T. Model II	S.T. Model III	S.T. Model IV
38	4		1	11.82	11.83	11.83	14.92
39	9	1		25.50	25.50	25.50	22.26
40	4	1		25.50	25.50	25.50	21.38
41	10	1		14.24	25.50	15.90	13.50
42	6	1		4.81	4.81	4.81	0.00
43	4		1	0.00	0.00	0.00	0.00
44	6		1	18.51	18.51	18.51	18.51
45	6		1	14.83	14.83	14.83	14.83
46	7		1	0.00	3.68	3.68	3.68
47	4		1	16.32	16.33	16.33	19.64
48	4	1		0.00	0.00	0.00	0.00
49	10		1	15.34	15.34	15.34	15.34
50	4		1	9.25	9.26	9.26	12.35
51	4		1	11.82	11.83	11.83	19.64
52	6		1	0.00	0.00	0.00	0.00
53	7	1		0.00	0.00	0.00	0.00
54	10		1	0.00	0.00	0.00	0.00
55	10		1	17.50	17.50	17.50	17.50
56	10	1		2.24	2.24	2.24	2.24
57	10	1		4.64	4.64	4.64	4.64
58	6	1		22.41	22.41	22.41	22.41
59	6	1		4.81	4.81	4.81	4.81
60	6	1		11.36	11.36	11.36	11.17
61	10	1		0.00	0.00	0.00	0.00
62	6	1		14.65	14.65	14.65	14.46
63	7		1	21.25	21.25	21.25	21.25
64	6		1	22.19	22.19	22.19	22.19
65	6		1	23.89	23.89	23.89	23.89
66	7		1	23.57	23.57	23.57	23.57
67	8	1		0.00	0.00	0.00	0.00
68	8	1		3.46	3.46	3.46	3.46
69	8	1		10.99	10.99	10.99	10.99
70	4		1	22.57	22.57	22.57	22.57
71	10		1	4.50	4.50	4.50	4.50
72	8	1		17.70	17.70	17.70	17.70
73	8	1		23.10	23.10	23.10	23.10
74	10	1		8.99	17.68	10.65	8.25

Table B.4 Results of Experimental Study on Mathematical Model

Problem Set	SPF	OS	CT	Number of Workers	Solution Time(second)	Gap Percentage
20 Task	0.3	0.2	250	6	85.79	0.00%
			300	5	85.31	0.00%
		0.4	250	7	26.34	0.00%
			300	6	20.65	0.00%
		0.6	250	7	14.39	0.00%
			300	6	16.04	0.00%
		0.8	250	8	13.82	0.00%
			300	6	12.45	0.00%
	0.5	0.2	250	6	31.85	0.00%
			300	5	14.48	0.00%
		0.4	250	7	36.40	0.00%
			300	5	31.65	0.00%
		0.6	250	7	16.70	0.00%
			300	6	17.37	0.00%
		0.8	250	8	15.53	0.00%
			300	6	13.53	0.00%
	0.7	0.2	250	6	28.75	0.00%
			300	5	17.4	0.00%
		0.4	250	6	358.04	0.00%
			300	5	31.37	0.00%
		0.6	250	7	46.01	0.00%
			300	6	17.53	0.00%
		0.8	250	7	16.43	0.00%
			300	6	13.59	0.00%

Table B.4 (continued)

Problem Set	SPF	OS	CT	Number of Workers	Solution Time(second)	Gap Percentage
30 Task	0.3	0.2	250	9	10800	13.07%
			300	8	10800	14.67%
		0.4	250	9	97.34	0.00%
			300	8	932.12	0.00%
		0.6	250	10	20.95	0.00%
			300	8	34.9	0.00%
		0.8	250	12	17.04	0.00%
			300	9	13.26	0.00%
	0.5	0.2	250	9	10800	14.76%
			300	7	1322.84	0.00%
		0.4	250	9	342.09	0.00%
			300	8	8331.31	0.00%
		0.6	250	10	119.03	0.00%
			300	8	405.17	0.00%
		0.8	250	12	19.9	0.00%
			300	9	13.09	0.00%
	0.7	0.2	250	9	10800	14.76%
			300	7	513.92	0.00%
		0.4	250	9	10800	14.40%
			300	7	129.14	0.00%
		0.6	250	10	2907.64	0.00%
			300	8	2444.43	0.00%
		0.8	250	11	65.51	0.00%
			300	9	62.51	0.00%

Table B.4 (continued)

Problem Set	SPF	OS	CT	Number of Workers	Solution Time(second)	Gap Percentage
40 Task	0.3	0.2	250	12	10800	12.47%
			300	10	10800	19.47%
		0.4	250	12	10800	19.14%
			300	10	10800	17.54%
		0.6	250	13	2496.5	0.00%
			300	11	19.93	0.00%
		0.8	250	15	16.46	0.00%
			300	12	18.6	0.00%
	0.5	0.2	250	12	10800	19.47%
			300	9	2687.21	0.00%
		0.4	250	12	10800	19.47%
			300	10	10800	19.23%
		0.6	250	14	10800	30.96%
			300	11	97.56	0.00%
		0.8	250	14	455.56	0.00%
			300	12	19.32	0.00%
	0.7	0.2	250	12	10800	19.47%
			300	9	157.07	0.00%
		0.4	250	11	10800	12.15%
			300	9	541.42	0.00%
		0.6	250	12	10800	19.47%
			300	10	217.07	0.00%
		0.8	250	13	100.51	0.00%
			300	11	199.68	0.00%

Table B.4 (continued)

Problem Set	SPF	OS	CT	Number of Workers	Solution Time(second)	Gap Percentage
50 Task	0.3	0.2	250	16	10800	26.25%
			300	13	10800	24.35%
		0.4	250	16	10800	26.25%
			300	14	10800	29.76%
		0.6	250	16	10800	26.25%
			300	15	10800	34.44%
		0.8	250	17	10800	10.00%
			300	14	10800	15.55%
	0.5	0.2	250	16	10800	26.25%
			300	12	10800	18.05%
		0.4	250	16	10800	26.25%
			300	13	10800	24.35%
		0.6	250	18	10800	34.44%
			300	13	10800	24.35%
		0.8	250	17	10800	27.88%
			300	13	10800	7.69%
	0.7	0.2	250	No Solution	10800	-
			300	No solution	10800	-
		0.4	250	15	10800	21.23%
			300	13	10800	24.36%
		0.6	250	16	10800	26.25%
			300	12	10800	18.06%
		0.8	250	16	10800	25.22%
			300	13	10800	22.21%

APPENDIX C

DETAILS OF DECODING ALGORITHM AND SOLUTIONS OBTAINED WITH HEURISTIC PROCEDURE

This part includes the detailed algorithm of decoding procedure. Before procedure is presented, necessary definitions will be introduced.

