# MODELLING AND SOLVING MIXED-MODEL PARALLEL TWO-SIDED ASSEMBLY LINE PROBLEMS

Submitted by **Ibrahim Kucukkoc**,
to the University of Exeter as a thesis for the degree of
Doctor of Philosophy in Engineering, in **August 2015**.

Signature: …………………………………………………………..

# Abstract

The global competitive environment and the growing demand for personalised products have increased the interest of companies in producing similar product models on the same assembly line. Companies are forced to make significant structural changes to rapidly respond to diversified demands and convert their existing single-model lines into mixed-model lines in order to avoid unnecessary new line construction cost for each new product model. Mixed-model assembly lines play a key role in increasing productivity without compromising quality for manufacturing enterprises.

The literature is extensive on assembling small-sized products in an intermixed sequence and assembling large-sized products in large volumes on single-model lines. However, a mixed-model parallel two-sided line system, where two or more similar products or similar models of a large-sized product are assembled on each of the parallel two-sided lines in an intermixed sequence, has not been of interest to academia so far. Moreover, taking model sequencing problem into consideration on a mixed-model parallel two-sided line system is a novel research topic in this domain. Within this context, the problem of *simultaneous balancing and sequencing of mixed-model parallel two-sided lines* is defined and described using illustrative examples for the first time in the literature. The mathematical model of the problem is also developed to exhibit the main characteristics of the problem and to explore the logic underlying the algorithms developed. The benefits of utilising multi-line stations between two adjacent lines are discussed and numerical examples are provided.

An agent-based ant colony optimisation algorithm (called ABACO) is developed to obtain a generic solution that conforms to any model sequence and it is enhanced step-by-step to increase the quality of the solutions obtained. Then, the algorithm is modified with the integration of a model sequencing procedure (where the modified version is called ABACO/S) to balance lines by tracking the product model changes on each workstation in a complex production environment where each of the parallel lines may a have different cycle time. Finally, a genetic algorithm based model sequencing mechanism is integrated to the algorithm to increase the robustness of the obtained solutions. Computational tests are performed using test cases to observe the performances of the developed algorithms.

Statistical tests are conducted through obtained results and test results establish that balancing mixed-model parallel two-sided lines together has a significant effect on the sought performance measures (a weighted summation of line length and the number of workstations) in comparison with balancing those lines separately. Another important finding of the research is that considering model sequencing problem along with the line balancing problem helps algorithm find better line balances with better performance measures. The results also indicate that the developed ABACO and ABACO/S algorithms outperform other test heuristics commonly used in the literature in solving various line balancing problems; and integrating a genetic algorithm based model sequencing mechanism into ABACO/S helps the algorithm find better solutions with less amount of computational effort.

# Acknowledgements

I would like to thank all help and support I have received from the following people during the course of my PhD research.

Above all, I would like to express my special and deepest appreciation to my family; *my beautiful wife Emine and lovely son Eren*, you are my daily inspiration and the foundation of all I do. Very special thanks to *my mother and father*, and *my brother and sister*. You all have provided me your unique love and support, as always, for which my mere expression of thanks likewise does not suffice. Special thanks also go to *my parents-in-law and brothers-in-law;* whose constant encouragement and support at every stage of my research were invaluable for me.

I would like to express my sincere gratitude to my supervisor, *Prof. David Z. Zhang*, whose continuous support, patience, motivation, enthusiasm, and immense knowledge helped me throughout my PhD research. His invaluable guidance and constructive comments helped me a lot in every stage of this research, writing of this thesis, attending conferences and publishing papers in esteemed journals. Also, I am grateful to *Dr Edward C. Keedwell* who has widened my horizons in developing nature inspired algorithms.

Many thanks to all of my colleagues in *Exeter Manufacturing Enterprise Centre (XMEC)*, and other staff members in the *College of Engineering, Mathematics and Physical Sciences* for their kind assistance.

I am extremely grateful for the financial support of the *Turkish Council of Higher Education (YÖK, shortly in Turkish)* and *Balikesir University* and their staff, particularly in obtaining the award of a PhD Research Scholarship that provided the necessary fund to carry out this research in England.

My sincere thanks also go to *all my relatives and friends* overseas for being just one call away and supporting me through this entire process.

# Table of Contents

# List of Tables

# List of Figures

# Abbreviations

The abbreviations used in the thesis are presented as follows. Please note that the notation that will be used in the equations will be introduced as the need arises.

| | |
|---|---|
| 2-ANTBAL | : Ant colony optimisation algorithm with two ants |
| ABACO | : Agent-based ant colony optimisation |
| ABACO/S | : Agent-based ant colony optimisation (including sequencing) |
| ABACO/S-GA | : Hybrid agent-based ACO-GA algorithm (including sequencing) |
| ACO | : Ant colony optimisation |
| ANOVA | : Analysis of variance |
| ANTBAL | : An ACO Approach developed for assembly line balancing problem |
| BA | : Balancing Agent |
| BC | : Buffer capacity |
| BOH | : Best other heuristic |
| BOH/S | : Best other heuristic (including sequencing) |
| B/S | : Balancing and sequencing |
| C | : Cycle time |
| CBS | : Colony best solution |
| COMSOAL | : Computer method of sequencing operations for assembly lines |
| CoS | : Cost of setups |
| E | : Either side of a two-sided line |
| EW | : Weighted workload difference between stations |
| FA | : Facilitator agent |
| GA | : Genetic algorithm |
| JIT | : Just-in-time |
| L | : Left side of a two-sided line |
| LBT | : Last Best Time |
| LCM | : Least common multiple |
| LE | : Line efficiency |
| LL | : Line length |
| LNP | : Least number of predecessors |
| LNS | : Least number of successors |
| LPT | : Longest processing time |
| MAL | : Mixed-model assembly line |
| MMS | : Mixed-model sequencing |
| MNS | : Maximum number of successors |
| MNP | : Maximum number of predecessors |
| MPS | : Minimum part set |
| MPTALB/S | : Mixed-model parallel two-sided assembly line balancing and sequencing |
| MTO | : Make to order |
| NI | : Number of iterations |
| NM | : Number of mated-stations |
| NS | : Number of stations |
| O | : Other special objective(s) |
| OBJ | : Objective value |
| PA | : Planning agent |
| PhD | : Doctor of Philosophy |
| PSONK | : Particle swarm optimisation with negative knowledge |

| | |
|---|---|
| PUR | : Part usage rate |
| PW | : Physical workload |
| R | : Right side of a two-sided line |
| RB | : Request for a new balancing solution |
| RI | : Rate of incomplete jobs |
| RPWM | : Ranked positional weight method |
| RQ | : Research question |
| RRPWM | : Reverse ranked positional weight method |
| RSM | : Response surface methodology |
| SA | : Sequencing agent |
| SD | : Standard deviation |
| SALBP | : Simple assembly line balancing problem |
| SALBP-X | : Simple assembly line balancing problem of type-X |
| SAn | : Simulated annealing |
| SPT | : Shortest processing time |
| STN | : Smallest task number |
| TUW | : Total utility work |
| WIP | : Work in process |
| WIT | : Weighted idle time |
| WS | : Workstation |
| WLS | : Workload smoothness |
| WLE | : Weighted line efficiency |
| WR | : Work-relatedness |

# Publications

## Journal Publications:

- Kucukkoc, I., Zhang, D. Z. (revision submitted), Mixed-model Parallel Two-sided Assembly Line Balancing Problem: A Flexible Agent based Ant Colony Optimisation Approach, *Computers & Industrial Engineering*.
- Kucukkoc, I., Zhang, D. Z. (2015), Integrating ant colony and genetic algorithms in the balancing and scheduling of complex assembly lines, *The International Journal of Advanced Manufacturing Technology*, doi: 10.1007/s00170-015-7320-y.
  *http://link.springer.com/article/10.1007%2Fs00170-015-7320-y*
- Kucukkoc, I., Zhang, D. Z. (2015), Type-E parallel two-sided assembly line balancing problem: Mathematical model and ant colony optimisation based approach with optimised parameters, *Computers & Industrial Engineering*, 84, 56-69, doi: 10.1016/j.cie.2014.12.037.
  *http://www.sciencedirect.com/science/article/pii/S0360835214004665*
- Kucukkoc, I., Zhang, D. Z. (2015), A Mathematical Model and Genetic Algorithm-based Approach for Parallel Two-sided Assembly Line Balancing Problem, *Production Planning & Control: The Management of Operations*, 26(11), 874-894, doi: 10.1080/09537287.2014.994685.
  *http://www.tandfonline.com/doi/full/10.1080/09537287.2014.994685*
- Kucukkoc, I., Zhang, D. Z. (2014), Mathematical Model and Agent based Solution Approach for the Simultaneous Balancing and Sequencing of Mixed-model Parallel Two-sided Assembly Lines, *International Journal of Production Economics*, 158, 314-333, doi: 10.1016/j.ijpe.2014.08.010.
  *http://www.sciencedirect.com/science/article/pii/S0925527314002680*
- Kucukkoc, I., Zhang, D. Z. (2014), Simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines, *International Journal of Production Research*, 52(12), 3665-3687, doi: 10.1080/00207543.2013.879618.
  *http://www.tandfonline.com/doi/full/10.1080/00207543.2013.879618*

## Conference Publications:

- Kucukkoc, I., Zhang, D. Z. (2014), An Agent based Ant Colony Optimisation Approach for Mixed-model Parallel Two-sided Assembly Line Balancing Problem, *Pre-prints of the Eighteenth International Working Seminar on Production Economics*, Innsbruck, Austria, February 24-28, Volume 3, 313-328.
- Kucukkoc, I., Zhang, D. Z. (2013), Balancing Parallel Two-sided Assembly Lines via a Genetic Algorithm based Approach, *Proceedings of 43rd International Conference on*

*Computers and Industrial Engineering (CIE43)*, The University of Hong Kong, Hong Kong, October 16-18, Volume 1, 106-121.
http://goo.gl/pFxNsZ

- Zhang, D. Z., Kucukkoc, I. (2013), Balancing Mixed-model Parallel Two-sided Assembly Lines, *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IEEE – IESM 2013)*, Rabat, Morocco, October 28-30, Article Number 6761429, 391-401 (ISBN: 978-2-9600532-4-1).
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6761429
- Kucukkoc, I., Zhang, D. Z. (2013), On Applications of Ant Colony Optimisation Techniques in Solving Assembly Line Balancing Problems, *OR55 Annual Conference Keynote Papers and Extended Abstracts Book*, The OR Society UK, University of Exeter, Exeter, September 3-5, pp.114-120.
http://goo.gl/VDbBsh
- Kucukkoc, I., Zhang, D. Z., Keedwell, E. C. (2013), Balancing Parallel Two-sided Assembly Lines with Ant Colony Optimisation Algorithm, *Proceedings of the 2nd Symposium on Nature-Inspired Computing and Applications, NICA 2013 – Held at the AISB Convention 2013*, University of Exeter, Exeter, April 3-5, pp.21-28.
http://goo.gl/d8Se0V
- Kucukkoc, I., Zhang, D.Z., Keedwell, E. C. (2013), An Improved Ant Colony Optimisation Algorithm for Type-I Parallel Two-sided Assembly Line Balancing Problem, *YOR18 Biennial Conference*, The OR Society UK, University of Exeter, Exeter, April 9-11.

*The following link provides an up to date list of author's publications:*

http://www.researcherid.com/rid/C-1207-2011

# 1

# INTRODUCTION

**Contents**

## 1.1. Chapter Introduction

This chapter provides an overview of the research. After a very brief research background about assembly line balancing problem and its historical development in Section 1.2, an overview of the assembly line balancing problems with the main characteristics and nomenclature are presented in Section 1.3. Some definitions and notations related to assembly line systems are studied first, followed by the mathematical model and assumptions of the simple straight assembly line balancing problem. Various key concepts; including product variations, different layout types, task processing times, parallel workstations and some more realistic constraints are also presented in this chapter. Section 1.4 provides discussion on research questions. Section 1.5 describes the major objectives to be achieved within the scope of this Doctor of Philosophy (PhD) thesis. The structure of the thesis is revealed in Section 1.6 and finally, a summary of the chapter is given in Section 1.7.

## 1.2. Research Background

Assembly lines are the most crucial constituents of mass production systems and provide improved labour productivity especially for companies which have to produce high volume products in a cost effective manner within a reasonable time. An assembly line is a sequence of workstations at which a group of tasks related to the assembly of a product are performed by operators or robots within designated time (called cycle time) for this line. Workstations are connected together by a material handling system which moves work-pieces from one workstation to another (Kara *et al*., 2010). The throughput level of a line is one of the key factors which determine the capacity of an entire manufacturing system.

Assembly line balancing problem is to assign tasks into an ordered sequence of workstations with the aim of maximising efficiency of the line, where line efficiency corresponds to the time needed to perform all tasks divided by the time allocated to perform all tasks. Some constraints, such as precedence relationships and capacity constraints, which will be explained in detail in subsequent sections, need to be satisfied in doing so. It is one of the most important problems in designing and managing assembly lines (Ozbakir and Tapkan, 2011).

The initial serious attempts to increase productivity by using carefully designed manufacturing operations that comprised of machine-assisted specialised labour emerged in the eighteenth century in England. With the industrial revolution (1750-1900), the manufacturing industry experienced crucial structural changes and companies started to adopt mass production techniques to increase capacity and productivity of their manufacturing systems. Henry Ford and his colleagues in the US later have been referred to as pioneers of developing the first moving-belt (or conveyor system) to assemble flywheel magnetos, in 1913 (Tanenbaum and Holstein, 2012). Although the early aim of Ford was to produce only a 'horseless carrier', as a common idea (Ford, 2009), sales of model-T passed 250 thousand in 1914 through the efficiency of the assembly line put into practice. Since then, assembly line balancing problems have been considered by a large number of researchers from academia and industry to enable companies to satisfy customer demands on time and keep up with the changes in global dynamic business environment.

Also, assembly line balancing domain has been of continuing interest to both academia and industry with the development of manufacturing systems (Ghosh and Gagnon, 1989). The idea of distributing tasks among workstations, which is referred to as the core of assembly line balancing problem, was introduced by Bryton (1954) and Helgeson *et al.* (1954). Salveson (1955) was the first to model the problem in its mathematical form through a linear programming formulation.

Only trial-and-error methods were used to balance the lines during the first forty years of the existence of assembly lines. Since then, many types of assembly lines have been studied and various types of solution methods have been developed to solve these complex problems (Erel and Sarin, 1998).

Over time, the configuration of the lines has evolved with characteristics of the products being assembled on them. In recent years, two-sided assembly lines (on which more information will be given in detail in Section 2.2) are one of the widely seen layout types in industry to assemble large-sized products such as automobiles, buses and trucks. In such a two-sided assembly line system, operators that work at opposite workstations (called mated-stations) of the same line perform their jobs concurrently on the same product, within the same cycle. Therefore, two-sided lines provide more flexibility to accomplish synchronised tasks at opposite sides of the same line.

A two-sided assembly line offers several advantages over a one-sided line, such as reduced material handling cost and tools/equipment cost, less throughput time (and so increased line efficiency), shorter line length, and less set-up time (Taha *et al*., 2011). However, balancing a two-sided line requires more constraints to be taken into consideration such as the operation directions and interference. Some operations require to be performed at one specific side of the line (Left-L, or Right-R) while some operations can be performed at either side of the line (Either-E). For that reason, precedence relationships must be checked very carefully while balancing a two-sided line as tasks which have precedence relationships between each other can be performed on opposite sides of the same line. In such a situation, all predecessors of a task must have been completed before initialising that particular task. In other words, if the two close related tasks are assigned to mated-stations, one of them must wait the completion of the other. This situation is called interference and violation of this rule may yield infeasible solutions (Hu *et al*., 2010, Ozbakir and Tapkan, 2011).

On the other hand, parallel assembly lines (on which more information will be given in detail in Section 2.3) are another line configuration type in the industry. Despite the fact that they have been in use for several years, parallel assembly line balancing problem has been addressed in the last decade by Gökçen *et al*. (2006). These lines have attracted researchers from academia and industry notwithstanding it requires high first construction expense and more layout space. The reason is that, parallel lines not only provide flexibility and productivity in the manufacturing of different types of products, but also increase the strength of the system against any unforeseeable breakdown, which may be caused by many reasons.

With the popularity of mass customisation in the last few decades, mixed-model assembly lines became one of the most appealed systems in various sectors. The flexibility of producing similar product models on the same line helps manufacturers deal with changing customer requirements in a cost effective manner. However, this brings the complexity of model sequencing problem, which is an important issue that needs to be taken into account in mixed-model lines for short term planning periods along with long term line balancing decisions. The reason is that, tasks may have different processing times for different product models and the workloads of workstations utilised across the line (so efficiency of the line system) vary depending on the sequence of product models produced as well as the balance configuration of the line. That is why model sequencing problem is also considered within the scope of this thesis.

Although a large number of studies exist on various types of aforementioned assembly line systems, individually; there is no study which considers producing mixed-models on two-sided lines located in parallel to each other. Moreover, simultaneous line balancing and model sequencing problem has not been studied for producing mixed-models on parallel two-sided assembly lines. Based on this motivation, *mixed-model parallel two-sided assembly line balancing and sequencing (MPTALB/S)* problem constitutes the main topic of this thesis.

To show the complexity of the assembly line balancing problem, Wee and Magazine (1982) considered the *simple assembly line balancing problem (SALBP)* under the assumptions that there are: *(i)* no precedence relationships and *(ii)* no grouping preferences between tasks. Thus the problem was reduced to packing tasks into few number of workstations, which is the same as the well-studied NP-hard problem, called 'bin-packing problem'. By this way, it was shown that the SALBP is an NP-hard class of combinatorial optimisation problem, which means it is difficult to obtain an optimal solution within reasonable computational times when the problem size increases. That is why the solution space grows exponentially with the increasing number of tasks that need to be allocated into workstations (Wu *et al*., 2008). It is widely believed that no easy method exists to solve assembly line balancing problems since these problems have extensively been studied for several decades (Rekiek and Delchambre, 2006). This is the major reason why a considerable amount of researches in the literature strives to develop heuristics and meta-heuristics instead of exact algorithms to solve assembly line balancing problems.

The MPTALB/S problem studied in this thesis is a much more sophisticated version of SALBP and is beyond the complexity of NP-hardness. It is obvious that considering precedence relationships between tasks which have particular operation side (left and/or right) preferences, and producing various models of a base product on a same line, where a task may have a different processing time for each product model, significantly increase the complexity of the problem. Furthermore, handling line balancing and model sequencing problems concurrently on mixed-model parallel two-sided assembly lines in such an environment that multi-line stations are established between lines, which may have different cycle times, increases the complexity to a great extent and makes the problem even harder to solve. To give an insight about this, please refer to numerical examples which will be provided when defining the problem and solving it using proposed algorithms starting from Chapter 4.

## 1.3. Fundamentals and Definitions

Assembly lines are most used flow oriented production systems in mass production environment and assembly line balancing problem has been dealt with for more than 50 years by several researchers (Salveson, 1955, Arcus, 1966, Yaman, 2008). Workers who perform partial *tasks*, where a task is referred to as a minimum rational work element and represented with $i$ (where $i = 1, \dots, T$), in workstations through the assembly line can assemble complex products (Gunasekaran and Cecille, 1998, Becker and Scholl, 2006). To clarify, a task is a smallest indivisible work element that cannot be split into two or more stations without conflict, and that is separable from other activities. To give an example, the process of mounting a component on a product model being assembled on the line represents an indivisible work element, called a task.