Indices:

i, j, k: task indeces

m: model index

Parameters:

n: number of tasks

C: target cycle time

$side(i) =$ preferred side of $task\ i$

$t_{im} =$ processing time of $task\ i$ for $model\ m$

$p(i,j) = \begin{cases} 1, \text{if } task\ i \text{ precedes } task\ j \\ 0, \text{otherwise} \end{cases}$

Sets:

AT: set of available tasks for the next assignment

U: set of all tasks

Variables:

$X(i)$: binary variable; 1 if $task\ i$ is assigned to a station, 0 otherwise

$NUP(i)$: current number of unassigned immediate predecessors of $task\ i$

$IS(i)$: the station number $task\ i$ is assigned

$T(i)$: the side $task\ i$ is assigned. 1 if right, 2 if left

$FT(i)$: finishing time of $task\ i$

$PST(i)$: possible starting time of $task\ i$

CTR: current time of right side station under consideration for assignment

CTL: current time of left side station under consideration for assignment

LL: current line length; equals number of mated station already opened.

v: binary variable

Initialization

$CTR = 0$

$CTL = 0$

$LL = 1$

$NUP(i) =$ number of immediate predecessors of $task\ i\ \forall i \in U$

$X(i) = 0\ \ \forall i \in U$

$PST(i) = 0\ \ \forall i \in U$

$FT(i) = 0\ \ \forall i \in U$

$IS(i) = 0\ \ \forall i \in U$

$T(i) = 0\ \ \forall i \in U$

$v\ =\ 0$

Step1

If $CTR <=\ CTL$ then Go to Step 2

Else Go to Step 3

Step2

I. If $n = 0$, STOP!

II. $AT = \{i \in U : side(i) = "R" \text{ or } "E",\ X(i) = 0,\ NUP(i) = 0\}$

III. Calculate $PST(i)\ for\ \forall i\ \in AT$.

$PST(i)\ =\ max\{\ CTR, F(j)\ \}\quad \forall\ j :\ p(j,i) = 1\ and\ IS(j) = LL$

IV. Eliminate elements that cannot fit in the remaining time.

A task is eliminated if $PST(i)\ +\ max_m\{t_{im}\}\ >\ C$

If no task remains in reduced set:

If $m = 0$ then

Set $m = 1$

Go to Step3

Else if $m = 1$

$LL\ =\ LL\ +\ 1$

$CTR\ =\ 0$

$CTL\ =\ 0$

Go to Step 1

Endif

V. Pick task with minimum PST value having minimum priority

VI. If task has a synchronous task, go to step 4.

VII. Assign task to the current station

Set:

$X(i)\ =\ 1$

$IS(i)\ =\ LL$

$T(i)\ =\ 1$

$CTR\ =\ PST(i)\ +\ max_m\{t_{im}\}$

$F(i)\ =\ PST(i)\ +\ max_m\{t_{im}\}$

$v = 0$

$n\ =\ n - 1$

VIII. $NUP(j) = NUP(j) - 1,\ for\ \forall j \in U\ and\ p(i,j) = 1$

IX. Go to Step 1.

Step3

I. If $n = 0$, STOP!

II. $AT = \{i \in U : side(i) = "L" \text{ or } "E",\ X(i) = 0,\ NUP(i) = 0\}$

III. Calculate $PST(i)\ for \forall i \in AT$.

$PST(i) = max\{CTL, F(j)\} \quad \vee j : p(j,i) = 1\ and\ IS(j) = LL$

IV. Eliminate elements that cannot fit in the remaining time.

A task is eliminated if $PST(i) + max_m\{t_{im}\} > C$

If no task remains in reduced set:

If $m = 0$ then

Set $m = 1$

Go to Step2

Else if $m = 1$

$LL = LL + 1$

$CTR = 0$

$CTL = 0$

Go to Step 1

Endif

V. Pick task with minimum PST value having minimum priority

VI. If task has a synchronous task, go to step 5.

VII. Assign task to the current station

Set:

$X(i) = 1$

$IS(i) = LL$

$T(i) = 2$

$CTL = PST(i) + max_m\{t_{im}\}$

$F(i) = PST(i) + max_m\{t_{im}\}$

$v = 0$

$n = n - 1$

VIII. $NUP(j) = NUP(j) - 1,\ for \forall j \in U\ and\ p(i,j) = 1$

IX. Go to Step 1.

Step 4

I. Right − side $task\ i$ has synchronous left − side $task\ k$.

Set

$PST(i) = max\{CTR, CTL\}$

$PST(k) = max\{CTR, CTL\}$

$X(i) = 1$

$X(k) = 1$

$T(i) = 1$

$T(k) = 2$

$m = 0$

$n = n - 2$

If $PST(i) + max_m\{t_{im}\} <= C$ then

$CTR = PST(i) + max_m\{t_{im}\}$

$CTL = PST(k) + max_m\{t_{km}\}$

Else

$LL = LL + 1$

$CTR = max_m\{t_{im}\}$

$CTL = max_m\{t_{km}\}$

Endif

Set

$IS(i) = LL$

$IS(k) = LL$

$F(i) = CTR$

$F(k) = CTL$

II. $NUP(j) = NUP(j) - 1\ ,\ \forall j \in U\ and\ (\ p(i,j) = 1\ and/or\ p(k,j) = 1)$

III. Go to Step 1

Step5

I. Left − side $task\ i$ has synchronous right − side $task\ k$.

Set

$PST(i) = max\{CTR, CTL\}$

$PST(k) = max\{CTR, CTL\}$

$X(i) = 1$

$X(k) = 1$

$T(i) = 2$

$T(k) = 1$

$m = 0$

$n = n - 2$

If $PST(i) + max_m\{t_{im}\} <= C$ then

$CTL = PST(i) + max_m\{t_{im}\}$

$CTR = PST(k) + max_m\{t_{km}\}$

Else

$LL = LL + 1$

$CTL = max_m\{t_{im}\}$

$CTR = max_m\{t_{km}\}$

Endif

Set

$IS(i) = LL$

$IS(k) = LL$

$F(i) = CTL$

$F(k) = CTR$

II. $NUP(j) = NUP(j) - 1\ ,\ \forall j \in U\ and\ (\ p(i,j) = 1\ and/or\ p(k,j) = 1)$

III. Go to Step 1

Table C.1 Heuristic Solution for 24-Task Problem

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
3	7	R		-	1	1	7
7	4	R	3	-	1	1	11
10	4	E	7	-	1	1	15
2	7	L		-	1	2	7
6	3	E	3 2	-	1	2	10
5	4	L	2	-	1	2	14
9	6	E	6	-	2	1	6
14	9	R	10 9	-	2	1	15
8	3	E	5	-	2	2	3
1	3	L		-	2	2	6
13	3	E	9	-	2	2	9
11	4	L	1	-	2	2	13
18	7	E	13	-	3	1	7
22	8	E	18	-	3	1	15
16	9	L	11	-	3	2	9
12	3	L	9 8	-	3	2	12
17	2	E	12	-	3	2	14
19	9	E	14 13	-	4	1	9
4	5	R		-	4	1	14
21	8	L	17 16	-	4	2	8
15	5	R	4	-	5	1	5
20	9	R	15	-	5	1	14
23	9	R	20 19	-	6	1	9
24	9	E	20	-	6	2	9

Table C.2 Heuristic Solution for 65-Task Problem

TASK	TASK TIME	TASK SIDE	TASK IP	IS(i)	T(i)	F(i)
44	34	R		1	1	34
13	135	E		1	1	169
3	71	E	2 1	1	1	240
23	104	E	3	1	1	344
5	42	E	4	1	1	386
11	68	E	4	1	1	454
1	49	E		1	2	49
2	49	E		1	2	98
30	49	L		1	2	147
29	89	L		1	2	236
4	26	E	3	1	2	266
41	52	E	4	1	2	318
12	52	E	4	1	2	370
27	62	L	4	1	2	432
6	30	E	4	1	2	462
28	272	R	27	2	1	272
7	167	R	4	2	1	439
42	193	E	41	2	2	193
43	34	E	42	2	2	227
9	52	L	4	2	2	279
10	153	L	9	2	2	432
8	91	R	7	3	1	91
14	54	E	13 12 11 10 8 6 5	3	1	145
18	194	R	14	3	1	339
15	57	E	14	3	1	396
24	84	R	23	3	1	480
25	113	L	4	3	2	113
45	97	L	4	3	2	210
46	37	E	45 44	3	2	247
20	119	E	14	3	2	366
47	25	L	46	3	2	391
48	89	L	47	3	2	480
22	38	E	14	4	1	38
19	35	R	18 17	4	1	252
21	34	E	20 19	4	1	286
31	11	E	26 25 24 22 21	4	1	300
55	47	R	31	4	1	347

Table C.2 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	IS(i)	T(i)	F(i)
62	23	E	43 31	4	1	370
59	30	R	31	4	1	400
32	45	E	31	4	1	445
52	46	E	31	4	1	491
49	27	E	48 4	4	2	27
16	151	L	15	4	2	178
17	39	L	16	4	2	217
26	72	L	4	4	2	289
54	118	E	31	4	2	418
58	69	L	31	4	2	487
61	106	R	31	5	1	106
51	46	R	31	5	1	152
60	25	E	31	5	1	177
56	164	E	31	5	1	341
33	54	E	32	5	1	395
53	55	L	31	5	2	55
63	118	L	62	5	2	173
64	155	L	63	5	2	328
36	52	E	31	5	2	380
57	113	E	56	5	2	493
34	106	E	33	6	1	106
35	132	R	34	6	1	238
40	32	R	38	6	1	298
65	65	E	64 61 60 59 58 57 55 54 53 52 51 50	6	1	413
37	157	E	36	6	2	157
38	109	E	37	6	2	266
39	32	L	38	6	2	298
50	50	E	49 40 39 35 30 29 28	6	2	348

Table C.3 Heuristic Solution for 148-Task Problem

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
132	36	E			1	1	36
74	40	E			1	1	76
2	30	E			1	1	106
56	28	E			1	1	134
11	17	E			1	1	151
1	16	E			1	1	167
91	115	E			1	1	282
105	58	E	91		1	1	340
60	3	E			1	1	343
141	151	L			1	2	151
52	11	L			1	2	162
93	26	L			1	2	188
32	10	L			1	2	198
57	12	L			1	2	210
3	7	E	2		1	2	217
6	8	E	3 1		1	2	225
9	32	E	6		1	2	257
7	39	E	3 1		1	2	296
4	47	E	3		1	2	343
61	3	E			1	2	346
33	14	R			2	1	14
8	37	E	4 1		2	1	51
70	11	R			2	1	62
35	42	R	33		2	1	104
142	148	R	141		2	1	252
50	33	E			2	1	285
59	14	E			2	1	299
138	15	E			2	1	314
30	29	R			2	1	343
62	8	E	61		2	2	8
63	16	E	62		2	2	24
119	55	E	105		2	2	79
42	16	L			2	2	95
58	52	L			2	2	147
5	29	E	3 1		2	2	176
92	35	E			2	2	211
94	46	E			2	2	257
34	41	L	32		2	2	298