Line balancing is to divide the total workload of assembly line into several *workstations*, represented by $k$ (where $k = 1, \dots, K$) and to determine which task will be performed at which workstation (while each task is allocated exactly once). Generally, these workstations are linked together by a transportation system whose primary mission is to move products among serially constructed workstations (Bautista and Pereira, 2011).

A set of tasks is performed at each workstation and each task has its own *processing time* $(pt_i)$. Due to technological and organisational conditions, *precedence constraints*, which are usually represented as a network, must be satisfied in the assignment process (Sarker and Shanthikumar, 1983, Simaria, 2006). *Workload* (or *station time*) of a workstation, the time required to perform the set of tasks assigned to that workstation, cannot exceed *cycle time (C)* determined by the designer or manager of the line. Hence, the production rate of the system is determined by cycle time (Darel and Cother, 1975, Simaria, 2006) in paced lines (the definition of the paced line will be given in Section 1.3.4).

The main objective of assembly line balancing is to minimise the sum of differences between the cycle time and individual workloads, and so to minimise the total idle time of the line, by minimising: *(i)* the number of workstations given the cycle time or *(ii)* the cycle time given the number of workstations.

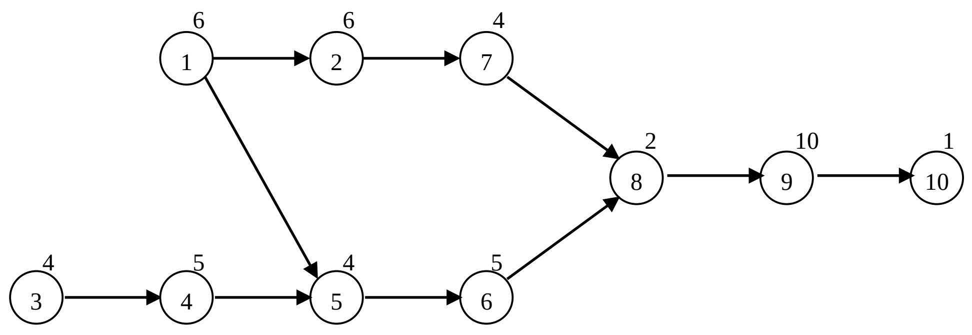


Figure 1-1. Example of a precedence relationships diagram for assembly line balancing problem (Boysen *et al.*, 2007)

An example of the precedence relationships diagram is given in Figure 1-1. In this diagram, nodes represent tasks while arcs symbolise the precedence relationships among these tasks. Due to precedence constraints, for example, task 5 can only be performed after tasks 1, 4 (*direct predecessors*), and 3 (*indirect predecessors*) are completed. Numbers given over nodes represent processing times of relevant tasks.

In the assembly line balancing domain, the SALBP is a basic and special case of the big problem family called *general assembly line balancing problem*. The major assumption of the SALBP is that there is only one product model assembled on a single straight assembly line at a constant production rate. The product model demand and task processing times are also known and deterministic. Other assumptions are (Baybars, 1986b):

- All tasks must be processed and each task must be assigned to exactly one workstation. Splitting tasks between two or more workstations is not allowed.
- All stations are equipped and manned to process any task.
- Task times are independent of the workstation at which they are performed and of the preceding tasks.
- All input parameters are known exactly. The cycle time of the line is given and fixed (for SALBP-I), or the number of workstations is given and fixed (for SALBP-II).

If these assumptions are relaxed or further constraints (*i.e.* positional constraints, zoning constraints) and features (*i.e.* parallel workstations, product model variations, stochastic times) are added to the SALBP, the problem turns into a more complicated structure

(Baybars, 1986b, Bautista and Pereira, 2007). So, all other problem types belonging to the general assembly line balancing problem (*e.g.* mixed-model assembly line balancing problem, parallel assembly line balancing problem, two-sided assembly line balancing problem *etc.*) are more sophisticated versions of the SALBP problem. These problem types will be explained in the following sub-sections in details.

To give an insight about a SALBP formulation, an integer programming model is presented as follows (Bowman, 1960, White, 1961):

$$min \sum_{k=1}^{K} z_k. \tag{1.1}$$

$$\sum_{k=1}^{K} x_{ik} = 1; \quad \forall i = 1, \dots, T. \tag{1.2}$$

$$\sum_{i=1}^{T} pt_i \cdot x_{ik} \leq C \cdot z_k; \quad \forall k = 1, \dots, K. \tag{1.3}$$

$$\sum_{k=1}^{K} k \cdot x_{ik} - \sum_{k=1}^{K} k \cdot x_{jk} \leq 0; \quad \forall i = 1, \dots, T; \quad j \in Suc_i. \tag{1.4}$$

$$x_{ik}, z_k \in \{0,1\}. \tag{1.5}$$

The objective function given in Equation (1.1) minimises the total number of workstations, where $z_k$ gets the value of 1 if and only if at least one of the tasks is assigned to workstation $k$ and $x_{ik}$ gets 1 if and only if task $i$ is assigned to workstation $k$. Constraint (1.2) ensures that each task is assigned to exactly one workstation while Constraint (1.3), which is called capacity constraint, guarantees that the workload of a workstation does not exceed the cycle time determined. The precedence relationship constraints are handled by Constraint (1.4) which ensures that no successor of a task is assigned to an earlier station than that particular task ($Suc_i$ denotes the set of successors of task $i$). Constraint (1.5) defines the domain of the decision variables.

SALBPs can be divided into four different types according to the goal carried on (Boysen *et al.*, 2008):

- Type-I: Minimises the number of workstations for a given cycle time.
- Type-II: Minimises the cycle time (or maximises the production rate) for a given number of workstations.

- Type-E: Maximises the line efficiency (minimises both number of workstations and cycle time simultaneously).
- Type-F: Looks for a feasible solution when the number of workstations and cycle time are given.

Figure 1-2 depicts the objective function based classification scheme for SALBPs.

| | **Cycle Time** | |
|---|---|---|
| **Number of stations** | *Given* | *Minimise* |
| *Given* | SALBP-F | SALBP-II |
| *Minimise* | SALBP-I | SALBP-E |

Figure 1-2: Classification of SALBPs, adapted from (Boysen *et al.*, 2008)

Some different problem types of assembly line balancing problems are briefly given in this subsection. Additionally, surveys of Ghosh and Gagnon (1989), Erel and Sarin (1998), Becker and Scholl (2006), Scholl and Becker (2006), Battini *et al.* (2007), Boysen *et al.* (2007), Boysen *et al.* (2008) and Rashid *et al.* (2012) can be investigated for classification and comprehensive review of the literature on different types of assembly line balancing problems and solution methods.

### 1.3.1. Product variations

Assembly lines can be classified considering the variety of products assembled on the line: single-model assembly lines, mixed-model assembly lines, and multi-model assembly lines (see Figure 1-3).

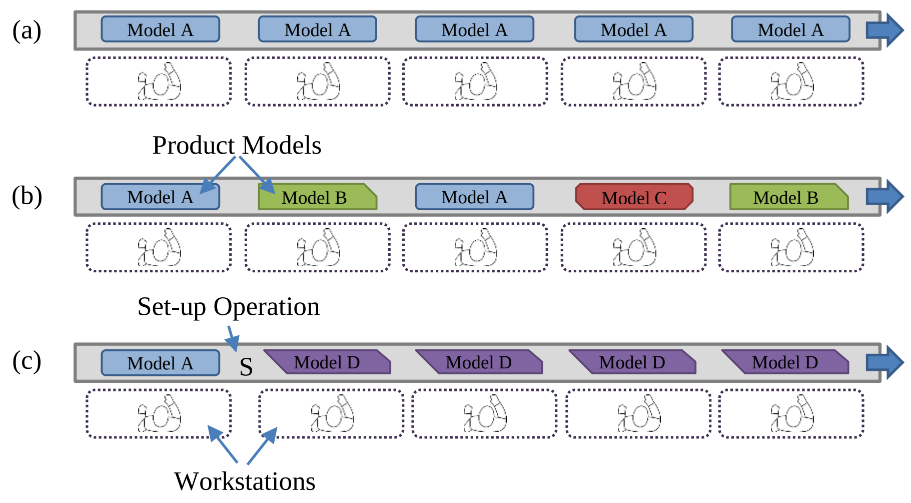


Figure 1-3. (a) Single-model, (b) mixed-model and (c) multi-model assembly lines, adapted from (Becker and Scholl, 2006)

To explain briefly (Rekiek *et al.*, 2002, Rekiek and Delchambre, 2006, Boysen *et al.*, 2008, Hamzadayi and Yildiz, 2012):

- Single-model assembly lines are used to assemble a single homogenous product in large quantities.
- Mixed-model assembly lines are used to assemble a set of different product models (or product variants) simultaneously in an intermixed sequence.
- Multi-model assembly lines are used to assemble batches of similar product models with intermediate setup operations, where there are significant differences between the product models (or product variants) produced.

Assembly lines were used for high-volume production of a single (homogenous) commodity in its traditional form. SALBP, the extensively studied form of line balancing problems, assumes the single-model production and was considered by a vast number of publications, such as Baybars (1986a), Saltzman and Baybars (1987), Hoffmann (1992), Rubinovitz and Levitin (1995), Klein and Scholl (1996), Sprecher (1999), Peeters and Degraeve (2006), Gökçen *et al.* (2005), Zhang *et al.* (2007), Liu *et al.* (2008) and Nourmohammadi and Zandieh (2011).

However, with the change of global market, companies converted single-model lines into mixed-model lines in order to provide diversity and meet customised customer demands on time in an intelligent way (Zhang and Sharifi, 2007). Due to the complexity of the problem, heuristic procedures were preferred rather than exact algorithms to solve mixed-model problems, see for example Sparling and Miltenburg (1998), Dong *et al.* (2002), Vilarinho and Simaria (2002), McMullen and Tarasewich (2003), Mendes *et al.* (2005), Su and Lu (2007), Zhang *et al.* (2008), Hwang and Katayama (2009), Ozcan and Toklu (2009a), Akgunduz and Tunali (2010), Liao *et al.* (2010), Zhang and Gen (2011), Akpinar and Bayhan (2011), Hamzadayi and Yildiz (2012) and Chutima and Chimklai (2012).

A set-up operation is required between two product models if there are big differences between these two product models, for which multi-model lines are constructed. An advantage of mixed-model lines over multi-model lines is that the setup process is not required between product model changes, as can be comprehended from the figure. Multi-model lines are used rarely since they require set-up times between passes from one product model to another. They have been studied by few researchers such as Berger *et al.* (1992), Pastor *et al.* (2002) and Eryuruk *et al.* (2008, 2011).

### 1.3.2. Layout type variations

Another classification criterion for assembly lines is the layout type. In traditional form, workstations are designed serially (in a straight form) and are connected together with a linear conveyor belt (see Figure 1-4). The conveyor belt moves parts from the upstream workstation to its downstream workstation, each of which is located next to each other. The main objectives of balancing straight assembly lines are generally improving line efficiency (or reducing cost) by minimising number of workstations, and maximising workload smoothness between workstations. Although different objectives can be pursued in line balancing problems as it has been extracted in Section 1.3, the main aim is to increase the line efficiency and so the throughput rate while minimising production costs.

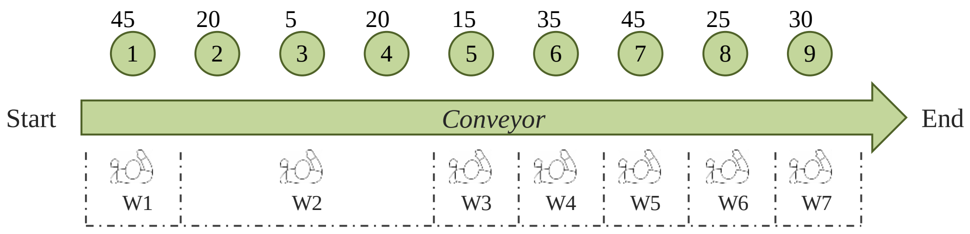


Figure 1-4. Straight assembly line, adapted from (Aase *et al.*, 2004)

In U-shaped lines (or U-lines shortly), the entrance and the exit of the line system are very close to each other and form a ‘U’ shape allowing operators to handle work-pieces both at the front and at the back of the line (Jayaswal and Agarwal, 2014). U-type layouts (a typical representation of which is given in Figure 1-5) were proposed as a new problem derived from the traditional assembly line balancing problem by Miltenburg and Wijngaard (1994). Since then, U-type layouts have been studied and utilised extensively due to the use of just-in-time production principles in last decade (Gökçen *et al.*, 2005). The main benefits of the U-lines compared to the straight lines can be summarised as follows (Monden, 1983, Gökçen *et al.*, 2005, Toksari *et al.*, 2008, Kara *et al.*, 2011):

- reduction in the wasted movement of operators and work-in-process inventory
- improved productivity
- easier implementation of zero-defects campaign
- higher flexibility in workforce planning in the face of changing demand
- improvement in material handling

- multi-skilled operators performing tasks located in different parts of the assembly line
- reduced number of workstations.

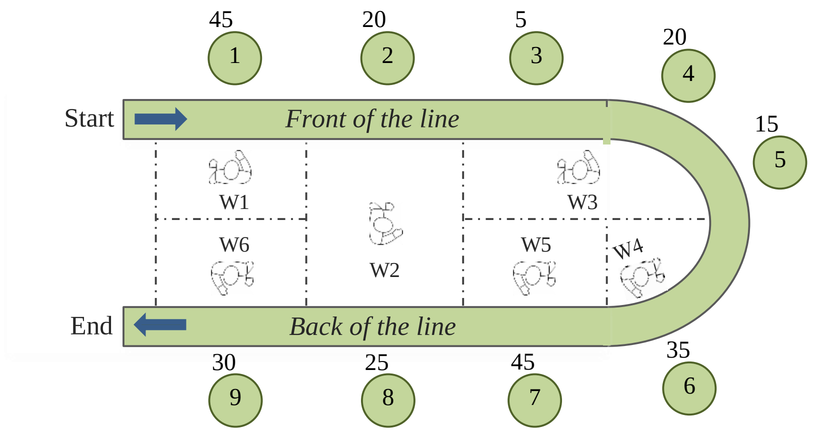


Figure 1-5. U-shaped assembly line, adapted from (Aase *et al*., 2004)

In some cases, other layout types may also be seen in industrial facilities such as C-shaped layout (see Figure 1-6) (Simaria, 2006).

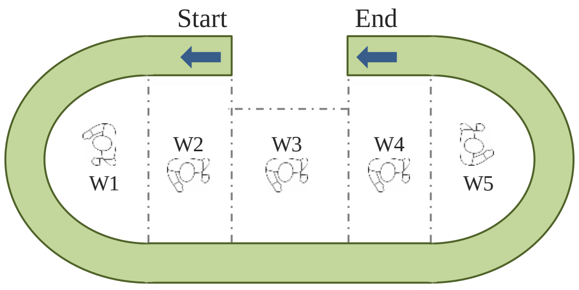


Figure 1-6. C-shaped assembly line, adapted from (Aase *et al*., 2004)

### 1.3.3. Task processing times

Tasks (jobs) are assigned to workstations considering task processing times. Therefore, task processing time is an essential input for assembly line balancing problems. In real world applications, *deterministic times* are suggested to be used if tasks are small and well defined, and expected processing time variability is adequately small. Automated assembly lines enable us to use deterministic task times for allocating tasks to computer controlled machines and robots. However, machine breakdowns may also require considering stochastic times (Rekiek *et al*., 2002).

If tasks are performed by human operators, the variability of the task processing times may be considered by line manager. Processing time may be modified to include the *stochastic time* by adding the stochastic component reflecting the *meantime before failure* and the *mean time to repair* to *operation time*. Sometimes, the adaptation period of workers may be taken into account when constructing a new assembly line. Then, *dynamic times* enable systematic reductions due to learning effects or successive improvements of the production process. Even in case of automated lines, the global duration may vary from the sum of the processing time of all equipment in the group. Since this reason, *hidden times* are employed by Pellichero (1999) to accomplish unexpected delays (Rekiek *et al*., 2002, Simaria, 2006).

### 1.3.4. Paced and un-paced assembly lines

In a *paced* (synchronous) assembly line, the operation time of any workstation cannot exceed the cycle time as a maximum value for each work piece, and production rate (reciprocal of the cycle time) is fixed due to the cycle time constraint. In other words, the production rate is determined by cycle time. However, in *un-paced (asynchronous)* lines, stations are amplified by buffers and workstations can operate at an individual speed while buffers can keep in-process inventories (Malakooti, 1994, Dolgui *et al*., 2002, Becker and Scholl, 2006). In the un-paced lines, there is no fixed time to complete a task and completed parts are delivered either to the following station or buffer upon completion.

### 1.3.5. Parallel workstations

Cycle time is restricted to be equal or larger than maximum task time in a traditional serial line and hence limits the production rate. However, parallel workstations can be constructed to allow a specific task to be performed at more than one workstation simultaneously, thereby reducing the effective task time by the number of times the facility is replicated. Although paralleling may lead to additional cost for performing the tasks that are paralleled at different stations, it is one of the least costly methods of increasing production rate among other alternatives (Pinto *et al*., 1975). Johnson (1983), Pinto *et al.* (1975), Bard (1989), Daganzo and Blumenfeld (1994) and Askin and Zhou (1997) considered *"a trade-off between the incremental tooling/equipment cost of the duplicated workstation and the cost of hiring workers for the original line in order to satisfy the demand"* as a base to decide on creating parallel workstations (Vilarinho and Simaria, 2002).

The use of parallel workstations yields flexible assignment of tasks and requires less cycle time. However, the advantage of the division of labour inherent to assembly lines can be lost if the replication of workstations is not controlled (Simaria, 2006). Figure 1-7 illustrates how to parallel a task.

Please refer to Pinto *et al*. (1981), Bard (1989), McMullen and Fraizer (1998), Simaria and Vilarinho (2001) and Akpinar and Bayhan (2011) for more information on utilisation of parallel workstations. Unlike parallel workstations, parallel lines (which is studied in this research) are constructed to assemble one or several products on different lines with more flexibility (*e.g.* different cycle times). There is no relationship between parallel stations and parallel lines used in this research and detailed information about parallel lines will be given in next chapters.