Table C.3 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
51	34	L	50		2	2	332
12	11	E	11		2	2	343
69	28	R	51		3	1	28
71	118	R	69 63		3	1	146
139	34	E	138		3	1	180
140	22	E	139		3	1	202
10	29	E	8		3	1	231
14	15	E	10 9 7 5		3	1	246
16	53	R	14		3	1	299
76	5	E	53		3	1	304
78	8	E	76		3	1	312
85	20	E	78		3	1	332
13	32	E	12		3	2	32
43	32	L	42		3	2	64
44	66	L	43		3	2	130
97	19	E	74		3	2	149
53	118	L	52 51		3	2	267
15	53	L	14		3	2	320
80	7	E	78		3	2	327
17	8	E	16 15		3	2	335
19	24	R	17		4	1	24
72	25	R	63 53		4	1	49
86	21	E	72 57		4	1	70
134	20	R	72		4	1	90
21	7	R	20		4	1	97
89	13	E	72 58 53		4	1	110
88	23	E	72 57		4	1	133
24	13	R	20		4	1	146
25	10	R	24 23 22 21		4	1	156
26	25	R	24 23 22 21		4	1	181
64	33	R			4	1	214
99	51	E	64		4	1	265
68	14	E	67		4	1	279
100	39	E	99		4	1	318
101	30	E	100 98 95		4	1	348
87	47	E	85		4	2	47
18	24	L	17		4	2	71
20	8	E	19 18		4	2	79

Table C.3 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
22	8	L	20		4	2	87
23	14	L	20		4	2	101
96	31	E	72		4	2	132
104	45	E	96		4	2	177
28	25	L	24 23 22 21		4	2	202
27	11	L	24 23 22 21		4	2	213
29	11	E	28 27 26 25		4	2	224
65	8	E	64		4	2	232
66	18	E	65		4	2	250
67	10	E	66 63		4	2	260
98	34	E	67		4	2	294
95	20	E	67		4	2	314
31	25	E	29		4	2	339
36	47	R	35 34 31		5	1	47
102	26	E	101		5	1	73
127	55	E	103 102		5	1	128
146	64	R	142 79		5	1	192
37	7	R	36		5	1	199
38	80	R	37		5	1	279
83	21	E	81		5	1	300
84	26	E	81		5	1	326
39	7	R	38		5	1	333
103	13	E	101		5	2	13
79	111	E	68		5	2	124
143	64	L	142 79		5	2	188
81	26	E	79		5	2	214
45	80	L	37		5	2	294
46	7	L	45		5	2	301
47	41	L	46		5	2	342
40	41	R	39		6	1	41
41	47	R	40		6	1	88
107	8	E	106		6	1	96
108	43	E	107		6	1	139
109	40	E	108		6	1	179
110	34	E	109		6	1	213
55	7	R	54		6	1	220
73	40	E	55		6	1	260
90	19	E	73 71 52		6	1	279

Table C.3 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
111	23	E	90		6	1	302
106	28	E	84		6	2	28
49	47	L	47		6	2	75
48	13	E	47 40		6	2	88
54	25	L	49 41		6	2	113
144	170	L			6	2	283
133	23	L	55		6	2	306
75	101	E	73 58		7	1	101
77	28	E	75		7	1	129
145	137	R	144		7	1	266
82	10	E	80 77 56		7	1	276
135	46	E	134 133 132 92		7	2	46
136	64	E	135		7	2	110
112	162	L	111		7	2	272
147	78	L	145 142		7	2	350
148	78	R	145 142		8	1	78
121	32	E	120		8	1	110
122	26	E	121		8	1	136
126	48	E	122 118		8	1	184
125	19	E	124 115		8	1	216
113	11	L	112		8	2	11
120	31	E	113		8	2	42
116	31	E	113		8	2	73
117	32	E	116		8	2	105
118	26	E	117		8	2	131
123	19	E	113		8	2	150
124	14	E	123		8	2	164
114	19	E	113		8	2	183
115	14	E	114		8	2	197
128	8	L	113		8	2	205
129	11	L	128		8	2	216
130	27	L	129		8	2	243
137	22	L	130		8	2	265
131	18	L	130		8	2	283

Table C.4 Heuristic Solution for 205-Task Problem

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
57	13	R			1	1	13
2	42	E			1	1	55
3	261	R	2		1	1	316
1	692	E			1	1	1008
25	86	R			1	1	1094
7	54	R	5		1	1	1148
8	67	R	7		1	1	1215
9	30	R	8		1	1	1245
10	106	R	9		1	1	1351
11	32	R	10		1	1	1383
22	37	E	21 20		1	1	1420
23	41	E	22		1	1	1461
19	56	E			1	2	56
4	261	L	2		1	2	317
5	157	E	4 3		1	2	474
51	217	L			1	2	691
52	72	L	51		1	2	763
6	90	E			1	2	853
21	86	E			1	2	939
53	85	L			1	2	1024
20	67	E			1	2	1091
13	54	L	5		1	2	1145
14	67	L	13		1	2	1212
15	30	L	14		1	2	1242
16	106	L	15		1	2	1348
17	32	L	16		1	2	1380
18	62	L	17		1	2	1442
34	15	L	23		1	2	1476
24	72	E	23		2	1	72
28	66	R	25 24		2	1	138
29	41	R	28		2	1	179
33	15	R	29		2	1	194
30	72	R	29		2	1	266
32	16	R	30		2	1	282
27	51	R	24		2	1	333
31	51	R	30		2	1	384
31	51	R	30		2	1	384
36	59	E	35 19 18 12 6 1		2	1	528

Table C.4 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
75	371	E	36		2	1	899
110	308	R	36		2	1	1207
62	43	E	36		2	1	1250
63	37	E	62		2	1	1287
64	37	E	63		2	1	1324
65	103	E	64		2	1	1427
12	62	L	11		2	2	62
26	16	L	24		2	2	88
35	85	E	34 33 32 31 27 26		2	2	469
69	51	E	36		2	2	579
70	88	E	69		2	2	667
42	80	E	36		2	2	747
37	23	L	36		2	2	770
38	13	L	37		2	2	783
111	116	L	36		2	2	899
71	53	E	70		2	2	952
39	19	L	38		2	2	971
41	214	E	36		2	2	1185
72	144	E	36		2	2	1329
68	35	E	64		2	2	1364
40	108	E	36		2	2	1472
73	337	E	72 71		3	1	337
74	107	E	73		3	1	444
76	97	E	74		3	1	541
79	92	R	76		3	1	633
54	43	R	42 40		3	1	676
55	97	R	54		3	1	773
61	85	R	55		3	1	858
56	37	R	55		3	1	895
58	35	R	56		3	1	930
77	166	E	76		3	1	1096
82	92	E	77		3	1	1188
112	312	R	36		3	1	1500
66	140	E	65		3	2	140
67	49	E	66		3	2	189
43	37	L	42 40		3	2	226
44	84	L	43		3	2	310

Table C.4 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
45	18	L	44 39		3	2	328
48	37	L	45		3	2	365
49	13	L	48		3	2	378
50	70	L	49		3	2	448
46	12	L	45		3	2	460
47	29	L	46		3	2	489
83	371	E	36		3	2	860
78	92	L	76		3	2	952
80	106	E	79 78 77 68 67		3	2	1202
81	49	E	80		3	2	1251
84	87	E	81		3	2	1338
85	162	E	84		3	2	1500
86	96	E	85		4	1	96
89	42	E	88		4	1	138
90	88	R	85		4	1	226
91	90	R	90		4	1	316
59	217	R	57 55		4	1	533
60	72	R	59		4	1	605
92	97	R	91 89 87 83 82 75 61 60 58 53 52 50 47 41		4	1	702
95	48	R	92		4	1	750
97	34	E	92		4	1	784
99	50	E	92		4	1	834
96	338	E	92		4	1	1172
93	270	R	92		4	1	1442
130	38	R	100		4	1	1480
88	96	E	85		4	2	96
87	79	E	86		4	2	175
94	452	E	92		4	2	1154
98	65	E	92		4	2	1219
100	112	E	99 98 97		4	2	1331
109	97	E	100		4	2	1428
103	50	E	100		4	2	1478
134	445	R	100		5	1	445
101	48	E	100		5	1	493
102	117	E	101		5	1	610

Table C.4 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
104	68	R	103		5	1	678
132	511	R	131		5	1	1455
131	944	E	100		5	2	944
105	232	L	100		5	2	1176
107	151	E	105		5	2	1327
106	122	L	105		5	2	1449
108	31	L	107 106		5	2	1480
133	625	R	132		6	1	625
163	68	R	113		6	1	693
161	68	R	113		6	1	761
121	66	E	113		6	1	827
117	55	E	113		6	1	882
116	175	R	113		6	1	1057
162	68	R	113		6	1	1125
205	165	E	113		6	1	1290
113	34	E	112 111 110 109 108 104 102 96 95		6	2	34
119	59	E	113		6	2	93
118	306	E	113		6	2	399
174	68	L	113		6	2	467
115	54	E	113		6	2	521
123	23	E	113		6	2	544
120	59	E	113		6	2	603
122	66	E	113		6	2	669
114	128	L	113		6	2	797
204	94	E	113		6	2	891
171	68	L	113		6	2	959
124	244	E	113		6	2	1203
125	54	E	124		6	2	1257
203	93	L	113		6	2	1350
169	68	L	113		6	2	1418
126	294	R	125 123 122 121 120 119 118		7	1	294
129	57	E	126		7	1	351
128	61	E	126		7	1	412
150	49	R	135		7	1	529
144	332	E	135		7	1	861

Table C.4 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
151	107	E	135		7	1	968
149	67	R	135		7	1	1035
148	58	R	135		7	1	1093
139	236	E	135		7	1	1329
158	35	R	135		7	1	1364
159	58	R	158		7	1	1422
154	68	E	153		7	1	1490
127	84	E	126		7	2	378
135	68	L	129 128 127 94 93		7	2	480
138	92	E	135		7	2	572
140	116	L	135		7	2	688
141	265	L	135		7	2	953
152	38	L	135		7	2	991
153	27	L	135		7	2	1018
145	324	E	135		7	2	1342
147	51	L	135		7	2	1393
136	53	L	135 130		7	2	1446
137	49	E	135		7	2	1495
155	207	E	154		8	1	207
156	202	E	155		8	1	409
178	187	E	160		8	1	596
164	103	R	163 160		8	1	699
165	103	R	164 162		8	1	802
166	103	R	165		8	1	905
167	103	R	166 161		8	1	1008
168	103	R	167		8	1	1111
177	10	E	176 168		8	1	1494
142	149	L	135		8	2	149
143	74	L	142 141 140		8	2	223
146	104	L	145		8	2	327
160	42	E	152 151 150 149 148 147 146 144 143 139 138 137 117 116 115 114		8	2	369
179	134	L	160		8	2	503
184	53	L	160		8	2	556

Table C.4 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
157	83	E	156		8	2	639
180	89	L	179 178		8	2	728
181	58	L	180		8	2	786
182	49	L	181		8	2	835
183	134	L	180		8	2	969
170	103	L	169 160		8	2	1072
172	103	L	171 170		8	2	1175
173	103	L	172		8	2	1278
175	103	L	174 173		8	2	1381
176	103	L	175		8	2	1484
186	24	R	177		9	1	24
187	76	R	177		9	1	100
194	115	E	177		9	1	215
196	56	R	195		9	1	473
199	18	R	196		9	1	491
200	29	R	199		9	1	520
189	192	E	188 187 186 185 159 157 136 134 133		9	1	712
198	303	R	196		9	1	1015
191	258	R	189		9	1	1273
197	29	R	196 194		9	1	1302
193	38	R	189		9	1	1340
185	334	E	177		9	2	334
195	83	L	177		9	2	417
188	76	L	177		9	2	493
201	154	L	196		9	2	647
202	90	L	201		9	2	737
190	98	E	189		9	2	835
192	165	E	191		9	2	1438