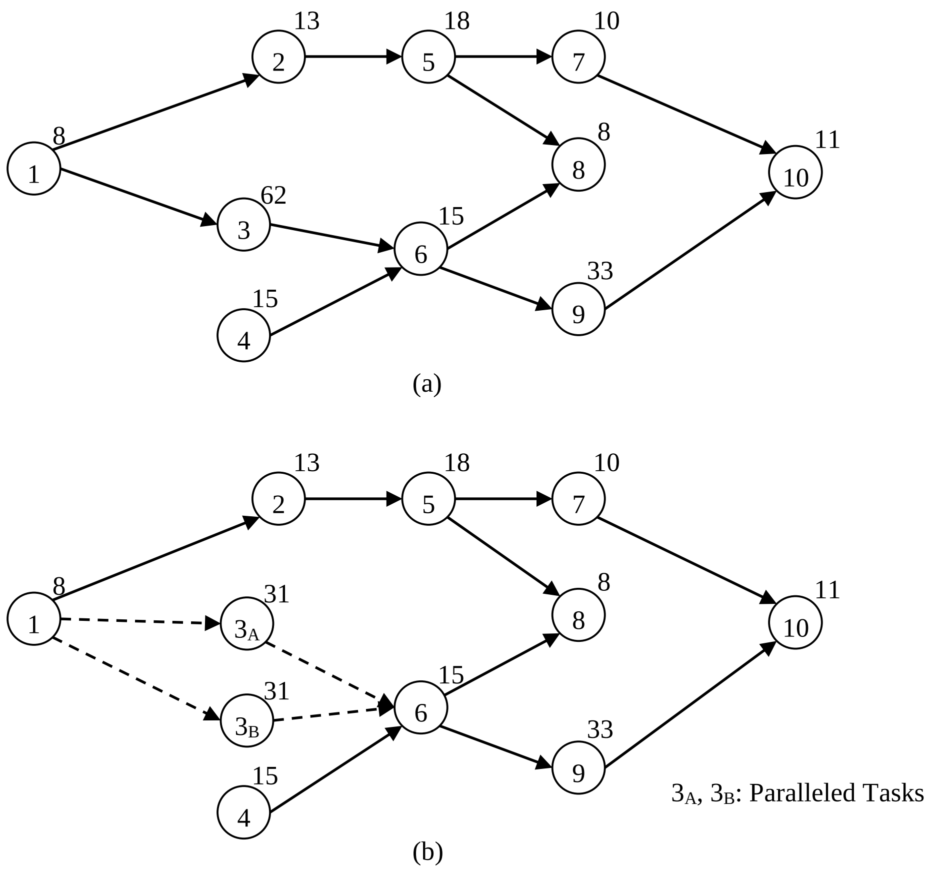


Figure 1-7. Sample precedence diagrams, (a) before paralleling, (b) after task 3 paralleled, adapted from (Pinto *et al*., 1975)

### 1.3.6. Assignment constraints

Several types of assignment constraints may be included in assembly line balancing problem. These constraints may arise from technological circumstances, and special features of tasks, workstations or operators.

In manufacturing systems, some tasks are compelled or prohibited to be performed at the same workstation, called *positive* or *negative zoning constraints*, respectively. There are several reasons why tasks might (or might not) need to be kept together. For example, some tasks that require the vehicle to be turned upside-down may be performed at same workstations to reduce the times the vehicle must be inverted. Therefore, if two different tasks require same equipment or similar working conditions (like temperature, pressure, *etc.*) it is appropriate to assign them to the same workstation (positive zoning). Negative zoning constraints occur if two different tasks must be assigned to different workstations because of technological issues or safety reasons (Simaria, 2006).

*Position related constraints* group tasks according to the position in which they are performed. For example, in a two-sided assembly line that work-pieces have fixed positions and cannot be turned, tasks may require to be performed at both sides of the assembly line simultaneously. These tasks can be called *synchronous tasks*.

If a special equipment is available only at a pre-determined station, *workstation related constraints* are needed to assign tasks which require that particular equipment into that workstation. For example, welding operation must be assigned to the workstation in which welding robot is fixed. Sometimes, some tasks may need a qualified operator to be performed. In this cases, *operator related constraints* are desired (Simaria, 2006).

### 1.3.7. Space constraints

In real world applications, some workstations may require more space to perform some kinds of tasks. Especially in the automotive industry, space may be crucial to balance assembly line with no violation. Since this reason, a new scheme was proposed: the time and space assembly line balancing problem. Some objectives of considering space are (Bautista and Pereira, 2007, Chica *et al*., 2010):

- To provide operators enough space in order to perform their jobs.
- To share common required tools and components between workstations.
- To keep some components in temporary batches, especially in mixed-model assembly lines.
- To redesign an assembly line when a larger or more component-rich product model will be assembled at the same line.

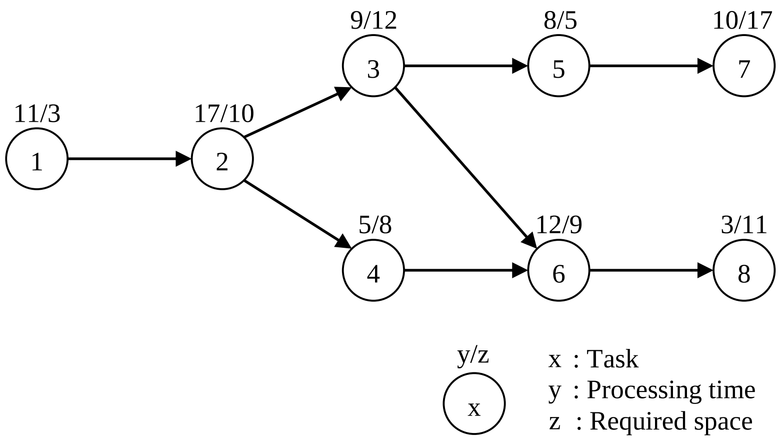


Figure 1-8. Illustration of precedence diagram for time and space constrained assembly line balancing problem, adapted from (Bautista and Pereira, 2007)

An example of precedence diagram for time and space assembly line balancing problem is given in Figure 1-8. Over the top of each task, processing time and required space to perform the task are indicated as numbers.

Recently, time and space constraints are considered simultaneously by few researchers such as Blum *et al.* (2006), Blum *et al.* (2008), Chica *et al.* (2009) and Chica *et al.* (2011).

## 1.4. Research Questions

Manufacturing systems that consist of more than one two-sided assembly line, on which large-sized products are produced, are encountered frequently in practical applications. Different products or similar models of the same product are assembled on each of these parallel two-sided lines. However, there are few studies on balancing parallel two-sided assembly lines in the relevant literature, unlike traditional simple line configurations. Also, flexibility, productivity, immunity against breakdowns and failures, and smoothness of the system are crucial parameters in today’s highly competitive dynamic manufacturing environment and parallel mixed-model assembly lines provide very efficient solution strategies to satisfy these key requirements especially when the capacity of a production system is insufficient. However, the combination of these two line layouts, which constitute a mixed-model parallel two-sided assembly line system, has not been studied before, to the best knowledge of the author.

As a new research project, which aims to model mixed-model parallel two-sided assembly line system and propose efficient solution procedures for solving balancing and sequencing problems of such systems, the research questions (RQs) that shape the idea of this thesis can be released as follows:

**RQ1:** *If two-sided lines are put together in parallel to produce variants of a large-sized product, would this system be improved in terms of line length and/or the total number of workstations compared to independently balanced two-sided lines?*

**RQ2:** *How to model and solve the problems of mixed-model parallel two-sided assembly line systems effectively and how to cope with newly arising and possibly challenging issues in these problems?*

First of all, a comprehensive review of the literature on parallel assembly lines and two-sided assembly lines needs to be accomplished to check whether the parallel two-sided line system has already been utilised for the aim of producing mixed product models. The main characteristics of parallel lines and two-sided lines need to be presented and generic solution methods for such line configurations and their combinations, *e.g.* parallel two-sided lines, need to be explained through illustrations and numerical examples.

Afterwards, the proposed mixed-model parallel two-sided assembly line system may be introduced and formulated mathematically. The ways of not only increasing flexibility but also minimising the total number of workstations (and so the idle times) of the entire line system should be sought. The issues on what kind of constraints, which reflect more realistic conditions in real assembly lines, can be taken into account and the possible key benefits over conventional configurations to be questioned.

The challenging issues of the proposed line system also need a broad investigation. It is highly possible that considering such a complex mixed-model parallel two-sided assembly line configuration will result in new problems that must be dealt with carefully. For example, there will be different product models being produced on different lines. How to tackle different model sequences in multi-line stations (which are constructed between two adjacent lines and considered to be the essential advantage of parallel line systems) and avoid violation of certain constraints (*i.e.* capacity constraint and precedence relationship constraint)? Moreover, when different cycle times are considered for the parallel two-sided lines, product models will change at different times in multi-line stations. So that, dynamic workloads of the multi-line stations may be the most challenging issue to ensure that the capacity is not exceeded and all of the predecessors of a task are completed before it is started.

Instead of building a balancing solution for a given sequence of product models, or sequencing product models where the line balance is given (as commonly applied in the literature), can the balancing and sequencing problems (which are two tightly interrelated problems) be handled simultaneously? In such an environment, what are the benefits and shortcomings of utilising multi-line stations? What are the effects of *(i)* balancing mixed-model parallel two-sided lines together and *(ii)* solving the line balancing problem together with the model sequencing problem on the sought performance measures (namely line length and the number of workstations)? All of these questions and their potential answers shape the core idea of this thesis.

## 1.5. Research Objectives

Although a large number of exact, heuristic, and meta-heuristic algorithms have been proposed for two-sided assembly line balancing problems and parallel assembly line balancing problems separately; the literature on the combination of these two problems, namely parallel two-sided assembly line balancing problem, is extremely limited. Moreover, there is no study which takes into account product model variations for parallel two-sided assembly lines.

Based on this motivation, the main aim of this thesis is *to introduce and model a new line configuration, mixed-model parallel two-sided assembly line system, and to develop powerful techniques for tackling two tightly interrelated problems, which are line balancing and model sequencing, simultaneously under more realistic conditions*.

The overall objectives of this PhD research could be drawn as follows to respond to the research questions defined in the previous subsection:

- *To establish the evidence for the originality of this research with a comprehensive review of the literature on parallel assembly lines, two-sided assembly lines, mixed-model assembly lines and their combinations. In doing so, to explain the main characteristics and generic solution approaches of these lines.*
- *As a new assembly line configuration, to describe the mixed-model parallel two-sided assembly line system and mathematically formulate the balancing and*

*sequencing problem for this system. To define the key benefits as well as the challenging issues of the introduced system and elaborately illustrate them through numerical examples.*

- *To identify the key requirements for implementing an efficient mixed-model parallel two-sided assembly line system in terms of line length and the number of workstations and to develop efficient solution methods. The methods developed should handle all of these requirements and challenging issues arising due to the complex nature of the problem studied.*
- *To express the solution building procedures of the developed methods in details and to demonstrate their applications on illustrative examples derived from the literature.*
- *To solve miscellaneous test cases, which are obtained through combining existing test problems in the literature, in order to observe the advantages of the proposed manufacturing line configuration and the performances of the proposed algorithms.*
- *To establish a well-designed set of statistical tests for the purpose of measuring the significances of potential benefits to be gained within the scope of this thesis.*
- *Finally, to identify new research areas contextualised with this research and to offer new topics that could be of interest for further researches over the problem studied in this thesis.*

## 1.6. Thesis Structure

The thesis is divided in nine chapters in which research questions will be addressed accordingly. Figure 1-9 illustrates the flow and connection of the content in each chapter. This chapter (Chapter 1) has already introduced the reader to the main area of study. Definitions of the key concepts in assembly line balancing domain have been provided for the common understanding of the subjects being considered. After a broad fundamental information, the research questions and objectives were identified.

Following the introduction, parallel assembly line balancing problems and two-sided assembly line balancing problems are examined separately in Chapter 2, to construct a more distinguished review of the literature. As there is no published work on mixed-model parallel two-sided assembly line balancing problems; the survey is conveyed based on its sub-problems. The typical configurations of the lines are also exhibited in the chapter with their generic solution approaches.

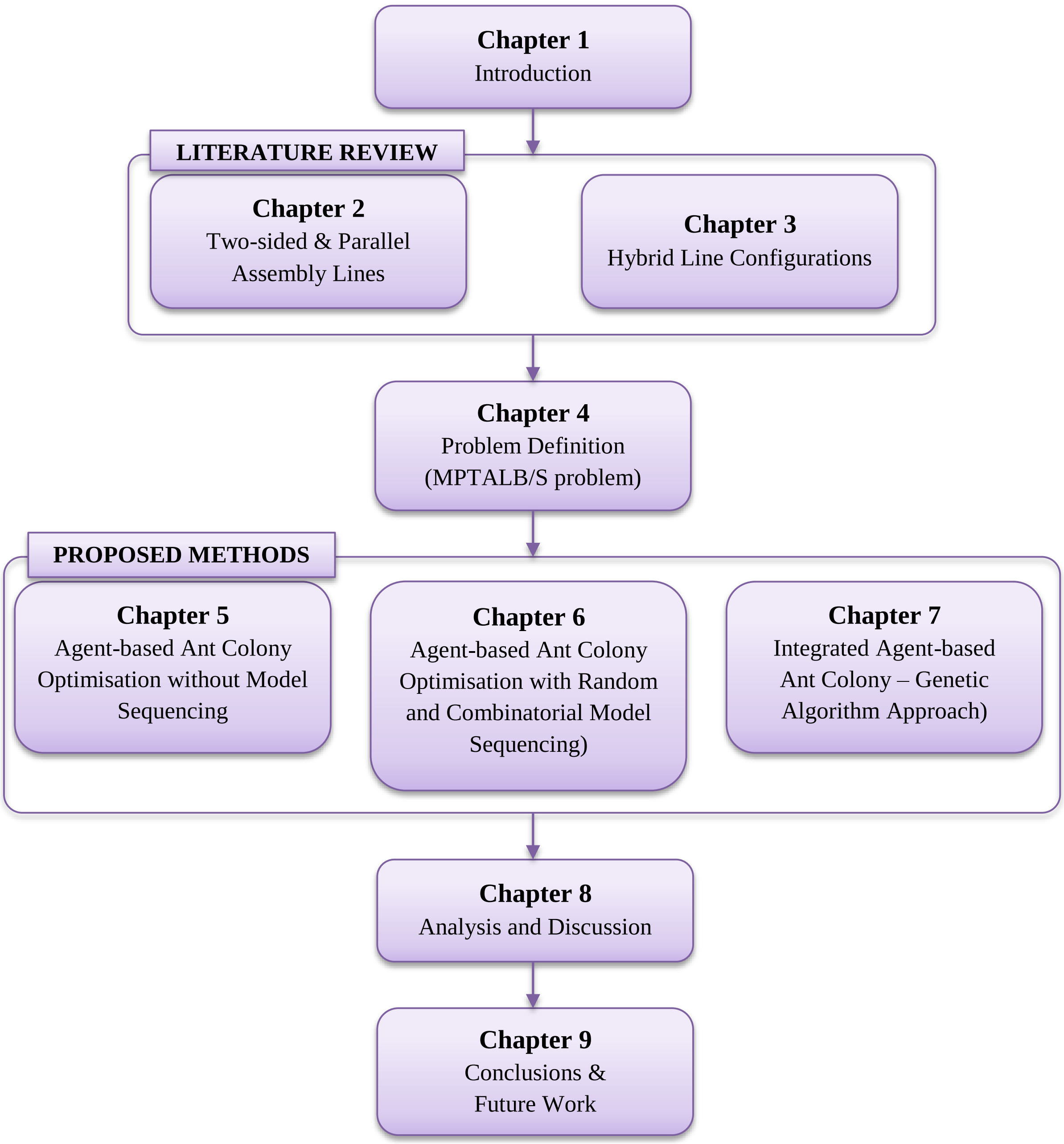

Chapter 1
Introduction
LITERATURE REVIEW
Chapter 2
Two-sided & Parallel Assembly Lines
Chapter 3
Hybrid Line Configurations
Chapter 4
Problem Definition (MPTALB/S problem)
PROPOSED METHODS
Chapter 5
Agent-based Ant Colony Optimisation without Model Sequencing
Chapter 6
Agent-based Ant Colony Optimisation with Random and Combinatorial Model Sequencing)
Chapter 7
Integrated Agent-based Ant Colony – Genetic Algorithm Approach)
Chapter 8
Analysis and Discussion
Chapter 9
Conclusions & Future Work

Figure 1-9. Thesis structure

Chapter 3 presents the review of the literature on the combinations (called hybrid line configurations) of the sub-problems exhibited in the previous chapter after a brief description of such systems. Detailed surveys are also carried out about ant colony optimisation (ACO shortly) techniques and multi-agent systems, which have been developed so far to solve numerous kinds of assembly line balancing problems in the literature. Previous researches are explored and summarised using tables to provide an easy and holistic overview for readers.

Following a brief research on existing simultaneous balancing and model sequencing studies to exhibit the current situation in the literature, Chapter 4 comprehensively defines the MPTALB/S problem with a mathematical model. Problem specific constraints are explained with the assumptions, and benefits of using such systems are expressed. The necessity of considering model sequencing problems together with line balancing problems are also examined by demonstrating various possible production cycles in schematic diagrams.

Chapter 5 describes the proposed generalised solution approach, an agent-based ACO algorithm (called ABACO shortly), for the mixed-model parallel two-sided assembly line balancing problem. To decide whether a task is available or not, the algorithm considers the maximum processing time of this task amongst all different product models. Therefore, the results obtained here satisfy any product model sequence on the lines, which leads to improved flexibility by allowing demand changes with no re-balancing requirement. The running mechanism of the algorithm is illustrated through explanatory figures and an example is given for the interpretation. Experimental tests are carried out, and parallel lines are balanced separately and then together to show the advantage of balancing lines together thanks to the utilisation of multi-line stations. The performance of the proposed ABACO algorithm is also compared to heuristics commonly used in the literature.

Chapter 6 exhibits the improved version of the algorithm proposed in Chapter 5, to obtain more efficient designs and cope with the product model changes in multi-line stations. A model sequencing mechanism is integrated to ABACO algorithm (and so ABACO/S is obtained, where 'S' refers to sequencing mechanism) and the available tasks to be assigned to the workstations are determined through a more complicated control mechanism. For this aim, all possible product model combinations of the lines are permuted and workloads of workstations are recorded along with the earliest starting times of the tasks, for every possible production cycle. The procedures of developed ABACO/S algorithm are depicted using figures and examples to provide a better understanding. Also, the performance of ABACO/S is tested through computational experiments and comparisons are made with heuristic solutions of the addressed problem.

In Chapter 7, a new genetic algorithm (GA) based model sequencing procedure is integrated to ABACO/S algorithm (and so ABACO/S-GA is obtained) to improve the robustness and efficiency of the developed algorithm. The evolution of the chromosomes and the solution building mechanism of the algorithm are exhibited through a numerical example. Moreover, test problems are solved and the performance of the improved method is compared with that of the method proposed in the previous chapter.

Chapter 8 is devoted to analysing the results obtained through test cases using the algorithms developed in Chapter 5, Chapter 6, and Chapter 7. A set of paired sample t-tests is performed to statistically analyse the results, and the outputs are discussed from a critical point of view.

Chapter 9 concludes the thesis by referring back to the research questions. Research contributions are highlighted and implications, limitations and future research are identified.

## 1.7. Chapter Summary

In this chapter, an introduction to this PhD research was given with an historical view and the main characteristics of the assembly line balancing problems, followed by the research questions, research objectives and the general outline of the thesis to describe the scope of this study. Some basic definitions, special features and miscellaneous constraints of the assembly line balancing problems were also provided in this chapter.

# 2

# THE REVIEW OF THE LITERATURE ON TWO-SIDED LINES & PARALLEL LINES

**Contents**

## 2.1. Chapter Introduction

In this chapter and the next chapter, review of the relevant literature is given pertaining the MPTALB/S problem, which constitutes the main topic of this thesis. Since there is no published study regarding mixed-model parallel two-sided assembly lines, the literature survey is done on its sub-problems. Therefore, the aim of this chapter is to present a detailed review of the literature on two sub-problems of the MPTALB/S problem, two-sided assembly line balancing problem (in Section 2.2) and parallel assembly line balancing problem (in Section 2.3). Prior to the survey, brief definitions on both problem types are also given along with the generic solution approaches to tackling the related problems. This is to provide a better understanding about the state of the art and to exhibit the current situation which will lead to identifying the gap in the literature. Then, surveys are given on existing procedures for two-sided assembly line balancing problems, and parallel assembly line balancing problems. Summary tables are also presented to provide an overview of the literature with key features of the solution approaches and tackled problems. The chapter is concluded with a summary in Section 2.4.