Table C.5 Heuristic Solution for Case Study Problem

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
38	4.7	E			1	1	4.7
12	16.5	E			1	1	21.2
13	4	E			1	1	25.2
2	4.3	E			1	2	4.3
9	4.3	E			1	2	8.6
3	9.8	E			1	2	18.4
4	3.6	E	3		1	2	22
5	1.8	E	3		1	2	23.8
14	24	R	13 12	15	2	1	24
15	24	L	13 12	14	2	2	24
43	3.5	R	15 14	48	3	1	3.5
47	2.9	R	43		3	1	6.4
50	2.6	R	48 43		3	1	9
51	6	R	50		3	1	15
10	2.9	R			3	1	17.9
11	2.9	R			3	1	20.8
29	1.1	R	21		3	1	21.9
34	3.3	E	29		3	1	25.2
48	4.3	L	15 14	43	3	2	4.3
17	5.6	L	15 14		3	2	9.9
19	6.3	L	17		3	2	16.2
21	3.4	L			3	2	19.6
42	4.8	L	15 14		3	2	24.4
52	2.2	R	51		4	1	2.2
70	2.9	R	48		4	1	5.1
1	13.6	E			4	1	18.7
7	3.6	E	1		4	1	22.3
49	2.2	R	48 43		4	1	24.5
20	21.3	L	19		4	2	21.3
58	3.1	L	52		4	2	24.4
22	8.4	R	20 3		5	1	8.4
46	6.1	R	22 1		5	1	14.5
25	3.8	E	22		5	1	18.3
26	4.4	R	22		5	1	22.7
53	2.5	E	52		5	1	25.2
33	22.8	L	20		5	2	22.8
56	2.4	L	52		5	2	25.2
8	11.3	E	1		6	1	11.3
6	4.3	E	3 1		6	1	15.6

Table C.5 (continued)

TASK	TASK TIME	TASK SIDE	TASK IP	SYN	IS(i)	T(i)	F(i)
23	3	R	22		6	1	18.6
30	2.7	E	23		6	1	21.3
31	2.6	E	30		6	1	23.9
32	22.2	L	20		6	2	22.2
39	3.2	L	33 32		6	2	25.4
55	2.2	R	53		7	1	2.2
45	3.7	E	25		7	1	5.9
24	3.7	E	22		7	1	9.6
44	3.7	E	25		7	1	13.3
27	10	R	22		7	1	23.3
66	1.9	R	62		7	1	25.2
59	6.4	L	52		7	2	6.4
61	2.2	L	59		7	2	8.6
60	3.3	L	59		7	2	11.9
62	7.5	L	60		7	2	19.4
57	2.9	L	52		7	2	22.3
18	2.9	L	15 14		7	2	25.2
63	2.3	R	62		8	1	2.3
65	1.6	R	62		8	1	3.9
64	1.7	R	62		8	1	5.6
71	3.5	E	66 65 64 63		8	1	9.1
67	3.5	E	66 65 64 63		8	1	12.6
28	3.2	R	27		8	1	15.8
54	4.5	R	52		8	1	20.3
16	4.3	L	15 14		8	2	4.3
35	4.1	L	16		8	2	8.4
36	2.4	L	35		8	2	10.8
37	6	L	35		8	2	16.8
68	7.5	L	67		8	2	24.3
69	6.7	E	68		9	2	6.7
73	2.4	L	69		9	2	9.1
72	5.4	L	69		9	2	14.5
40	4.1	L	16		9	2	18.6
41	2.4	L	40		9	2	21
74	2.6	L	69		9	2	23.6

Table C.6 Experimental Study with Heuristic Procedure

			Number of Workers			
SPF	OS	CT	20-Task	30-Task	40-Task	50-Task
0.3	0.2	250	7	12	14	18
		300	6	9	12	15
	0.4	250	9	12	16	18
		300	7	10	14	15
	0.6	250	10	13	16	21
		300	7	10	14	18
	0.8	250	11	15	18	22
		300	9	12	15	19
0.5	0.2	250	7	12	14	18
		300	6	10	12	14
	0.4	250	9	12	16	18
		300	8	10	13	15
	0.6	250	10	13	17	20
		300	8	10	14	18
	0.8	250	11	14	18	22
		300	9	11	16	18
0.7	0.2	250	8	12	14	18
		300	6	10	12	14
	0.4	250	9	12	14	18
		300	7	10	12	15
	0.6	250	10	13	16	20
		300	8	10	13	17
	0.8	250	10	14	17	22
		300	8	11	14	18

Table C.7 Heuristic Solution for Scenario 1 and Actual Finishing Times

Heuristic Solution					Actual Finishing Times			
Mated Station	**Side**	**Task**	**F(i)**		**Model**	**Model**	**Model**	**Model**
1	R	10	2.9		2.9	2.9	2.9	2.9
		11	5.4		2.9	5.9	5.9	5.9
		1	17.6		13.6	12.7	19.5	19.5
		7	18.7		17.2	16.2	19.5	19.5
		38	22		17.2	16.2	24.2	24.2
	L	13	4		4.0	4.0	4.0	4.0
		2	7.6		4.0	8.2	8.2	8.2
		12	24.1		20.4	24.7	24.7	24.7
		21	24.6		23.8	24.7	24.7	24.7
2	R	14	24		24.0	24.0	24.0	24.0
		29	25.1		25.1	25.1	25.1	25.1
	L	15	24		24.0	24.0	24.0	24.0
3	R	34	2.2		3.3	3.3	0.0	3.3
		8	10.6		14.6	3.3	9.6	12.9
		3	20.2		22.7	13.2	19.4	22.8
		43	23.7		26.2	16.6	22.9	26.2
		5	25.5		28.0	18.4	24.7	28.0
	L	16	4.3		4.3	4.3	4.3	4.3
		40	7.2		4.3	4.3	8.5	8.5
		41	8.9		4.3	4.3	10.9	10.9
		18	11.8		7.2	7.2	13.7	13.7
		17	16.5		7.2	12.8	19.3	19.3
		42	19.9		7.2	12.8	24.1	24.1
		48	24.5		27.0	17.5	23.7	27.1
4	R	50	2.6		2.6	2.6	2.6	2.6
		51	6.1		7.1	7.1	2.6	8.6
		52	8.3		9.2	9.2	4.7	10.7
		53	10		9.2	9.2	7.2	13.2
		55	12.2		11.5	11.5	9.4	15.4
		54	16.7		16.0	16.0	13.9	19.9
		70	19.6		18.9	18.9	16.9	22.9
		47	22.5		21.8	21.8	19.8	25.8
		49	24.7		24.0	24.0	21.9	27.9