## 2.2. Two-sided Assembly Line Balancing Problem

Assembly lines can be divided into two different groups based on product characteristics and some technical requirements: *(i)* one-sided assembly lines, and *(ii)* two-sided assembly lines. While only one restricted side (either Left-L or Right-R side) is used in one-sided assembly lines, both left and right sides are used in two-sided assembly lines. Two-sided assembly lines are usually constructed to produce large-sized high volume products such as buses, trucks, automobiles, and some domestic products. Two directly facing workstations, called *mated-station*, are allocated at each working position (Chutima and Chimklai, 2012).

As mentioned, two-sided assembly lines are chiefly used in the production of large-sized products and the workers at each pair of opposite stations work in parallel on different tasks but on the same individual item (Bartholdi, 1993). The main difference of this kind of systems is that some tasks are required to be performed on a specific side of the line (L or R) while others may be performed on either side of the line (Either-E). Two-sided assembly lines are more practical for large products like trucks than for small ones like electrical drills because of the *interference* phenomenon (Kim *et al.*, 2000c, Lee *et al.*, 2001). Interference was already explained briefly in Section 1.2 and will be explained in detail in Chapter 4.

Figure 2-1b illustrates the configuration of a typical two-sided assembly line problem for which the input data is given in Figure 2-1a. In Figure 2-1a, the numbers in the nodes represent the tasks. The labels above the nodes denote completion time and preferred operation direction of tasks, respectively. The directed arrows between nodes imply the precedence relationships between tasks, *e.g.* tasks 1 and 2 immediately precede task 3. In Figure 2-1b, a pair of two directly facing workstations, *e.g.* WS1 and WS2, is called *mated-station*, and one of them calls the other as a *companion*. WS1 and WS2 perform different tasks in parallel but on the same individual item.

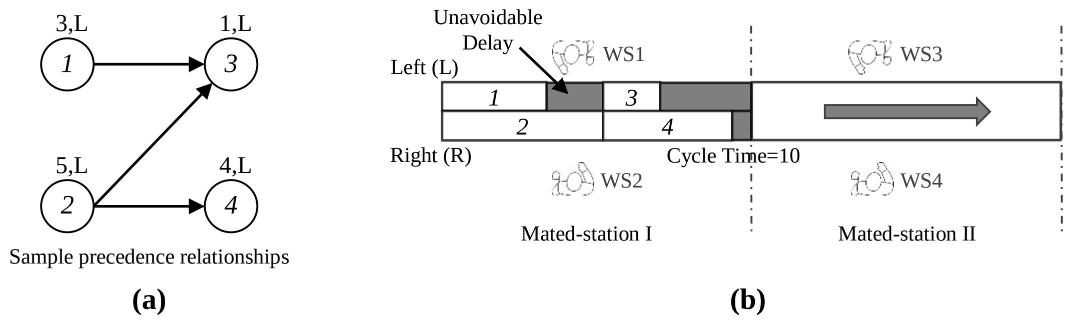


Figure 2-1. Configuration of a two-sided assembly line, adapted from (Chutima and Chimklai, 2012)

Two-sided lines can provide many substantial advantages over one-sided assembly lines. These include (Lee *et al.*, 2001):

- shorter line length
- reduced throughput time
- lower cost of tools and fixtures through shared equipment
- reduced material handling and operator’s movement
- less set-up time.

Karp (1972) proved that two-sided assembly line balancing problem is NP-hard. As aforementioned, the tricky part of balancing two-sided assembly lines is to determine which task will be performed at which workstation on which side of the line and in which sequence. This complexity makes solving two-sided assembly line balancing problems more difficult than balancing traditional lines. This is the truth lies behind plenty number of studies proposing solution methods to balance conventional one-sided assembly lines while the number of studies dealing with two-sided lines is very scarce (Ozbakir and Tapkan, 2010).

Figure 2-2 depicts the precedence relationships diagram of a typical two-sided assembly line balancing problem with task times and preferred operation directions (sides). Figure 2-3a and Figure 2-3b illustrates typical balancing solutions for one-sided and two-sided assembly lines, respectively, where the tasks are represented as numbers at their relevant positions inside the bars while starting and finishing times of each task are shown alongside the bars. Please note that operation directions are ignored (all tasks are assumed right-sided) for one-sided balancing solution. The shaded bars in the figures indicate either idle time towards the end of the cycle time (*e.g.* see workstation I in Figure 2-3b) or unavoidable delay because of the interference (*e.g.* see workstation II in Figure 2-3b) (Ozcan and Toklu, 2009b).

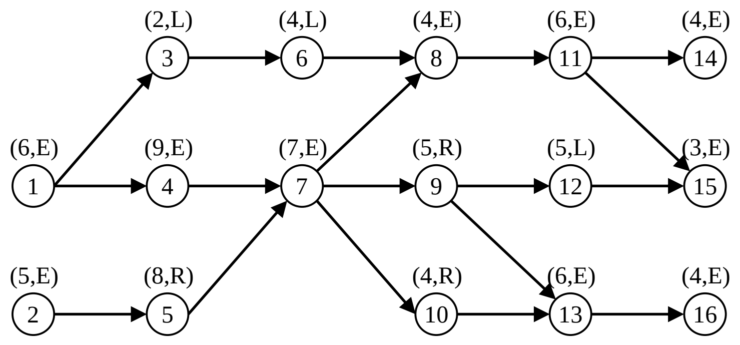


Figure 2-2. The precedence relationships diagram, task times and preferred operation directions of the 16-task problem, adapted from (Ozcan and Toklu, 2009b)

The general approach to solving these problems in the literature is based on listing available tasks considering constraints and other special circumstances (*i.e.* operation direction, availability of current station's capacity) and selecting a task to assign to current position of a workstation. While the selection of tasks from the available tasks list is conducted using the emergent intelligence of the algorithm in swarm intelligence based meta-heuristic approaches, *i.e.* foraging of ants (see Baykasoglu and Dereli (2008), and Simaria and Vilarinho (2009)), and foraging of bees (see Ozbakir and Tapkan (2010, 2011)); some other heuristic procedures may also be used for this aim. The solution procedure of the ant colony based heuristic proposed by Baykasoglu and Dereli (2008) is given in Figure 2-4.

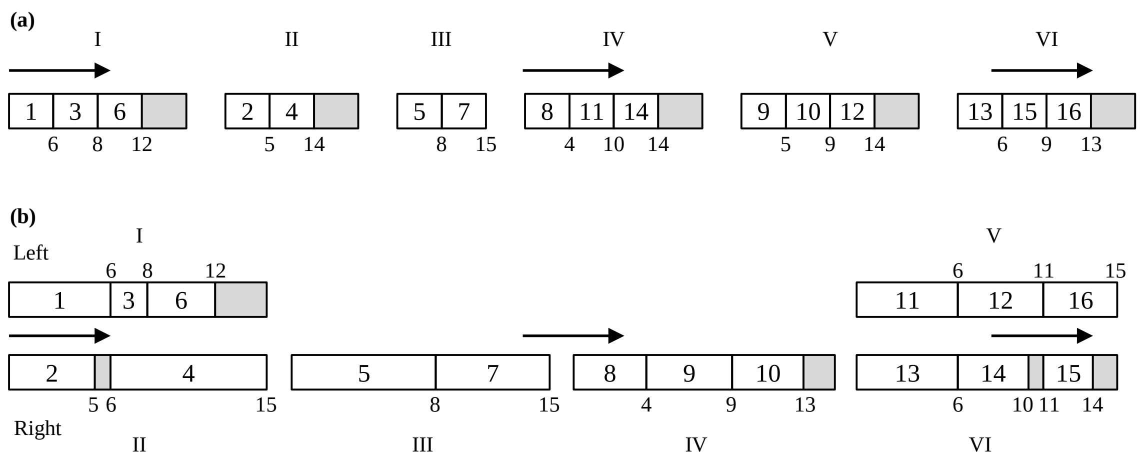


Figure 2-3. Task assignment solutions of (a) one-sided assembly line and (b) two-sided assembly line with a cycle time of 15, adapted from (Ozcan and Toklu, 2009b)

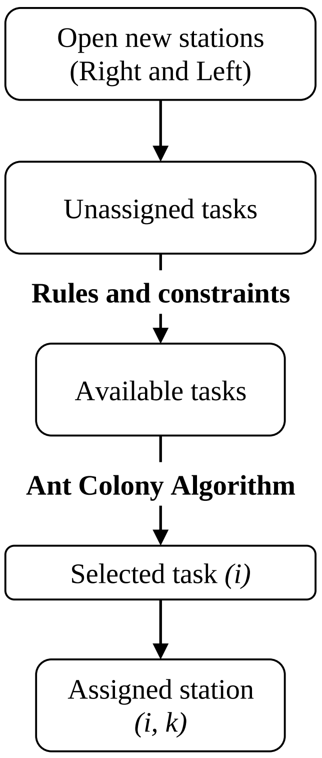


Figure 2-4. The steps of the algorithm used in Baykasoglu and Dereli (2008)

In some population-based approaches and meta-heuristics, such as the GA and simulated annealing algorithm, an initial solution is generated using the task assigning procedure explained above and neighbourhood solutions are sought based on special characteristics of the related algorithm. Figure 2-5 represents an example for task-based chromosome, from a GA-based approach developed by Taha *et al*. (2011).

Each individual (called chromosome or task sequence) in the population consists of a complete sequence of tasks that represent a feasible solution. Then, individuals are evaluated by assigning tasks into workstations according to a station oriented procedure which starts with the first mated-station (left and right sides). The tasks are assigned to the current station considering the sequence of tasks in the chromosome and next mated-station is opened when the left and right sides of the current station are loaded as much as possible (see Taha *et al*. (2011) and Ozcan and Toklu (2009c)).

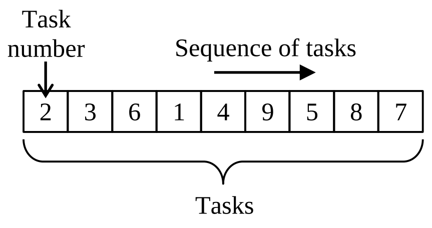


Figure 2-5. An example of task-based chromosome used in Taha *et al*. (2011)

In some cases, priority values are also integrated into the solution procedures to help the algorithm developed not only generate initial populations but also find more efficient solutions. Considering the operation direction preference, the first task with the highest priority value in a set of assignable tasks is allocated to the first mated-station, and the list of assignable tasks is updated. This cycle continues until all tasks are assigned (Ozcan and Toklu, 2009c). Different type of priority rules exist in the literature, *i.e.* ranked positional weight method (Kim *et al*., 2000c); shortest processing time, longest processing time, minimum total number of successor tasks, maximum total number of successor tasks, and minimum total processing time of successor tasks (Ozbakir and Tapkan, 2011). Please refer to Boctor (1995), Ponnambalam (2000), Scholl and Becker (2006) and Baykasoglu (2006) for more details on priority rules.

Table 2-1 presents detailed summary of the literature review on two-sided assembly line balancing problems since 1993. As can be seen from the table, there exist an increasing number of studies from 2008. However, none of them addressed the mixed-model parallel two-sided assembly line balancing topic.

Bartholdi (1993) described two-sided assembly line balancing problem with theoretical properties of two-sided lines for the first time in the literature; and developed a computer programme which embodies a balancing algorithm that emphasizes speed over accuracy for the interactive rapid refinement of solutions.

Table 2-1. Detailed summary of the literature review on two-sided assembly line balancing problems since 1993

| **Research** | **Method / solution approach** | **Main object. (min)** | | | **Additional constraints/features** |
|---|---|---|---|---|---|
| | | **NS** | **C** | **NM** | |
| Bartholdi (1993) | Modified first-fit heuristic based computer programme | ● | | | |
| Kim *et al.* (2000c) | GA with critically ranked positional weight method | ● | | | positional constraints |
| Lee *et al.* (2001) | Group assignment procedure | ● | ● | | work-relatedness and work slackness |
| Hu *et al.* (2008) | Station oriented enumerative algorithm integrated with Hoffmann heuristic | ● | | | |
| Baykasoglu and Dereli (2008) | Ant colony optimisation based algorithm | ● | | | zoning constraints |
| Wu *et al.* (2008) | Branch and bound algorithm | ● | | | |
| Kim *et al.* (2009) | GA | | ● | | |
| Simaria and Vilarinho (2009) | Ant colony optimisation algorithm | ● | | | synchronous tasks, zoning constraints, *mixed-models* |
| Ozcan and Toklu (2009a) | Simulated annealing algorithm | | | ● | minimising weighted smoothness index, synchronous tasks, positional constraints, zoning constraints, *mixed-models* |
| Ozcan and Toklu (2009b) | Goal and fuzzy goal programming models | | ● | ● | multiple criteria, zoning constraints |
| Ozcan and Toklu (2009c) | Tabu search algorithm | ● | | | smoothness index |
| Hu *et al.* (2010) | Branch and bound algorithm | ● | | | line length considered |
| Ozbakir and Tapkan (2010) | Bees algorithm | | | ● | synchronous tasks |
| Ozcan and Toklu (2010) | MIP model and COMSOAL based heuristic (2-COMSOAL/S) | ● | | ● | considers sequence dependent setup times |
| Ozcan *et al.* (2010b) | Tabu search algorithm | | | | *parallel lines* |
| Ozcan (2010) | Simulated annealing algorithm | ● | | ● | stochastic task times |
| Yegul *et al.* (2010) | Multi-pass random assignment algorithm | ● | | | two-sided U-shaped line |
| Ozbakir and Tapkan (2011) | Modified bees algorithm | | | ● | well balanced (smooth) solution |
| Taha *et al.* (2011) | GA with hybrid crossover and a modified scramble mutation operators | ● | | ● | |
| Chutima and Chimklai (2012) | Particle swarm optimisation with negative knowledge | ● | | ● | work-relatedness and workload smoothness |
| Purnomo *et al.* (2013) | GA and iterative first fit rule | | ● | | zoning, distance, resource, and positional constraints, workload balance between workstations |
| Rabbani *et al.* (2012) | GA-based heuristic and mixed integer programming formulation | ● | ● | | multiple U-shaped lines, *mixed-model*, synchronous tasks, zoning constraints |

NS: Number of workstations, C: Cycle time, NM: Number of mated-stations, MIP: Mixed integer programming, GA: Genetic algorithm, ACO: Ant colony optimisation, COMSOAL: Computer method of sequencing operations for assembly lines (Arcus, 1966), RPWM: Ranked positional weight method (Helgeson and Birnie, 1961)

Bartholdi (1993) found that the proposed approach is more useful than the typical optimality-seeking techniques to solve real assembly line balancing problem. Two main functional units, the product/task editor and the line/station editor, were embodied in the computer programme. The product editor and the line editor were responsible to manage the work standards of a product, and layout of the assembly line, respectively.

Unlike many other standard line-balancing algorithms, this algorithm considered positional and zoning constraints (that is, restrictions on where tasks should be placed) while minimising the number of stations for a given cycle time. Their method, called first fit, is a little different and more complicated than the familiar version since it looks for opportunities to schedule a task during idle periods in the midst of the schedule. The assignment procedure can be expressed in summary as follows:

Repeat until all tasks are assigned:

- *Find the next task in the list, all of whose predecessors have been assigned.*
- *Find the last position i to which a predecessor has been assigned, and for that position find the latest finish time t of any predecessor.*
- *Beginning at position i and time t, find the first place in a schedule into which the task can be inserted, observing the operation side restriction of the task and the work limits at the workstations.*

Some other heuristics were developed to solve two-sided assembly line balancing problem as well. Lee *et al.* (2001) designed an efficient group assignment procedure, which assigns a group of tasks at a time rather than a unit task, for two-sided assembly line balancing problems. The main idea of the procedure is to allocate directly connected tasks in the precedence diagram to the same workstation rather than splitting them into several workstations. They carried out experiments to demonstrate the superiority of the proposed method. Problem specific work-relatedness and work slackness were employed as decision criteria. Although work-relatedness and work slackness may conflict with traditional goals (the number of stations and cycle time) sometimes, the developed procedure could produce good solutions without causing much damage to the traditional performance measures.

They have also considered work slackness of tasks to avoid delays that can be sourced by a stalled preceding task. Predefined amount of time is inserted between two related tasks in the precedence diagram. First, tasks are grouped considering the operation directions (L, R, and E) of tasks and operation directions are determined for each group.

Afterwards, available tasks are selected in terms of cycle time and precedence constraints. This loop is repeated until all the tasks are assigned. This method may be more useful when the work-relatedness and work slackness are critical. Because, the measure of the proposed method is not directly related to traditional measures such as degrading the number of stations, cycle time, or balance delay.

Hu *et al.* (2008) presented a station oriented enumerative algorithm for two-sided assembly line balancing problem. They defined a transfer function and combined it with the precedence relations to compute the earliest and the latest starting time of tasks. Hoffman heuristic was also integrated to the algorithm in order to develop a system that can solve the two-sided assembly line balancing problem. A group of tasks that satisfy the direction, the precedence and the cycle time constraints is determined and assigned to each position, rather than workstation to minimise the positional idle time. The optimal subset is determined by enumerating all feasible assignments. In the procedure, tasks are assigned starting from the left station to the right station of the position and then the assignment dissatisfying the precedence constraint is removed by checking. So, this may cause unnecessary computing time.

A new hybrid design, a combination of two-sided line and U-shaped line, was introduced by Yegul *et al.* (2010). This new design takes the advantages of both designs at the same time. One part of the U-shaped line was designed as allowing station crossovers, and another part of the line is arranged like a traditional straight line. They applied a multi-pass random assignment algorithm to minimise required number of stations. This algorithm was very similar to COMSOAL (originally developed by Arcus (1966)), however it enables task sequencing, which plays a key role in two-sided assembly line balancing problems, to prevent interference of tasks at different sides. Although the new design has a potential to balance lines more effectively, it needs more space due to the shape of the line.

Ozcan and Toklu (2010) addressed sequence dependent setup times while solving the two-sided assembly line balancing problem with a multi-objective mixed integer programming model. The main reason of incorporating sequence dependent setup times was to avoid detentions of a line due to any setup requirement in a station between two tasks. A new COMSOAL based heuristic approach was developed for especially large-sized problems. The main objective of the algorithm was to minimise the number of mated-stations. It also minimised the number of stations (*i.e.* the number of operators) as a secondary objective for a given cycle time.

Exact solution methods were developed by Wu *et al*. (2008) and Hu *et al*. (2010). Wu *et al*. (2008) implemented the first exact method, a branch and bound algorithm, to solve two-sided assembly line balancing problem. They formulated two-sided assembly line balancing problem and constructed an enumeration tree inspired by task-oriented branch and bound algorithm. First, nodes are built and then composed by task number and its side. E-type tasks (tasks that can be assigned to either side) are assigned to left side and right side respectively. After these steps, a tree can be enumerated to prove all possible and feasible solutions. Hu *et al*. (2010) developed a new branch and bound algorithm which relaxes the two-sided assembly line to a one-sided assembly line. They extended dominance rules and reduction rules for the one-sided assembly line balancing problem into a station oriented assignment procedure for the two-sided assembly line balancing problem. Dominance rules, which compare the partial solution to find the dominance relationships, are *(i)* maximum load rule, *(ii)* Jackson dominance rule, *(iii)* feasible set dominance rule, and *(iv)* position ordering rule. Modified reduction rules that used to simplify the problem are *(i)* task time incrementing rule, and *(ii)* prefixing. Please refer to related study for more details.

To address the precise and imprecise goals, Ozcan and Toklu (2009b) developed a preemptive goal programming model, and a fuzzy goal programming model, respectively. The main objective of this first multiple-criteria decision-making approach in the literature was to minimise the number of mated-stations (referred to as line length) and minimise the number of stations. They have also considered zoning constraints. Three objectives; namely, minimisation of the number of mated-stations, cycle time, and the minimisation of the number of tasks assigned to each station, were considered. The model has a flexibility that allows users to improve the value of goals based on the preferences of a decision maker.

Meta-heuristics have also been presented to address two-sided assembly line balancing problem. GAs are flexible in dealing with different objectives, particularly in combinatorial optimisation problems and constraints, and have been proven on a wide variety of line balancing problems (Goldberg, 1989). Kim *et al*. (2000c) developed a new GA with genetic encoding - decoding scheme and problem specific genetic operators, to solve two-sided assembly line balancing problem. Proposed GA also had a strength to solve various types of assembly line balancing problems. The main objective of the algorithm was to find the minimum number of workstations for a given cycle time (predetermined by the line manager) while satisfying positional constraints originated from facility layout.

For decoding, a *weight* was used similar to ranked positional weight (sum of the processing times of a task and all of its successors) to improve the efficiency of the algorithm. Weight was modified to assign a critical task, which can cause delay, and its predecessors as early as possible. Therefore, a sufficiently large number was defined as the weight of the critical task to give higher priority to it and its predecessors. This technique was called *critically ranked positional weight* method. The developed GA algorithm had an adaptation feature and the genetic parameters were determined through a set of preliminary experiments. Another GA-based approach, with group-number encoding and a specific decoding scheme which can compute the number of mated-stations, was proposed by Kim *et al.* (2009). They adopted localised evolution and steady-state reproduction method to increase the diversity of population and performance of the algorithm.

A different GA-based approach was developed by Taha *et al.* (2011) to find the minimum number of mated-stations for two-sided assembly line balancing problem. This version of GA had a new initial population generating method that could generate feasible solutions in different areas of the search space. It also applied a hybrid crossover and a modified scramble mutation operators, which were able to preserve the feasibility of all solutions from one generation to another, to obtain optimum and near-optimum solutions. For the assignment of tasks to mated-stations, a specified station oriented procedure was adopted, which defined side assignment rules. Side assignment rules were also employed to reduce the solution space.

Purnomo *et al.* (2013) enhanced GA with an iterative first-fit rule to address type-II two-sided assembly line balancing problem with assignment restrictions; *i.e.* zoning, distance, resource, and positional constraints. The objective of the proposed mathematical model for both algorithms was to minimise the cycle time, which is equivalent to maximising the line efficiency, for a given number of mated-stations. Workload balance between workstations was also considered in the objective function for the aim of assigning tasks to the workstations smoothly. A modification of group encoding procedure was implemented to represent solutions in the GA. Crossover operator was also modified from a structural crossover. In each iteration of iterative first-fit rules, expected cycle time was recomputed according to the current best cycle time, dynamically. The main concept of this heuristic was narrowing the gap between the expected and current values of cycle time in each loop. GA performed better for small and large-sized instances while iterative first-fit rule performed better for medium-sized test problems.

Rabbani *et al*. (2012) proposed multiple U-shaped layout to handle the mixed-model two-sided assembly line balancing problems. The proposed approach was a modification of multiple U-shaped layout introduced by Miltenburg (1998). A mixed integer programming formulation was implemented to model such manufacturing systems with two conflicting objectives: minimising the *(i)* number of workstations, and *(ii)* cycle time by taking zoning constraints and synchronised tasks constraints into account. They also developed a GA-based heuristic that incorporates an alteration process for assignment of the empty positions in the generated offspring.

After GA, simulated annealing algorithm is the second popular among meta-heuristics in two-sided assembly line balancing domain. Ozcan and Toklu (2009a) presented a new mathematical model and a simulated annealing algorithm for the mixed-model two-sided assembly line balancing problem. The main objective of the algorithm was to minimise the number of mated-stations and to minimise the number of stations as an additional goal. To measure the solution quality, weighted line efficiency and weighted smoothness index were used simultaneously. To reflect the practical conditions in real life, positive and negative zoning constraints, positional constraints, and synchronous tasks constraints were also considered in the presented mathematical model.

Stochastic environment of two-sided assembly line balancing problem was first addressed by Ozcan (2010). Ozcan (2010) proposed a chance-constrained piecewise-linear mixed integer programme to reflect task time variations that may result from lack of training, operator skills, breakdowns, *etc.* A simulated annealing algorithm was also applied as a solution approach for the highly combinatorial nature of the problem, especially for large-sized instances. A balancing solution was represented using a priority list where $i$th position denotes the $i$th task. The value of the $i$th position represented the priority of that task. A station oriented procedure was used to obtain a feasible solution. After obtaining a solution, a classical simulated annealing algorithm sought better solutions by using neighbourhood generating mechanism. This study was a good starting point for future studies in stochastic two-sided assembly line balancing problem.

Baykasoglu and Dereli (2008) and Simaria and Vilarinho (2009) made use of ACO techniques to solve two-sided assembly line balancing problem. In the method of Baykasoglu and Dereli (2008), ant colony based heuristic accounted zoning constraints and aimed to minimise the number of workstations. In addition to this goal, proposed algorithm looked for solutions that maximise the work-relatedness, where possible. The work was one of the first attempts to solve two-sided assembly line balancing problem with zoning constraints using an ACO algorithm. The global solution was updated according to the current solution after each generated solution by an ant. To compare the performance of the algorithm two types of experiments were carried out: *(i)* with zoning constraints, and *(ii)* without zoning constraints.

Simaria and Vilarinho (2009) addressed mixed-model two-sided assembly line balancing problem and developed an ACO algorithm with two ants (called 2-ANTBAL) to solve this problem. Minimising number of workstations of the line was the ultimate goal. Synchronous tasks constraints and zoning constraints were also envisaged while building a balancing solution. Two ants were employed at each side of the line to build an efficient balancing solution by coordinating the assignment of tasks. A certain amount of pheromone was released in the paths used by the ants to build a solution, according to the quality of the solution obtained. Mixed-model two-sided assembly line balancing problem will be explained more detailed in Section 3.2.3.

Ozcan and Toklu (2009c) proposed a tabu search algorithm for solving two-sided assembly line balancing problems. The main objective was to maximise line efficiency (by minimising the number of stations) while considering smoothness index. They explained the method with two numerical example problems and tested its performance on a set of test problems taken from the literature. The initial solution was constructed as a priority list by assigning tasks to stations with regard to their priority values, between 0 to 1. In this manner, sequence dependent finishing times of tasks were considered and tasks could be ordered in stations, in a breeze. The first assignable task with the highest priority was assigned to the first mated-station by considering operation side, and the set of assignable tasks was updated. This process continued until all tasks were allocated. Then, tabu search algorithm was run iteratively to seek neighbourhood solutions and to obtain near optimal or optimal solutions.

Other researchers who addressed imprecise goals were Ozbakir and Tapkan (2010). They implemented bees algorithm, a relatively new member of swarm intelligence, to solve two-sided assembly line balancing problem by employing the fuzzy mathematical goal programming approach. This was also one of the first attempts to solve an assembly line balancing problem with bees algorithm. Some heuristic rules were integrated with bees algorithm to enhance its solution capacity: shortest processing time, longest processing time, minimum total number of follower tasks, maximum total number of follower tasks, minimum total processing time of follower tasks, maximum total processing time of follower tasks, maximum ranked positional weight, and maximum average ranked positional weight. Three fuzzy multiple objective methods (additive, preemptive, max-min) were also compared to each other in solving two-sided assembly line balancing problem.

Ozbakir and Tapkan (2011) improved bees algorithm to solve two-sided assembly line balancing problem with and without zoning constraints with the objective of minimising the number of mated-stations for a given cycle time. Heuristic-based representation, which builds a sequence of priority rules and applies these rules in order to generate solutions, was used in the research addressed. Some heuristic rules were integrated to the algorithm: shortest processing time, longest processing time, minimum total number of successor tasks, maximum total number of successor tasks, minimum total processing time of successor tasks, maximum total processing time of successor tasks, maximum ranked positional weight, maximum average ranked positional weight, and priority of zoning. The fitness function (referred to as the objective function to measure the quality of obtained solutions) used in this study consisted of two objectives: *(i)* minimising the number of stations, and *(ii)* obtaining a well balanced (smooth) solution.

Chutima and Chimklai (2012) have also addressed multi-objective mixed-model two-sided assembly line balancing problems. They developed particle swarm optimisation, an evolutionary metaheuristic inspired by flocks of birds, to minimise the number of stations. They employed the knowledge of the relative positions of different particles rather than modelling the positions of particles in an absolute manner in new solutions. The number of mated-stations, the number of stations, work-relatedness, and work smoothness were the objectives to be optimised hierarchically. Furthermore, Pareto optimality was examined to allow these conflicting objectives to be optimised concurrently.

To conclude, although there exist large numbers of studies on two-sided assembly line balancing problem, none of them considered parallel lines, and two-sided lines to produce similar product models in a mixed sequence. Moreover, model sequencing is another important issue that needs to be considered in two-sided lines consisting of various product model switches.

## 2.3. Parallel Assembly Line Balancing Problem

The parallel assembly line system is another type of line configuration, which was proposed by Gökçen *et al*. (2006) to increase line efficiency when demand is high enough. In parallel assembly line balancing problem, more than one assembly line is balanced simultaneously and the objective is not only to minimise the number of stations but also to minimise the number of workplaces (Scholl and Boysen, 2009, Ozbakir *et al*., 2011). Parallel assembly lines have some advantages such as minimised idle times, reduced operator requirements, enhanced communication between operators, and improved resource utilisation (Ozcan *et al*., 2010b), over single lines. Parallel assembly lines also provide the advantage of the shortening assembly line but may require tolerating first construction cost. These lines assure more flexibility as other lines can continue production if a problem occurs at a workstation on one of the lines (Gökçen *et al*., 2006).

The major utility of parallel lines is that *split workplaces* are allowed for adjacent lines to improve efficiency. The operator works on two directly opposite stations of neighbouring lines within the same cycle. The operator can perform operations on a work-piece at station $k$ of line $h$, and then performs on another work-piece at station $k$ of line $h + 1$, at each cycle (Scholl and Boysen, 2009).

Figure 2-6 represents input data for a parallel assembly line balancing problem with a single product to be assembled on each of the lines. Product I and Product II require 10 tasks and 8 tasks, respectively, while the common cycle time for both lines is 10 time units. Product I is performed on Line I and Product II is performed on Line II.

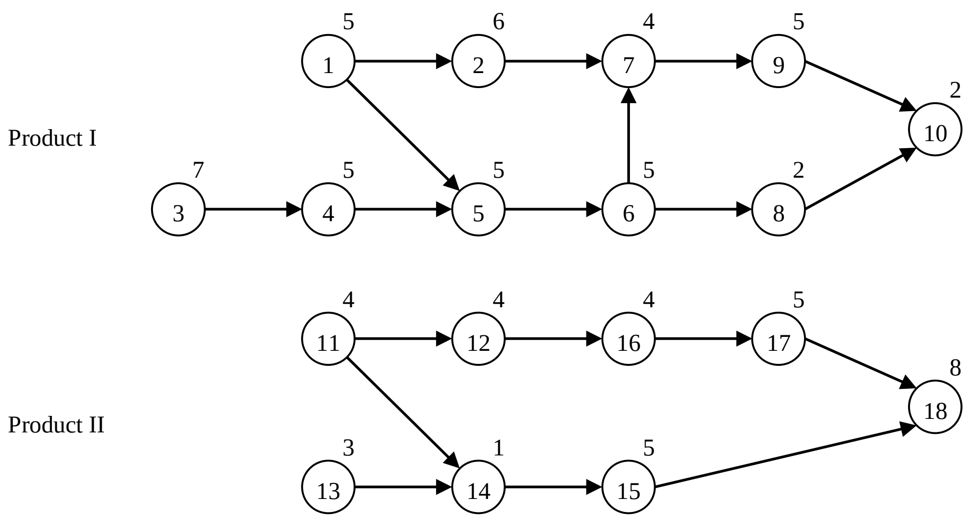


Figure 2-6. An example precedence graph for parallel assembly lines, adapted from (Becker and Scholl, 2009)

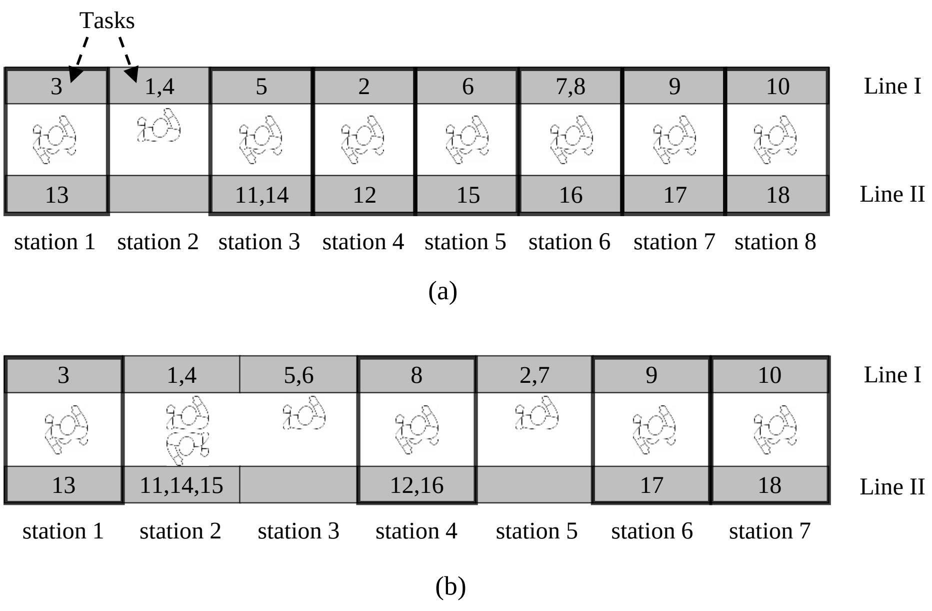


Figure 2-7. Two different solutions (a and b) with split and normal workplaces, adapted from Scholl and Boysen (2009)

If both lines are balanced independently, five stations (workplaces, operators) and four stations (a total of nine workstations) are required to assemble Product I and Product II, respectively. However, required number of workstations (workplaces, operators) is reduced from nine to eight if these lines are balanced together (with split workplaces). For example, if we consider the last stations of either line in Figure 2-7a (station 8), the operator first performs task 10 at a unit of Product I, which needs 2 time units (20% of the cycle time). Then, the operator turns around and performs task 18 at a unit of Product II, which takes 8 time units (remaining 80% of the cycle time). Walking or changing over between lines is not considered due to small distances. In Figure 2-7b, an alternative line configuration that needs the same number of workplaces is given for the same problem. Two operators work in parallel in station pair 2 while only one operator is allocated in station 3 on Line I (Scholl and Boysen, 2009).

As noted above, Gökçen *et al.* (2006) introduced parallel assembly line balancing problem with the idea of locating two or more straight assembly lines in parallel and balancing them simultaneously. They proposed a binary integer programming model and developed heuristics for balancing single-model parallel assembly lines with the objective of minimising the number of workstations.

Two types of scenarios were considered in their research and studied by other researchers thereafter: *(i)* active case, and *(ii)* passive case. If products assembled on the assembly lines are the same, this situation is called *passive case*, and if different, it is called *active case*. In active cases, the tasks performed by operators working in common stations in a production cycle belong to different products (as explained in the example given above). Proposed procedures for active and passive cases are different from each other.

In the passive case procedure, lines are balanced individually and then common stations are applied when there is a workstation whose idle time is equal to or larger than the half of the cycle time, after lines are balanced. So, any type of traditional line balancing procedure can be used in this procedure. Solution procedure used by Gökçen *et al.* (2006) for passive case is summarised as:

- Each line is balanced individually by using any single-model simple assembly line balancing approach.
- Idle times are computed for each workstation.
- Workstation $k$ with idle time equals or greater than the half of the cycle time is determined.
- Tasks in workstation $k$ of the adjacent line are assigned to the relevant station (which is determined in previous step) and this process continues until all workstations to be examined.

Figure 2-8 indicates the precedence relationships of an illustrative example for the passive case (assumes same products on each line where cycle time is considered as 10 time units).

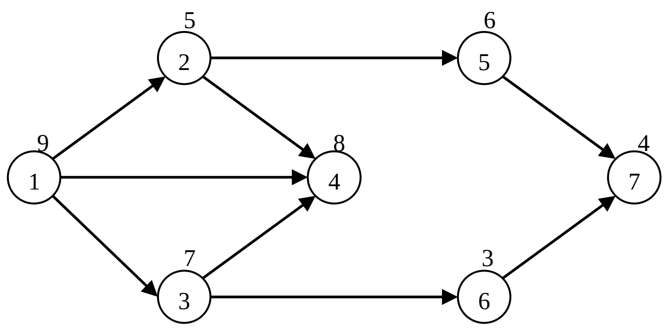
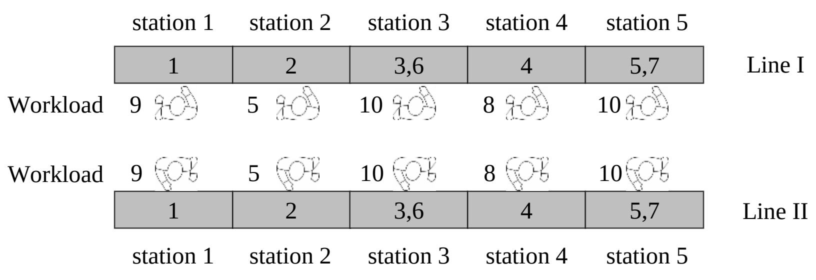


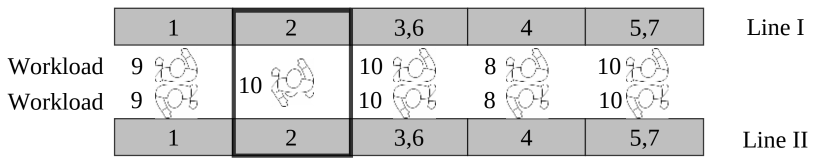
Figure 2-8: Precedence diagram for passive case example, adapted from Gökçen *et al.* (2006)

Figure 2-9a shows the workstation assignments of the lines individually. Since any type of simple line balancing procedure may be of interest, COMSOAL (Arcus, 1966) method can be used for this aim. COMSOAL method uses the approach of 'determining available tasks and assigning them to the current station until all tasks are assigned', which was explained in Section 2.2. After applying common workstations using the procedure summarised above (Gökçen *et al.*, 2006), a better solution is obtained with one lower number of workstations as could be distinguished from Figure 2-9b.



| | station 1 | station 2 | station 3 | station 4 | station 5 | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3,6 | 4 | 5,7 | Line I |
| Workload | 9 | 5 | 10 | 8 | 10 | |
| Workload | 9 | 5 | 10 | 8 | 10 | |
| | 1 | 2 | 3,6 | 4 | 5,7 | Line II |
| | station 1 | station 2 | station 3 | station 4 | station 5 | |

(a)



| | | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3,6 | 4 | 5,7 | Line I |
| Workload | 9 | 10 | 10 | 8 | 10 | |
| Workload | 9 | | 10 | 8 | 10 | |
| | 1 | 2 | 3,6 | 4 | 5,7 | Line II |

(b)

Figure 2-9. Task assignments to workstations, (a) individually (b) together, adapted from Gökçen *et al.* (2006)

The task assignment logic in the active case procedure is also quite similar to COMSOAL (Arcus, 1966) method. In summary, assignable tasks are determined for two adjacent lines and one of them is assigned to the current workstation in each step, unlike in passive case procedure. The procedure is run for all possible line sequences (*e.g.* 1-2-3, 1-3-2, *etc.*) and the balance with the least number of workstations is selected as the solution of the problem. The steps of the procedure proposed by Gökçen *et al.* (2006) for the active case situation is listed in Figure 2-10.