Table C.7 (continued)

Heuristic Solution					Actual Finishing Times			
Mated Station	Side	Task	F(j)		Model I	Model II	Model III	Model IV
4	L	4	3.6		3.5	3.6	3.6	3.6
		6	6.6		3.5	3.6	7.9	7.9
		19	11.9		3.5	9.9	14.2	14.2
		56	14.3		11.6	12.3	16.6	16.6
		57	17.2		14.5	15.2	19.5	19.5
		58	20.3		17.6	18.3	22.6	22.6
		9	24		17.6	22.6	26.9	26.9
5	L	20	21.3		21.3	21.3	21.3	21.3
		35	24		25.4	25.4	21.3	25.4
6	R	22	7.2		0.0	8.4	8.4	8.4
		24	10.3		0.0	12.1	12.1	12.1
		26	14.3		2.7	15.5	16.5	16.5
		46	20.4		8.9	21.7	22.6	22.6
		25	23		8.9	21.7	26.4	26.4
		23	23.5		11.9	21.7	26.4	26.4
		66	25.4		21.0	28.6	28.3	29.8
	L	37	3.5		4.8	4.8	0.0	6.0
		36	5.1		7.2	7.2	0.0	8.4
		59	11.5		13.6	13.6	6.4	14.8
		61	13.7		15.8	15.8	8.6	17.0
		60	17		19.1	19.1	11.9	20.3
		62	23.4		19.1	26.6	19.4	27.8
7	R	30	2.7		2.7	2.7	2.7	2.7
		31	5.3		5.3	5.3	5.3	5.3
		44	9		9.0	9.0	9.0	9.0
		65	10.6		10.6	10.6	10.6	10.6
		63	12.9		12.9	12.9	12.9	12.9
		45	16.6		16.6	16.6	16.6	16.6
		64	18		16.6	18.3	18.3	18.3
		71	21.5		20.0	21.7	21.7	21.7
		67	24.4		20.0	25.2	25.2	25.2
	L	33	22.8		22.8	22.8	22.8	22.8
8	R	27	8.5		0.0	10.0	10.0	10.0
		28	11.7		3.2	13.2	13.2	13.2
	L	68	7.5		7.5	7.5	7.5	7.5
		69	13.2		7.5	14.2	14.2	14.2
		72	18.6		12.9	19.6	19.6	19.6
		73	21		15.3	22.0	22.0	22.0
		74	23.6		17.9	24.6	24.6	24.6
9	L	32	22.2		22.2	22.2	22.2	22.2
		39	24.5		22.2	22.2	25.4	25.4

Table C.8 Heuristic Solution for Scenario 2 and Actual Finishing Times

Heuristic Solution					Actual Finishing Times			
Mated Station	**Side**	**Task**	**F(i)**		**Model I**	**Model II**	**Model III**	**Model IV**
1	R	1	11.2		10.7	6.8	13.6	13.6
		8	18.8		22.0	6.8	23.2	23.2
		11	21.0		22.0	9.8	26.1	26.1
		29	22.1		23.0	10.9	27.2	27.2
		34	24.6		26.4	14.2	27.2	30.5
	L	21	8		3.4	0.0	0.0	0.0
		12	17.3		19.8	16.5	16.5	16.5
		13	21.3		23.8	20.4	20.4	20.4
		38	23.7		23.8	20.4	25.1	25.1
		7	25.5		27.3	24.0	25.1	25.1
2	R	14	24.0		24.0	24.0	24.0	24.0
	L	15	24.0		24.0	24.0	24.0	24.0
3	R	3	9.4		8.1	9.8	9.8	9.8
		4	13.0		11.6	13.4	13.4	13.4
		2	16.2		11.6	17.7	17.7	17.7
		43	19.7		15.1	21.2	21.2	21.2
		47	22.6		18.0	24.1	24.1	24.1
		70	25.5		20.9	27.1	27.1	27.1
	L	17	4.2		0.0	5.6	5.6	5.6
		19	8.9		0.0	11.9	11.9	11.9
		42	11.3		0.0	11.9	16.7	16.7
		5	13.1		9.9	13.7	18.5	18.5
		48	20.5		15.9	22.0	22.0	22.0
		9	23.7		15.9	26.3	26.3	26.3
4	R	49	2.2		2.2	2.2	2.2	2.2
		50	4.8		4.7	4.7	4.7	4.7
		51	8.6		9.2	9.2	4.7	10.7
		52	10.8		11.4	11.4	6.9	12.9
		54	15.3		15.9	15.9	11.4	17.4
		53	16.5		15.9	15.9	13.8	19.8
		55	18.7		18.1	18.1	16.1	22.1
		10	21.6		21.1	21.1	19.0	25.0
		6	23.8		21.1	21.1	23.3	29.3
	L	20	21.3		21.3	21.3	21.3	21.3
		56	23.7		23.7	23.7	23.7	23.7

Table C.8 (continued)