(i) Initialization step (set $x = 0, h = 0, k = 0$).
(ii) Start new trial; set $x = x + 1$.
(iii) $h = h + 1$ until $H - 1$, set $k = k + 1$ (open new workstation).
(iv) For all tasks $i \in (L_h, L_{h+1})$ a set of assignable tasks ($F$ list) to workstation $k$ is determined (if processing time of task $i$, $t_{ih} \leq d_k$, then add $i$ to the $F$ list).
(v) For all tasks $i \in F$, if $t_{ih} = d_k$, then add $i$ to the $Z$ list.
(vi) If $Z \neq \emptyset$, then set $m = Card(Z)$. Randomly generate $RN \in Uniform(1, m)$. Assign the $RN$th task to the relating station and remove the $RN$th task from the relating precedence diagram and update the $d_k$ and $F$ list.
(vii) If $Z = \emptyset$ and $F \neq \emptyset$, set $m = Card(F)$. Randomly generate $RN \in Uniform(1, m)$. Assign the $RN$th task to the relating station and remove the $RN$th task from the relating precedence diagram and update the $d_k$ and $F$ list.
(viii) If ($F = \emptyset$) and unassigned tasks are available, then go to step (iii); if $F \neq \emptyset$, then go to (iv).
(ix) If the number of stations is less than the previous trial, update the best assignments. If $x = X$, then STOP, otherwise go to step (ii).

where $x$ $(x = 1, \dots, X)$, $h$ $(h = 1, \dots, H)$, $k$ $(k = 1, \dots, K)$, $i$, $L_h$, $d_k$, and $Card(Z)$ represent number of trials, line number, station number, task number, line $h$, idle time of $k$, and number of tasks in set $Z$, respectively.

Figure 2-10. The active case procedure proposed by Gökçen *et al.* (2006)

In meta-heuristics, an initial solution generation procedure is integrated to start the algorithm. When the number of initial feasible solutions required is obtained, the algorithm iterates to produce better solutions from obtained ones until the stopping criterion is satisfied. To describe the general approach, an explanatory example, which includes a tabu search approach, is given from Ozcan *et al.* (2009). Precedence relationships and task times are given in Figure 2-11 for the example problem. Cycle times of the lines are assumed 8 and 10 time units for Line I and Line II, respectively.

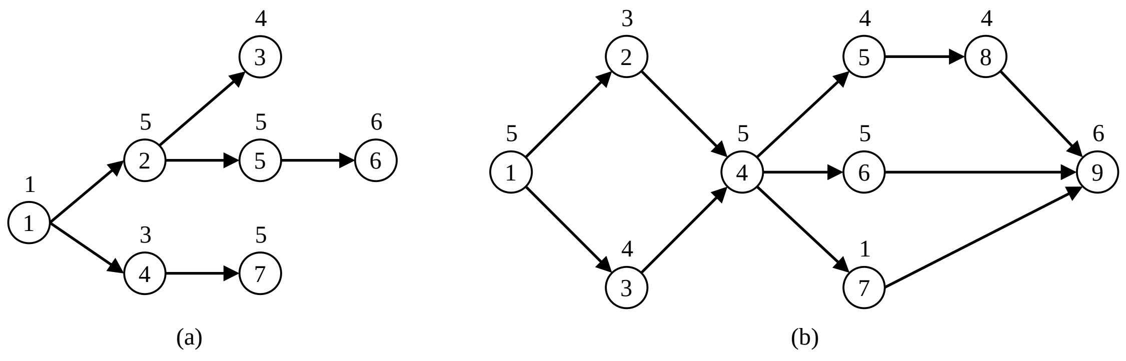
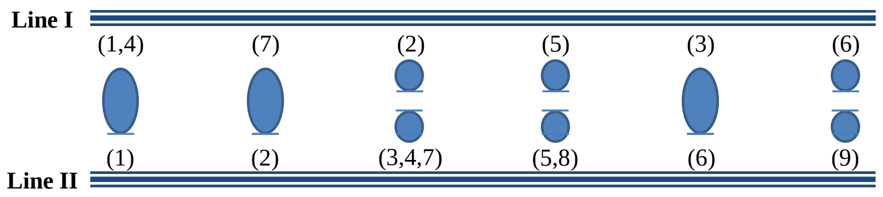


Figure 2-11. The input data for the example: (a) 7-task problem and (b) 9-task problem

The proposed algorithm starts with an initial solution (which is a random task assignment order), referred to as the current solution, and neighbourhood solutions are then generated by using a moving mechanism (swap operator). Initial assignment order and the parallel assembly line balance are given in Figure 2-12a and Figure 2-12b, respectively (Ozcan *et al*., 2009). As could be seen in Figure 2-12b, split workplaces lie across two adjacent lines.

| Task [Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Assignment Order | 7 | 9 | 16 | 6 | 10 | 15 | 8 | 5 | 12 | 13 | 1 | 3 | 11 | 4 | 2 | 14 |

(a)



(b)

Figure 2-12. Initial assignment order and line balance, adapted from Ozcan *et al*. (2009)

A neighbourhood generation procedure was used in tabu search algorithm proposed by Ozcan *et al*. (2009) to solve parallel assembly line balancing problem multi objectively. To obtain new solutions, all neighbourhood solutions are generated using swap operator and objective values are calculated. Table 2-2 lists all candidate moves and objective function values for the first iteration. As can be seen from Table 2-2, the best objective function value among the 15 generated solutions belong to move of ‘2[1]-1[2]’. So, this move is accepted as the new solution and added to tabu list. After the first iteration, new assignment order and line balance are shown in Figure 2-13 (Ozcan *et al*., 2009).

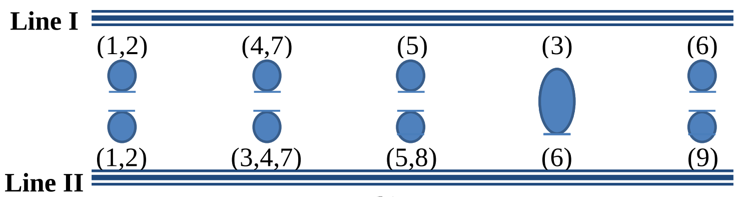
Table 2-2. Candidate moves and objective function values of first iteration for the given example, adapted from Ozcan *et al*. (2009)

| Candidate move (swap) | | Line efficiency | Workload variance | Obj. function value | Candidate move (swap) | | Line efficiency | Workload variance | Obj. function value |
|---|---|---|---|---|---|---|---|---|---|
| 2[1] | 1[1] | 81.39 | 0.163 | 2 | 2[1] | 3[2] | 73.25 | 0.183 | 2.56 |
| 2[1] | 3[1] | 73.25 | 0.183 | 2.56 | 2[1] | 4[2] | 81.39 | 0.163 | 2 |
| 2[1] | 4[1] | 81.39 | 0.163 | 2 | 2[1] | 5[2] | 81.39 | 0.163 | 2 |
| 2[1] | 5[1] | 81.39 | 0.163 | 2 | 2[1] | 6[2] | 81.39 | 0.163 | 2 |
| 2[1] | 6[1] | 73.25 | 0.183 | 2.56 | 2[1] | 7[2] | 81.39 | 0.163 | 2 |
| 2[1] | 7[1] | 81.39 | 0.163 | 3 | 2[1] | 8[2] | 81.39 | 0.163 | 2 |
| 2[1] | 1[2] | 81.39 | 0.147 | 1.90* | 2[1] | 9[2] | 73.25 | 0.183 | 2.56 |
| 2[1] | 2[2] | 81.39 | 0.163 | 2 | - | - | - | - | - |

A tabu list is employed to check the forbidden and allowed moves from a solution to its neighbourhood (Figure 2-14). If the objective value of any newly generated solution is better than the current best solution, then the best solution is updated. The current best solution is then taken as the final solution when the maximum iteration number is exceeded. The primary aim is to minimise the number of workstations while the secondary objective is to minimise the variation of workloads between stations (Ozcan *et al.*, 2009).

| Task [Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Assignment Order | 7 | 5 | 16 | 6 | 10 | 15 | 8 | 9 | 12 | 13 | 1 | 3 | 11 | 4 | 2 | 14 |

(a)



(b)

Figure 2-13. New assignment order and line balance after the first iteration for the given example, adapted from Ozcan *et al.* (2009)

| | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1[1] | | | | | | | | | | | | | | | |
| 2[1] | | | | | | | 4 | | | | | | | | |
| 3[1] | | | | | | | | | | | | | | | |
| 4[1] | | | | | | | | | | | | | | | |
| 5[1] | | | | | | | | | | | | | | | |
| 6[1] | | | | | | | | | | | | | | | |
| 7[1] | | | | | | | | | | | | | | | |
| 1[2] | | | | | | | | | | | | | | | |
| 2[2] | | | | | | | | | | | | | | | |
| 3[2] | | | | | | | | | | | | | | | |
| 4[2] | | | | | | | | | | | | | | | |
| 5[2] | | | | | | | | | | | | | | | |
| 6[2] | | | | | | | | | | | | | | | |
| 7[2] | | | | | | | | | | | | | | | |
| 8[2] | | | | | | | | | | | | | | | |

Figure 2-14. Tabu list after the first iteration for the given example (Ozcan *et al.*, 2009)

Although the literature on assembly line balancing problems is rather extensive, the studies on parallel assembly line balancing problem are quite limited. Table 2-3 summarises the main contributions regarding parallel line balancing problems and lists out the proposed approaches till now. Some researchers, such as Suer (1998) and Miltenburg (1998), studied on multiple line balancing problems. However, the problems considered in these researches are quite different from the real parallel assembly line balancing problem introduced by Gökçen *et al.* (2006) – please note that the problem

Table 2-3. Summary of the main contributions in the literature on parallel assembly line balancing problems between 1998-2013

| **Research** | **Method / solution approach** | **Main object. (min)** | **Additional features/constraints** |
|---|---|---|---|
| Suer (1998) | 3-phase heuristic with IP and MILP model | Number of lines and workstations | Dynamic number of lines |
| Gökçen *et al*. (2006) | Heuristic procedures and a mathematical programming model | Number of workstations | Fixed number of lines |
| Benzer *et al*. (2007) | A network model | Number of workstations | Fixed number of lines |
| Lusa (2008) | Survey | | |
| Baykasoglu *et al*. (2009) | ACO | Number of workstations | Fixed number of lines |
| Cercioglu *et al*. (2009) | Simulated annealing based approach | Number of workstations | Fixed number of lines |
| Ozcan *et al*. (2009) | Tabu search algorithm | Number of workstations | Fixed number of lines, workload balance between workstations |
| Scholl and Boysen (2009) | Binary linear programme and SALOME based exact solution procedure | Number of workstations and operators | Product-line assignment considered |
| Kara *et al*. (2010) | Two goal programming approaches | Number of workstations, cycle time, and task loads of workstations | Three conflicting goals |
| Ozcan *et al*. (2010a) | Simulated annealing algorithm | Number of workstations, workload variance between workstations | *Mixed-models* and model sequencing considered |
| Ozcan *et al*. (2010b) | Tabu search algorithm | Number of workstations | Two-sided lines |

studied by Gökçen *et al*. (2006) and their solution method have already been given above.

Suer (1998) studied alternative assembly line design strategies for a single product. Operations are pre-grouped into stations first, and the multiple operators are assigned to each parallel workstation. Instead of balancing multiple lines, the objective was to determine the number of parallel lines with minimum total manpower using a 3-phase methodology: *(i)* assembly line balancing, *(ii)* determining parallel stations, and *(iii)* determining parallel lines. Therefore, no more detail will be given here about that studies as the problem handled was quite different from the parallel assembly line balancing problem, which was initially proposed by Gökçen *et al*. (2006).

A network model, based on Gutjahr and Nemhauser (1964)'s shortest route model, was developed by Benzer *et al*. (2007) to find out the minimum number of workstations for parallel assembly line balancing problems. In the proposed procedure, the arcs represent workstations, and the nodes symbolise the first possible workstation for tasks while arc lengths correspond to idle times of workstations. Thus, the main aim of the procedure was to seek the shortest path in the network or the minimum number of arcs.

The only exact solution approach for parallel assembly line balancing problem belongs to Scholl and Boysen (2009). They addressed split workplaces while giving a detailed problem description of parallel assembly line balancing problem. This problem was slightly different from previous ones in terms of the design of workstations. Because, split workplaces allows a single worker to operate on two directly opposite stations of adjacent lines. Thus, an operator could perform by splitting the available cycle time between two directly related stations of neighbouring lines. To keep changeover times at a minimum, only the directly opposite stations were linked. They also modelled the problem as a binary linear programme and implemented an exact solution procedure based on SALOME, a well-known bidirectional branch and bound procedure developed by Scholl and Klein (1997) for assembly line balancing problem, and considered product-line assignment decision different from Gökçen *et al*. (2006).

In the literature, there exist relatively large numbers of studies which propose alternative meta-heuristics to solve the parallel assembly line balancing problem. The first and only ACO algorithm was developed by Baykasoglu *et al*. (2009) to balance any type of parallel assembly lines. They developed an ACO algorithm, which employs maximum total time of successor tasks priority rule as heuristic information. The proposed algorithm was first soft computing approach to solving a multi-product assembly line balancing problem.

Another pioneering attempt to balance parallel lines with a swarm intelligence based meta-heuristic was made by Ozbakir *et al.* (2011). They tackled parallel assembly line balancing problem providing some opportunities in improving flexibility, productivity, system balance and reducing breakdown reactiveness. These objectives become more important especially when the capacity of a production system is not sufficient. Therefore, Ozbakir *et al.* (2011) developed a new multiple-colony ant algorithm for balancing bi-objective parallel assembly lines. The algorithm intended to minimise idle times of the workstations and to maximise line efficiency. Pheromone trail was deposited between task and station, and heuristic information obtained from rank-based priority rules were utilised to construct a task selection strategy.

Simulated annealing algorithms also performed well for balancing parallel lines. Cercioglu *et al.* (2009) proposed a simulated annealing based approach which aimed to minimise the number of required workstations for two lines. They also compared their results with previous studies in the relevant literature. Another simulated annealing approach was proposed by Ozcan *et al.* (2010a) to address parallel mixed-model assembly line balancing and model sequencing problem. This problem was an extension of the parallel assembly line balancing problem introduced by Gökçen *et al.* (2006). The main objective of the proposed approach was to minimise the number of workstations while distributing the total workload among the workstations as equally as possible. The proposed simulated annealing approach was run until maximum iteration number was exceeded. It was desired to escape from local optima by increasing the temperature level slowly in each iteration.

Kara *et al.* (2010) extended the formulation of Gökçen *et al.* (2006). They proposed two goal programming approaches containing precise and fuzzy goals to balance parallel assembly lines with multiple objectives. The aim of the proposed approach was to optimise three conflicting goals; namely, number of workstations, cycle time and number of tasks assigned to a workstation. By considering multiple objectives, the research provided flexibility to decision makers to change priorities based on their decision environment and preferences.

Then, Ozcan *et al.* (2010b) introduced parallel two-sided assembly line balancing problem which aimed at balancing more than one two-sided assembly line constructed in parallel, simultaneously. A tabu search algorithm, which combined the advantages of both parallel lines and two-sided lines, was developed to minimise the number of workstations. The proposed tabu search algorithm starts with an initial solution, which is built by randomly sequenced tasks with priority indexes on each of the two-sided assembly lines. Then the tasks are assigned to the workstations considering the priority value of tasks. The best solution is updated when a newly generated neighbourhood solution is better than the current best. This study will be explored in detail in Section 3.2.1.

A comprehensive review of the literature on parallel assembly lines has been presented by Lusa (2008). Lusa (2008) described the main literature contributions briefly with a summary of the state of the art. Multiple (or parallel) assembly line balancing problem has been discussed in detail with advantages and disadvantages of adopting multiple lines. Please refer to that study for more information about the literature on parallel assembly line balancing problems.

## 2.4. Chapter Summary

As already mentioned above, an assembly line is a flow oriented production system, which consists of a number of workstations that are connected by a material handling system, like a conveyor or moving belt (Purnomo et al., 2013). Assembly line balancing problem is determining the optimal assignment of tasks to the workstations by considering some constraints to obtain a cost-efficient line for the purpose of satisfying customer demands on time. Although the core of the line balancing problem is the same, it differentiates significantly based on the layout and configuration of the line utilised, product model(s) produced on the line, *etc.* Several different conditions in real manufacturing systems, such as marketing strategies, manufacturing techniques, and technological restrictions make assembly line balancing problem multi-faced.

This chapter presented existing research on two-sided line balancing and parallel line balancing problems, which are two main roots of the MPTALB/S problem. Both two-sided and parallel assembly line systems were explained with the help of illustrations and their generic solution building approaches were described. A comprehensive review of the literature was provided on sub-problems of MPTALB/S with the help of summary tables to define the gap in the literature. The next chapter continues to reviewing the literature on the binary combinations of mixed-model lines, parallel lines, and two-sided lines, and their solution approaches. Also, existing ACO and multi-agent systems applications implemented for various types of line balancing problems will be presented in the next chapter.

# 3

# THE REVIEW OF THE LITERATURE ON HYBRID LAYOUTS & RELATED ACO – MULTI AGENT SYSTEMS APPROACHES

**Contents**

## 3.1. Chapter Introduction

Further on the first half of the literature review presented in the previous chapter, this chapter constitutes the second half of the literature review, and surveys the existing methodologies on the combinations of the sub-problems of mixed-model parallel two-sided assembly lines. In Section 3.2, a brief introduction and a summary table of the existing methodologies on the mixed-model assembly line balancing problem are given first, followed by illustrative definitions of sub-problems' combinations and their solution approaches, which were developed till now. Existing ACO and multi-agent approaches, which were proposed to solve various assembly line balancing problems, are also respectively presented in Section 3.3 and Section 3.4. Finally, the summary of the chapter is provided in Section 3.5.

## 3.2. Studies on Hybrid Line Configurations

In terms of the variety and characteristics of product models assembled on the line, assembly lines can also be classified as single-model assembly lines and mixed-model assembly lines (Kara *et al.*, 2011) as given in Section 1.3.1. A production line in which more than one product model is assembled on the same line with no (or ignorable amount of) setup required between product model changes is called mixed-model assembly line (Battini et al., 2007). Mixed-model assembly lines offer several advantages over single-model assembly lines, including avoidance of constructing several lines, satisfaction of diverse customer demands, and minimisation of workforce requirement.

There are numerous studies on mixed-model assembly line balancing in the literature. However, none of them considered parallel lines with two sides as a manufacturing system to produce mixed-models on each of the line. Summary of the literature on parallel lines and two-sided lines were given in the previous chapter. Table 3-1 gives a summary of the main contributions in the literature on mixed-model assembly line balancing problem since 2007. As can be seen from the summary provided in Table 3-1, there is no doubt that there is a gap in the literature on the hybridisation of mixed-model lines with parallel lines and two-sided lines.

Table 3-1. Detailed summary of the main contributions in the literature on mixed-model assembly line balancing problem (since 2007)

| Research | Line configuration | | | | Parallel stations | Main objective (minimisation) | | | | Additional constraints | | | | Sequencing | Setup times | Methodology | Additional objectives/features |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Straight | U-shaped | Parallel | Two-sided | | NS | C | NM | O | Zoning | Positional | Synch. Tasks | Space | | | | |
| Kara *et al*. (2007b) | | ● | | | | | | | ● | | | | | | ● | SAn, new neighbourhood generation | JIT, WLS, constant rate of parts consumption |
| Simaria and Vilarinho (2009) | | | | ● | | ● | | | ● | ● | | ● | | | | Mathematical model, ACO | WLS |
| Choi (2009) | ● | | | | | | | | ● | | | | | | | Goal programming model | Processing time, PW |
| Kara and Tekin (2009) | | ● | | | | ● | | | | | | | | | | MIP formulation, COMSOAL based heuristic | Model mixes, operator travel times in crossover stations |
| Ozcan and Toklu (2009a) | | | | ● | | ● | | ● | | ● | ● | ● | | | | Mathematical model, SAn | WLS |
| Emde *et al*. (2010) | ● | | | | | *Computational evaluation* | | | | | | | | | | *Computational evaluation* | Evaluation of different WLS strategies |
| Ozcan *et al*. (2010a) | | | ● | | | ● | | | ● | | | | | ● | | Simulated annealing | WLS |
| Ozturk *et al*. (2010) | ● | | | | | | | | ● | | | | ● | ● | ● | MIP and Constraint Programming | Minimising the maximum completion time of tasks |
| Ozcan *et al*. (2011) | | ● | | | | | ● | | | | | | | ● | | GA | Stochastic environment |
| Xu and Xiao (2011) | ● | | | | | ● | | | | | | | | | | Robust GA | Uncertain times and *changing demands* |
| Yagmahan (2011) | ● | | | | | ● | | | | | | | | | | Multi-objective ACO | WLS |
| Akpinar and Bayhan (2011) | ● | | | | ● | ● | | | ● | ● | | | | | | Hybrid GA | WLS |
| Hamzadayi and Yildiz (2012) | | ● | | | ● | ● | | | ● | ● | | | | ● | | Priority-based GA, SAn based fitness evaluation approach | WLS |
| Rabbani *et al*. (2012) | | ● | | ● | | ● | | | ● | | | | | | | GA | Crossover stations, travel times |
| Chutima and Chimklai (2012) | | | | ● | | ● | | ● | ● | | | | | | | Multi objective PSONK | WR, WLS |
| Liao *et al*. (2012) | ● | | | | | ● | | | ● | | | | | | | Multi agents framework, TS | WLS |
| Manavizadeh *et al*. (2012) | ● | | | | | ● | ● | | | ● | | | | | | Multi-objective GA | MTO environment |
| Mosadegh *et al*. (2012) | ● | | | | | | | | ● | | | | | ● | | Simulated annealing | Minimising total utility work, station |
| Tiacci (2012) | ● | ● | | | ● | | *Simulation* | | | | | | | | | Object-oriented simulation | Stochastic times, buffers |
| Akpinar *et al*. (2013) | ● | | | | ● | ● | | | | ● | | | | | ● | Hybrid ACO + GA | - |
| Manavizadeh *et al*. (2013a) | | ● | | | | ● | | | | | | | | | | Simulated annealing | Human Eff, WLS, Kanban sys. |
| Kucukkoc *et al*. (2013) | ● | | | | ● | ● | | | | ● | | | | | | GA + RSM | - |

NS: Number of workstations, C: Cycle time, O: Other special objectives, NM: Mated-stations, WLS: Workload smoothness, WR: Work-relatedness, PW: Physical workload, JIT: Just-in-time, MTO: Make to order, ACO: Ant colony optimisation, GA: Genetic algorithm, SAn: Simulated annealing, PSONK: Particle swarm optimisation with negative knowledge, TS: Tabu search, RSM: Response surface methodology.

Ozcan and Toklu (2009a) introduced mixed-model two-sided assembly line balancing problem and proposed a simulated annealing algorithm to deal with the problem. Other meta-heuristics, which are ACO and particle swarm optimisation algorithms, have been developed by Simaria and Vilarinho (2009), and Chutima and Chimklai (2012), respectively, for mixed-model two-sided assembly lines. Rabbani *et al*. (2012) addressed two-sided U-shaped line balancing problem and proposed a GA approach which considers operator travel times as well. Nevertheless, parallel lines were not incorporated in these studies again.

The only study, which addresses product model variations on parallel assembly lines, belongs to Ozcan *et al*. (2010a) and this will be explained in more details in Section 3.2.2. However, there is no study which addresses parallel two-sided assembly line system on which different product models are produced. Although mixed-model parallel two-sided assembly lines are encountered in producing large-sized high-volume products in the industry, none of the researchers has considered this issue so far. Mixed-model parallel two-sided assembly lines offer many benefits to companies by combining the advantages of both parallel lines and two-sided lines with the flexibility of product model variation. Based on this motivation, MPTALB/S problem is described, illustrated and explored with numerical examples and solved using newly developed sophisticated solution techniques in this research.

### 3.2.1. Parallel two-sided assembly lines

Parallel two-sided assembly lines are widely used in the production of one or more similar product models that have similar production processes in a set of two-sided assembly lines constructed in parallel to each other. However, as can be observed from the detailed survey on two-sided lines and parallel lines in Section 2.2 and Section 2.3, respectively, studies on the combination of both configurations are quite new as well as scarce.

Figure 3-1 exhibits an example of the parallel two-sided assembly line configuration. As depicted in the figure, only one product model is allowed to be assembled on each line at a time. Operator 1 and operator 3 perform tasks assigned to the left side of Line I (1 and 2) and the right side of Line II (6), respectively. However, operator 2 performs at either the right side of Line I or the left side of Line II. This operator performs task 5 at the left side of Line II, afterwards he/she turns around and performs tasks 3 and 4 at the right side of Line I. On the other hand, operator 5 performs job only at the left side of Line II (11 and 12) as none of the tasks are assigned to the right side of Line I.

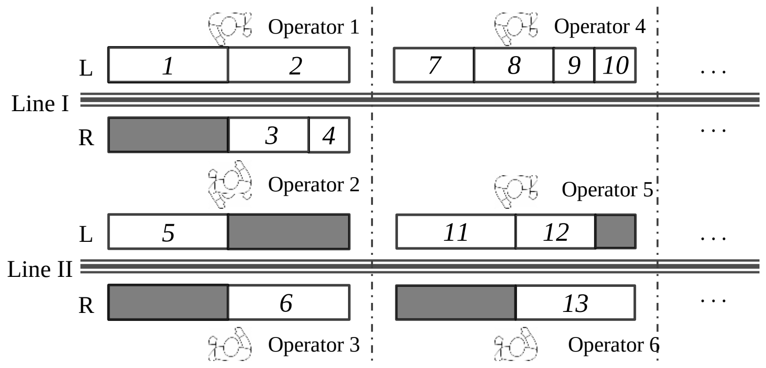


Figure 3-1. Representation of parallel two-sided assembly lines

Parallel two-sided assembly line balancing problem, which aims at balancing more than one two-sided assembly line constructed in parallel simultaneously, was introduced by Ozcan *et al.* (2010b). A tabu search algorithm, which combines the advantages of both parallel lines and two-sided lines, was developed to minimise the number of workstations. The proposed tabu search algorithm starts with an initial solution. The initial solution is built by randomly sequenced tasks with priority indexes on each two-sided assembly line. Then tasks are assigned to the stations considering the priority value of tasks and the best solution is updated when a newly generated neighbourhood solution is better than the current best.

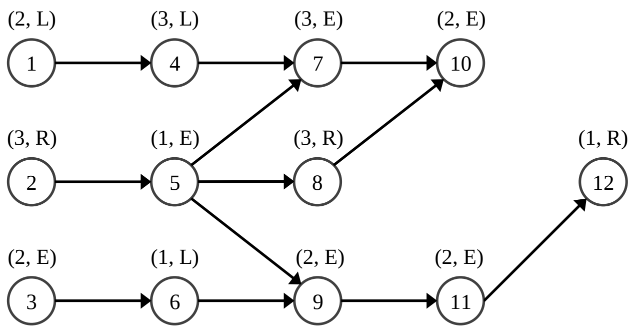


Figure 3-2. Input data for parallel two-sided assembly line balancing problem (Kim et al., 2000c)

To explain the generic solution approach for parallel two-sided assembly line balancing problems, a numerical example is given from Ozcan *et al.* (2010b). Two parallel two-sided assembly lines are subject to balancing for this example and the same product is produced on each line. Cycle time is assumed 8 time units for both of the lines. Figure

3-2 gives the input data of the problem. The numbers in the nodes represent the tasks. As known, some tasks need to be performed on only one side of a two-sided line, *i.e.* left side or right-side, and the remainders can be assigned to either side of a two-sided line. The labels above the nodes denote completion time and preferred operation direction of tasks, respectively. L, R and E indicate that the task should be assigned to a left side, right side and either side station, respectively.

The tabu search algorithm proposed by Ozcan *et al*. (2010b) starts with a feasible initial solution obtained from a sequence which includes priority indexes of all tasks produced on both of the two-sided lines. The priority index of a task is a randomly generated number between one and the total number of tasks (which is 24 for this example) and demonstrates the assignment priority of that particular task. The initial priority indexes of tasks are given in Figure 3-3.

| Task [Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 8[1] | 9[1] | 10[1] | 11[1] | 12[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] | 10[2] | 11[2] | 12[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Priority Index | 1 | 3 | 14 | 24 | 4 | 12 | 2 | 22 | 7 | 8 | 15 | 20 | 6 | 21 | 17 | 10 | 19 | 11 | 18 | 16 | 23 | 5 | 9 | 13 |

Figure 3-3. Initial priority list, adapted from Ozcan *et al*. (2010b)

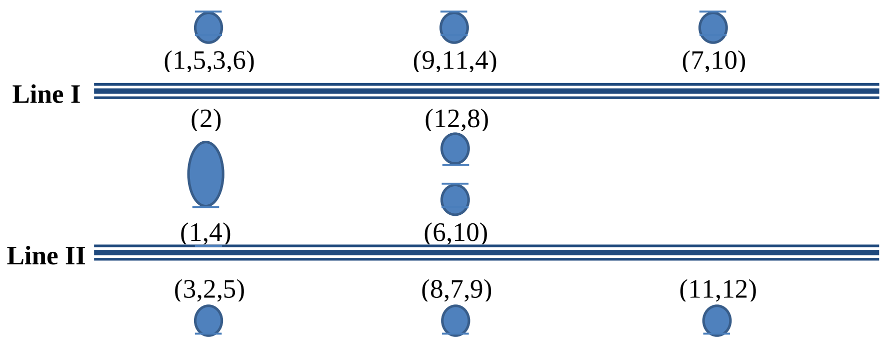


Figure 3-4. Initial feasible solution

The new neighbourhood solutions are generated from the list of initial priority indexes using swap operator. The performance value of each obtained solution is calculated using Equation (3.1) and the best solution is kept for the next generation. Please note that the notation used in the equation is changed to provide conformity with the notation used in this thesis.

$$max \sum_{k=1}^{K} \left( \sum_{i \in A_k \wedge i \in h} pt_{hi} \right)^2 , \qquad (3.1)$$

where, $i$ is the task index ($i = 1, \dots, T_h$), $T_h$ is the total number of tasks, $pt_{hi}$ is the processing time of task $i$ in line $h$, $k$ is the station number ($k = 1, \dots, K$), $K$ is the total number of workstations and $A_k$ is the set of tasks which are assigned to station $k$.

The top five candidate moves, which would yield new neighbourhood solutions, are shown in Table 3-2. The move (2,8) is taken as the first best move and this move is determined as tabu, so that the same move is forbidden for the next five iterations (see the tabu list given in Figure 3-5). In the tabu list, numbers in each cell correspond to the number of iterations remaining until the related moves are allowed again. The new priority list after this move is presented in Figure 3-6.

Table 3-2. Top five candidate solutions at the first iteration, adapted from Ozcan et al. (2010b)

| Swap (Move) | Objective Function |
|---|---|
| (1,4) | 314 |
| (1,21) | 318 |
| (2,8) | 332* |
| (8,19) | 332 |
| (8,20) | 332 |

|  | 2[1] | 3[1] | … | 7[1] | 8[1] | … | 12[1] | 1[2] | 2[2] | … | 12[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1[1] |  |  |  |  |  |  |  |  |  |  |  |
| 2[1] |  |  |  |  | 5 |  |  |  |  |  |  |
| 3[1] |  |  |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |  |  |
| 7[1] |  |  |  |  |  |  |  |  |  |  |  |
| 8[1] |  |  |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |  |  |
| 12[1] |  |  |  |  |  |  |  |  |  |  |  |
| 1[2] |  |  |  |  |  |  |  |  |  |  |  |
| 2[2] |  |  |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |  |  |
| 12[2] |  |  |  |  |  |  |  |  |  |  |  |

Figure 3-5. Tabu list used by Ozcan *et al.* (2010b)

| Task [Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 8[1] | 9[1] | 10[1] | 11[1] | 12[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] | 10[2] | 11[2] | 12[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Priority Index | 1 | 22 | 14 | 24 | 4 | 12 | 2 | 3 | 7 | 8 | 15 | 20 | 6 | 21 | 17 | 10 | 19 | 11 | 18 | 16 | 23 | 5 | 9 | 13 |

Figure 3-6. The new priority list

When the algorithm is terminated, the best solution (represented in Figure 3-7) which has the maximum objective function value is taken. As seen from the figure, seven workstations are needed to perform a total of 24 tasks belonging to both product models on parallel two-sided lines. The objective function value of the solution is 366. The operator located in the common workstations between two lines first completes task 1 on Line II and then completes task 2 on Line I, and finally tasks 3 and 6 on Line II. Shaded areas indicate idle times at the end of cycle time or unavoidable delays between two consecutive tasks.

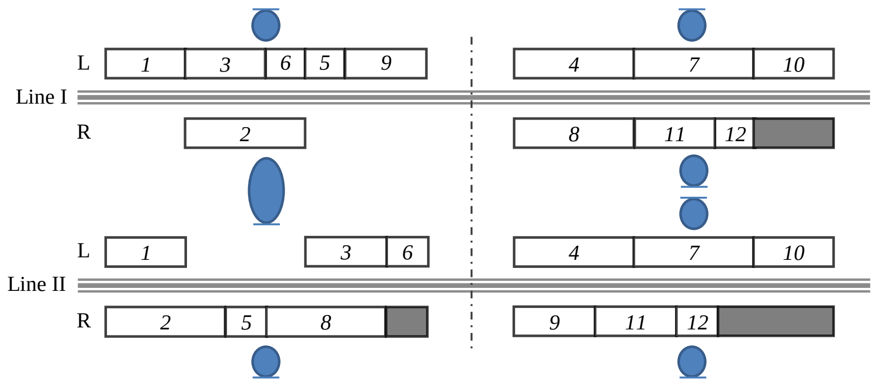


Figure 3-7. Final solution, adapted from Ozcan *et al.* (2010b)

### 3.2.2. Parallel mixed-model assembly lines

The idea of addressing parallel mixed-model assembly lines belongs to Ozcan *et al.* (2010a). This study is unique in the literature in terms of considering parallel line configuration and product model variations at the same time in a line balancing and model sequencing perspective. All possible product model mixes in different production cycles were considered to avoid infeasible balancing solutions. Figure 3-8 exhibits the line system proposed by Ozcan *et al.* (2010a).

As seen from the figure, three lines are constructed in parallel and two product models (A and B), three product models (C, D, and E), and two product models (F and G) are assembled concurrently in the first, second, and third line, respectively. Workstations IV, V, and VIII are called *common stations (or split workplaces)* and serve on pairs of adjacent lines, *i.e.* Line I – Line II, Line I – Line II, and Line II – Line III, respectively. Since many different product model mixes may appear at common stations, the workload of those stations for a cycle depends on the product model mixes as well as the line balance. For that reason, they suggested considering all possible product model mixes in different cycles and balancing lines by considering product model changes.

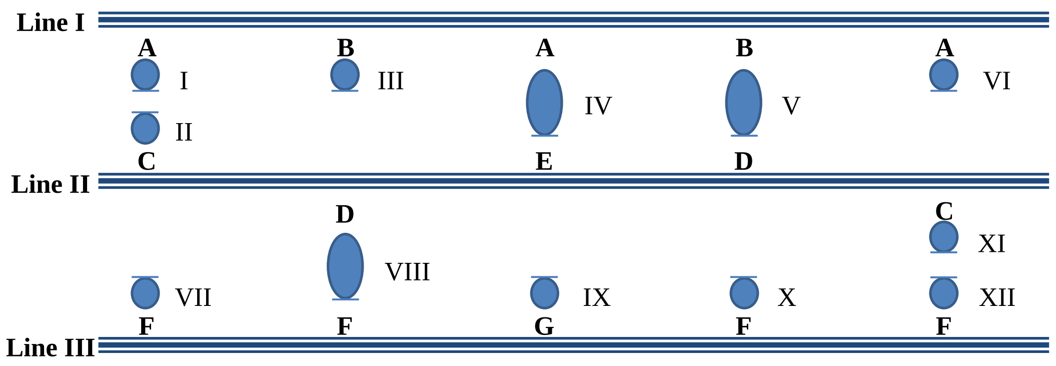


Figure 3-8. Illustration of parallel mixed-model assembly lines with three lines, adapted from (Ozcan *et al*., 2010a)

Ozcan *et al.* (2010a) proposed a simulated annealing approach for simultaneous balancing and model sequencing of parallel mixed-model assembly lines. The main objective of the proposed method was to maximise the efficiency of the line and to distribute workload smoothly across workstations. Two numerical examples, each of which had two parallel lines, were given to explain the proposed approach and test problems were solved to demonstrate the efficiency of the algorithm.

An explanatory example from Ozcan *et al.* (2010a) is given below to describe the simulated annealing algorithm proposed by them and to depict a balanced parallel mixed-model line system. Two mixed-model straight assembly lines are considered to produce product models A, B, C, D, and E; where A and B are assembled on the first line and C, D, and E are assembled on the second line. Figure 3-9a represents combined precedence diagram for product models A and B while Figure 3-9b does for product models C, D, and E. Task processing times are given in Table 3-3.

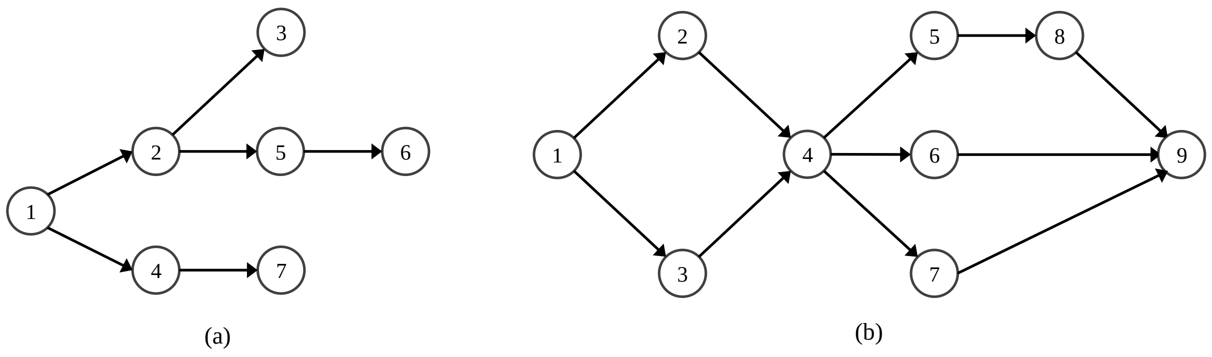


Figure 3-9. Combined precedence diagrams for the given example: (a) product models A and B, (b) product models C, D and E, adapted from Ozcan *et al.* (2010a)

The demands for product models are assumed 42, 42, 20, 20, and 20 units, respectively, while the pre-determined planning horizon is 840 time units. Cycle times of the lines are different and considered as 10 time units for the first line and 14 time units for the second line. In this case, greatest common divisors for lines, $cd_1$ and $cd_2$, are 42 and 20, respectively. Then, minimum part sets are equal to (1, 1) for the first line and (1, 1, 1) for the second line (please note that detailed information on calculation of minimum part sets will be given in Chapter 4).