Heuristic Solution				Actual Finishing Times			
Mated Station	**Side**	**Task**	**F(i)**	**Model I**	**Model II**	**Model III**	**Model IV**
5	R	22	6.3	0.0	8.4	8.4	8.4
		46	12.4	6.1	14.6	14.6	14.6
		26	16.1	8.9	18.0	19.0	19.0
		23	16.9	11.9	18.0	19.0	19.0
		30	19.6	14.6	20.7	21.7	21.7
		31	22.2	17.2	23.3	24.2	24.2
		65	23.8	18.8	24.9	25.8	25.8
		64	25.1	18.8	26.6	27.5	27.5
	L	58	3.1	3.1	3.1	3.1	3.1
		59	9.5	9.5	9.5	9.5	9.5
		60	12.8	12.7	12.7	12.7	12.7
		62	18.5	12.7	20.3	20.3	20.3
		61	20.7	15.0	22.5	22.5	22.5
		57	23.6	17.9	25.4	25.4	25.4
		25	25.5	17.9	25.4	29.2	29.2
6	R	44	3.7	3.7	3.7	3.7	3.7
		63	6.0	6.0	6.0	6.0	6.0
		66	7.9	7.9	7.9	7.9	7.9
		71	11.4	11.4	11.4	11.4	11.4
		27	18.9	11.4	21.4	21.4	21.4
		28	22.1	14.6	24.6	24.6	24.6
	L	45	3.7	3.7	3.7	3.7	3.7
		24	6.5	3.7	7.4	7.4	7.4
		18	9.4	6.5	10.2	10.2	10.2
		67	12.0	7.9	13.7	13.7	13.7
		68	19.5	15.5	21.2	21.2	21.2
		69	24.5	15.5	27.9	27.9	27.9
7	L	72	5.4	5.4	5.4	5.4	5.4
		73	7.8	7.8	7.8	7.8	7.8
		16	12.1	12.1	12.1	12.1	12.1
		35	15.2	16.3	16.3	12.1	16.3
		36	17.0	18.7	18.7	12.1	18.7
		37	20.9	23.5	23.5	12.1	24.7
		40	23.0	23.5	23.5	16.3	28.8
		41	24.2	23.5	23.5	18.7	31.2
8	L	33	22.8	22.8	22.8	22.8	22.8
		74	25.4	25.4	25.4	25.4	25.4
9	L	32	22.2	22.2	22.2	22.2	22.2
		39	23.8	22.2	22.2	25.4	25.4

Table C.9 Heuristic Solution for Scenario 3 and Actual Finishing Times

Heuristic Solution					Actual Finishing Times			
Mated Station	Side	Task	F(j)		Model I	Model II	Model III	Model IV
1	R	9	2.8		0.0	4.3	4.3	4.3
		38	4.2		0.0	4.3	9.1	9.1
		12	20.7		16.5	20.8	25.5	25.5
		10	23.6		19.4	23.7	28.4	28.4
		29	24.7		20.5	24.8	29.5	29.5
	L	13	4		4.0	4.0	4.0	4.0
		3	13.2		12.1	13.8	13.8	13.8
		4	16.8		15.5	17.4	17.4	17.4
		5	18.6		17.3	19.2	19.2	19.2
		2	21.4		17.3	23.5	23.5	23.5
		21	22.6		20.7	23.5	23.5	23.5
2	R	14	24		24.0	24.0	24.0	24.0
	L	15	24		24.0	24.0	24.0	24.0
		42	25.4		24.0	24.0	28.8	28.8
3	R	43	6.4		3.5	3.5	3.5	3.5
		47	9.3		6.4	6.4	6.4	6.4
		50	11.9		9.7	9.7	9.7	9.7
		70	14.8		12.7	12.7	12.7	12.7
		49	17		14.8	14.8	14.8	14.8
		51	21.1		19.3	19.3	14.8	20.8
		52	23.3		21.5	21.5	17.0	23.0
		53	24		21.5	21.5	19.4	25.4
	L	18	2.9		2.9	2.9	2.9	2.9
		48	7.2		7.2	7.2	7.2	7.2
		1	17.4		17.8	14.0	20.8	20.8
		7	19.9		21.4	17.5	20.8	20.8
		6	21.2		21.4	17.5	25.1	25.1
		34	24		24.7	20.8	25.1	28.4
4	R	55	2.2		2.2	2.2	2.2	2.2
		8	9		13.5	2.2	11.8	11.8
		54	13.5		18.0	6.7	16.3	16.3
		11	15.4		18.0	9.7	19.3	19.3
	L	58	3.1		3.1	3.1	3.1	3.1
		59	9.5		9.5	9.5	9.5	9.5
		61	11.7		11.7	11.7	11.7	11.7
		57	14.6		14.6	14.6	14.6	14.6
		56	17		17.0	17.0	17.0	17.0
		17	20.6		17.0	22.5	22.5	22.5
		19	24.7		17.0	28.8	28.8	28.8

Table C.9 (continued)

Heuristic Solution					Actual Finishing Times			
Mated Station	**Side**	**Task**	**F(i)**		**Model I**	**Model II**	**Model III**	**Model IV**
5	L	20	21.3		21.3	21.3	21.3	21.3
		60	24.6		24.5	24.5	24.5	24.5
6	R	22	5.5		0.0	8.4	8.4	8.4
		23	6.6		3.0	8.4	8.4	8.4
		65	8.2		4.6	10.0	10.0	10.0
		27	14.7		4.6	20.0	20.0	20.0
		63	17		7.0	22.3	22.3	22.3
		26	20.5		9.7	25.8	26.7	26.7
		64	21.6		9.7	27.5	28.5	28.5
		66	23.5		11.6	29.4	30.4	30.4
	L	62	4.9		0.0	7.5	7.5	7.5
		16	9.2		4.3	11.9	11.9	11.9
		25	10.3		4.3	11.9	15.6	15.6
		44	14		8.0	15.6	19.3	19.3
		40	15.2		8.0	15.6	23.4	23.4
		41	15.9		8.0	15.6	25.8	25.8
		45	19.6		11.7	19.2	29.5	29.5
		35	23.1		15.8	23.4	29.5	33.6
		36	25.1		18.2	25.8	29.5	36.0
7	R	67	2.2		0.0	3.5	3.5	3.5
		46	8.3		6.1	9.6	9.6	9.6
		71	11.8		9.6	13.0	13.0	13.0
		30	14.5		12.3	15.7	15.7	15.7
		31	17.1		14.9	18.3	18.3	18.3
		24	19.5		14.9	22.0	22.0	22.0
		28	22.7		18.0	25.2	25.2	25.2
	L	37	4.3		4.8	4.8	0.0	6.0
		68	11.8		12.3	12.3	7.5	13.5
		69	16.1		12.3	19.0	14.2	20.2
		73	18.5		14.7	21.4	16.6	22.6
		72	23.9		20.1	26.8	22.0	28.0
8	L	33	22.8		22.8	22.8	22.8	22.8
		74	25.4		25.4	25.4	25.4	25.4
9	L	32	22.2		22.2	22.2	22.2	22.2
		39	23.2		22.2	22.2	25.4	25.4