Table 3-3. Task processing times for the example problem, adapted from Ozcan *et al*. (2010a)

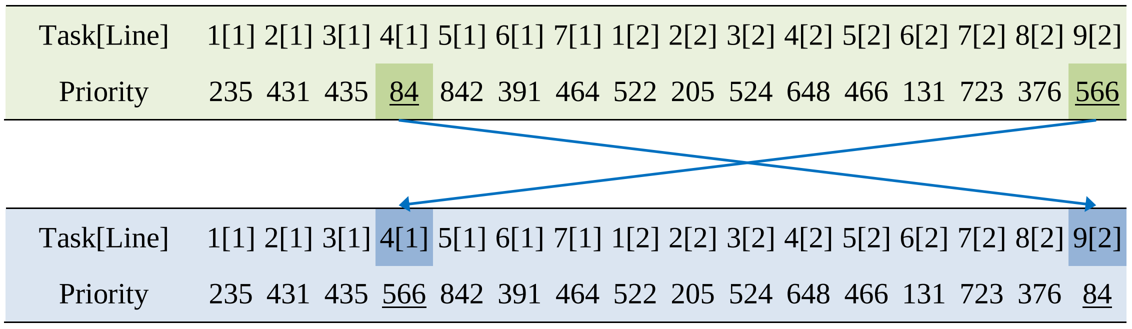
| Line I | | | Line II | | | |
|---|---|---|---|---|---|---|
| Task No | A | B | Task No | C | D | E |
| 1 | 1 | 5 | 1 | 5 | 6 | 1 |
| 2 | 5 | 2 | 2 | 3 | 4 | 2 |
| 3 | 4 | 3 | 3 | 4 | 1 | 5 |
| 4 | 3 | 1 | 4 | 5 | 3 | 3 |
| 5 | 5 | 1 | 5 | 4 | 1 | 3 |
| 6 | 6 | 0 | 6 | 5 | 5 | 1 |
| 7 | 5 | 0 | 7 | 1 | 0 | 4 |
| - | - | - | 8 | 4 | 1 | 3 |
| - | - | - | 9 | 6 | 0 | 2 |

Random task priorities between 1 and 1000 by uniform distribution (see Table 3-4) and product model sequences (*BA* for Line I and *EDC* for Line II) are generated randomly.

Table 3-4. Initial task priority values, adapted from (Ozcan *et al*., 2010a)

| Task[Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Priority | 235 | 431 | 435 | 84 | 842 | 391 | 464 | 522 | 205 | 524 | 648 | 466 | 131 | 723 | 376 | 566 |

After generating an initial line balancing solution according to the task priority values given in the table, new neighbourhood solutions are generated using two different procedures. A random number between zero and one is generated randomly and compared with a predetermined number. If the generated number is larger than the predetermined number, a new product model sequence is generated using swap operator; if not, priority values of two tasks are selected randomly and swapped using a swap operator. Then a new line balancing solution is obtained. Figure 3-10 represents new neighbourhood solution generation for priority assignment value of tasks (Ozcan *et al*., 2010a). This procedure is repeated until the maximum iteration number is exceeded. The best solution of the problem obtained using this method is given in Figure 3-11. As seen from the figure, six stations are utilised -where four stations are common- to perform all tasks required.



| Task[Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Priority | 235 | 431 | 435 | 84 | 842 | 391 | 464 | 522 | 205 | 524 | 648 | 466 | 131 | 723 | 376 | 566 |

| Task[Line] | 1[1] | 2[1] | 3[1] | 4[1] | 5[1] | 6[1] | 7[1] | 1[2] | 2[2] | 3[2] | 4[2] | 5[2] | 6[2] | 7[2] | 8[2] | 9[2] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Priority | 235 | 431 | 435 | 566 | 842 | 391 | 464 | 522 | 205 | 524 | 648 | 466 | 131 | 723 | 376 | 84 |

Figure 3-10. Swapping task priority assignment values, adapted from (Ozcan *et al*., 2010a)

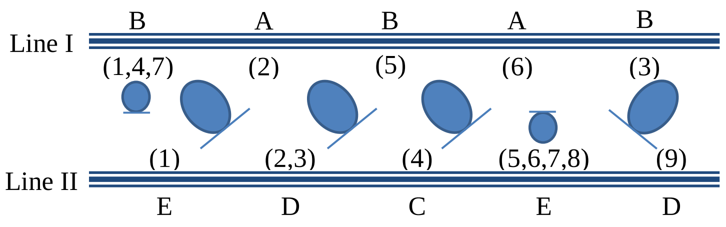


Figure 3-11. Obtained final line balancing solution using simulated annealing algorithm, adapted from (Ozcan *et al*., 2010a)

The advantages of parallel mixed-model assembly lines comprise flexibility of parallel lines and customisability of mixed-model lines. It is obvious that constructing mixed-model assembly lines in parallel provides many advantages over *(i)* single-model lines and *(ii)* mixed-model single lines (without parallelisation). However, in today's manufacturing environment, two-sided assembly lines are needed to produce large-sized products in timely manner. Therefore, a competitive line layout is needed to produce customised large-sized products on time.

### 3.2.3. Mixed-model two-sided assembly lines

In the literature, mixed-model two-sided assembly line balancing problem has been studied by only a few researchers. Unlike the parallel mixed-model assembly lines, only one assembly line which has two operational directions is constructed to produce different product models in a mixed-model basis. So, no common stations are utilised as there are no multiple lines and no shared workload is subject to consideration in those lines.

A schematic representation of mixed-model two-sided assembly lines is given in Figure 3-12. As can be seen in the figure, two different product models are produced on the line, and processing times of tasks $(1-9)$ may differ from one product model to another. Moreover, processing time of some tasks may equal to zero for some product models (*i.e.* task 9 for product model *B*). So, finishing time of each task, even at the other side of the line, must be taken into account carefully for each product model to avoid violation of precedence relationships among tasks.

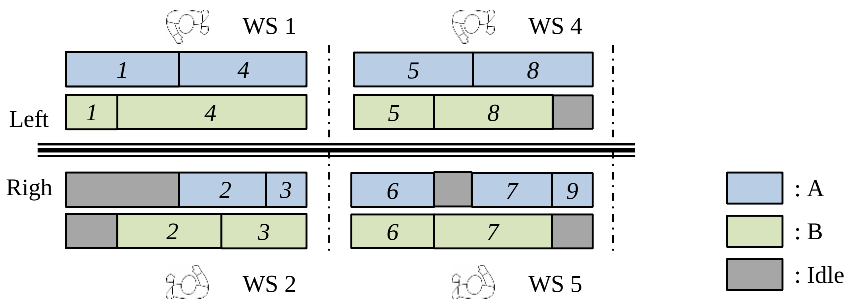


Figure 3-12. A schematic view of mixed-model two-sided assembly lines

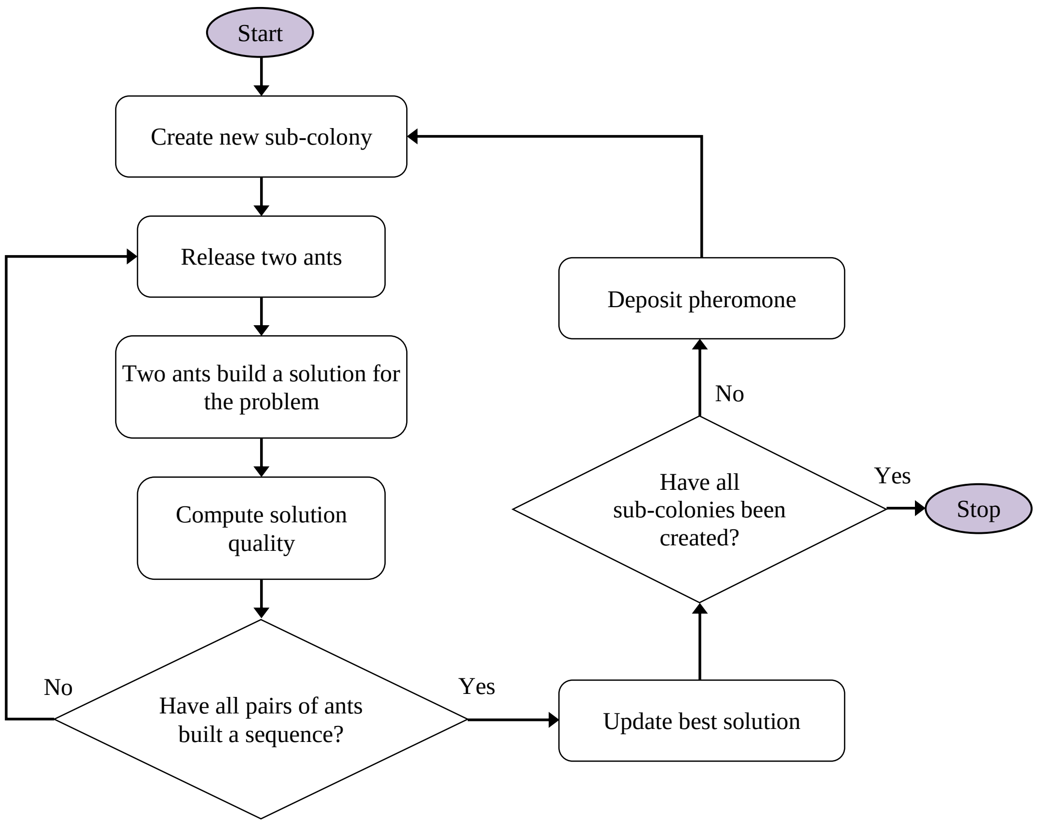


Figure 3-13. Outline of 2-ANTBAL proposed by Simaria and Vilarinho (2009)

The solution approach improved by Simaria and Vilarinho (2009) for this problem could be considered as the generic method followed by other researchers. This research was given in Section 2.2 briefly but will be explored in detail in this section. Simaria and Vilarinho (2009) developed an ACO algorithm (referred to as 2-ANTBAL) and formally described the problem with a mathematical model, which aims at minimising total number of workstations required as well as distributing workload smoothly within and between the workstations. Figure 3-13 outlines the proposed 2-ANTBAL, which has two ants that work simultaneously one at each side of the line. The procedure starts with generating pre-determined number of ant pairs and each pair of ants collaborate each other to build a feasible solution, unlike the traditional ACO algorithm. The rest of the procedures (*i.e.* quality measurement, pheromone depositing, best solution updating) remains the same as the traditional ACO algorithm.

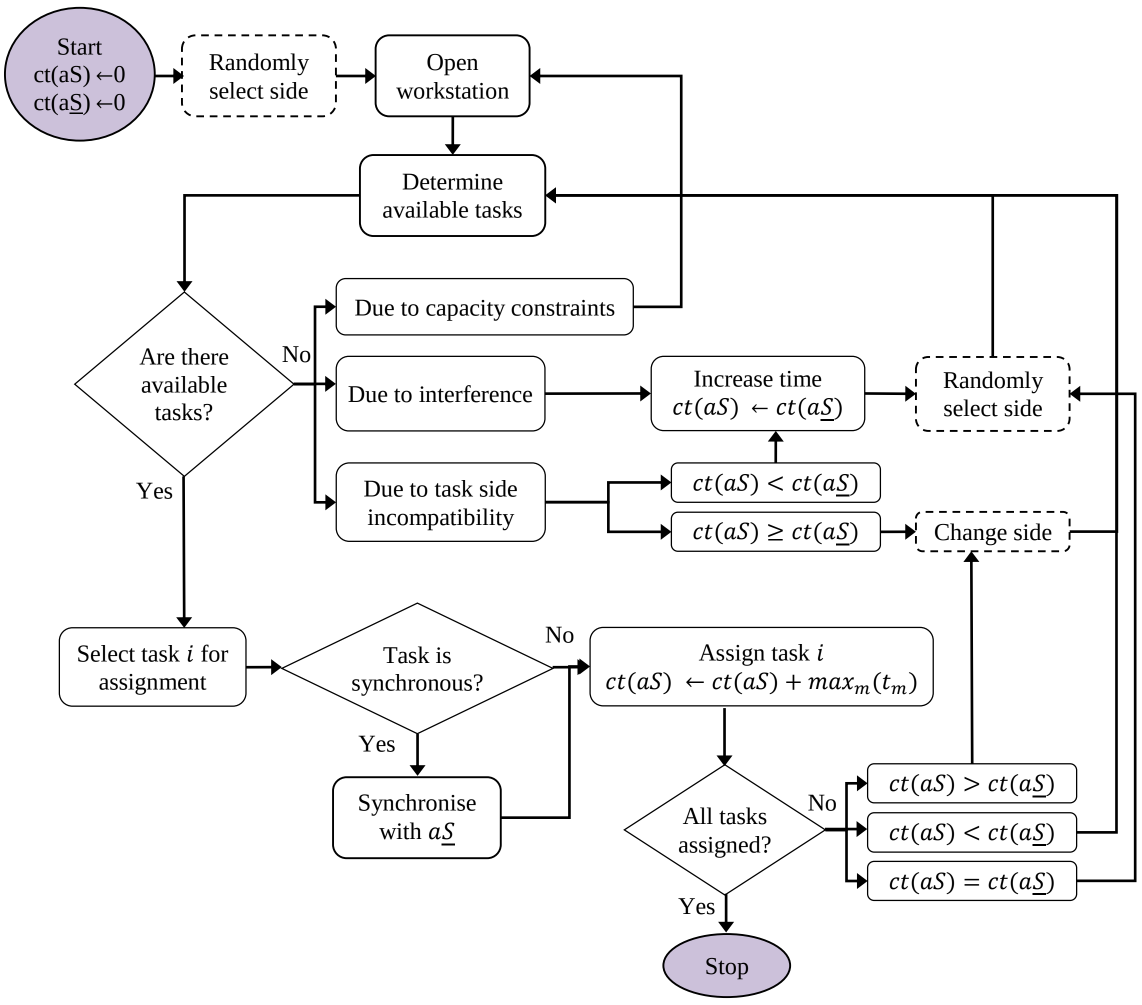

Start
ct(aS) ←0
ct(a$\underline{S}$) ←0
Randomly select side
Open workstation
Determine available tasks
Are there available tasks?
No
Yes
Due to capacity constraints
Due to interference
Due to task side incompatibility
Increase time $ct(aS) \leftarrow ct(a\underline{S})$
Randomly select side
$ct(aS) < ct(a\underline{S})$
$ct(aS) \geq ct(a\underline{S})$
Change side
Select task $i$ for assignment
Task is synchronous?
No
Yes
Synchronise with $a\underline{S}$
Assign task $i$ $ct(aS) \leftarrow ct(aS) + max_m(t_m)$
All tasks assigned?
No
Yes
$ct(aS) > ct(a\underline{S})$
$ct(aS) < ct(a\underline{S})$
$ct(aS) = ct(a\underline{S})$
Stop

Figure 3-14. Building a balancing solution procedure of 2-ANTBAL, adapted from Simaria and Vilarinho (2009)

The most important part of their research was the procedure developed to construct a balancing solution. The way of building a balancing solution by two ants is presented in Figure 3-14. The current time of the one side-ant ($ct(aS)$), and the current time of the opposite side-ant ($ct(a\underline{S})$) are initialised first. Then, one of the sides of the line is selected randomly to begin the assignment and a new workstation is opened. Available tasks are determined according to a set of rules to be assigned to a particular workstation starting at the current time. A task should conform to following conditions to be able to be considered as an available task:

- operation direction constraint
- precedence relationship constraint
- capacity constraint
- zoning constraint
- synchronism constraint.

In case that there is no available task found by the side-ant, one of the following actions is taken according to the unavailability reason (Simaria and Vilarinho, 2009):

- If there is no available capacity to perform the related job, a new workstation is opened by the side-ant.
- A timeline is used by side-ants in the balancing procedure. If there are tasks whose predecessors have been assigned to the opposite side but will be finished in a forward time (this phenomenon is called interference, as noted in Section 1.2), the side-ant moves its current time forward, to the opposite side ant's current time ($ct(aS) \leftarrow ct(a\underline{S})$). Then, the procedure continues with a randomly selected side-ant.
- If there are no tasks that can be assigned to the current side, task side incompatibility occurs. This occurs from one of the following reasons:
    - If the current time of the side-ant is inferior from the current time of the opposite side ant ($ct(aS) < ct(a\underline{S})$): The side-ant moves its current time forward to the current time of the opposite side-ant and then a random side is selected to continue.
    - If the current time of the side-ant is equal to or larger than the opposite side-ant's current time ($ct(aS) \geq ct(a\underline{S})$): The control of the assignment procedure is taken by the opposite side-ant.

Then, a side-ant selects one task from the available tasks list to be assigned to the current station starting at the current time. Task selection procedure includes two types of rules (static and dynamic), which will not be given here in detail since it is more related to the ACO mechanism. Please refer to Simaria and Vilarinho (2009) to find out more.

The current time of the related workstation is increased for an amount of time corresponding to the task processing time, every time a task is assigned to the workstation. Processing time is considered as the maximum processing time of that task for all product models ($max_m(t_m)$) due to the mixed-model nature of the problem. Current times of both side-ants are compared and one of the following conditional options is executed:

- $ct(aS) < ct(a\underline{S})$: The assignment continues on the same side.
- $ct(aS) > ct\left(a\underline{S}\right)$: The side is changed.
- $ct(aS) = ct(a\underline{S})$: The assignment continues on a randomly selected side.

Task assignment procedure is repeated until there is no unassigned task and the quality of the solution is computed to release pheromone. A new sub-colony is created and new solutions are built upon all pairs of ants complete their tours. The best solution is taken as the solution of the problem when all sub-colonies finish their tours.

For the associated problem, simulated annealing and particle swarm optimisation algorithms have been proposed by Ozcan and Toklu (2009a) and Chutima and Chimklai (2012), respectively, in the literature. Ozcan and Toklu (2009a) considered minimisation of the number of mated-stations as the primary goal and the number of stations as the secondary goal (for a given cycle time) in their newly developed mathematical model. A simulated annealing algorithm; which considers two performance criteria to measure the solution quality: *(i)* maximising weighted line efficiency, and *(ii)* minimising weighted smoothness index, has also been applied. Positive and negative zoning constraints, positional constraints and synchronous tasks constraints were also considered in the presented mathematical model to reflect the real world conditions in the industry.

Chutima and Chimklai (2012) have also addressed to mixed-model two-sided assembly line balancing problems. It will not be repeated here as it has already been given in Section 2.2.

Sequencing of the product models were not considered in these studies; which were carried out by Simaria and Vilarinho (2009), Ozcan and Toklu (2009a), and Chutima and Chimklai (2012), since product model mixes were not important as a setup operation is not required when passing from one product model to another.

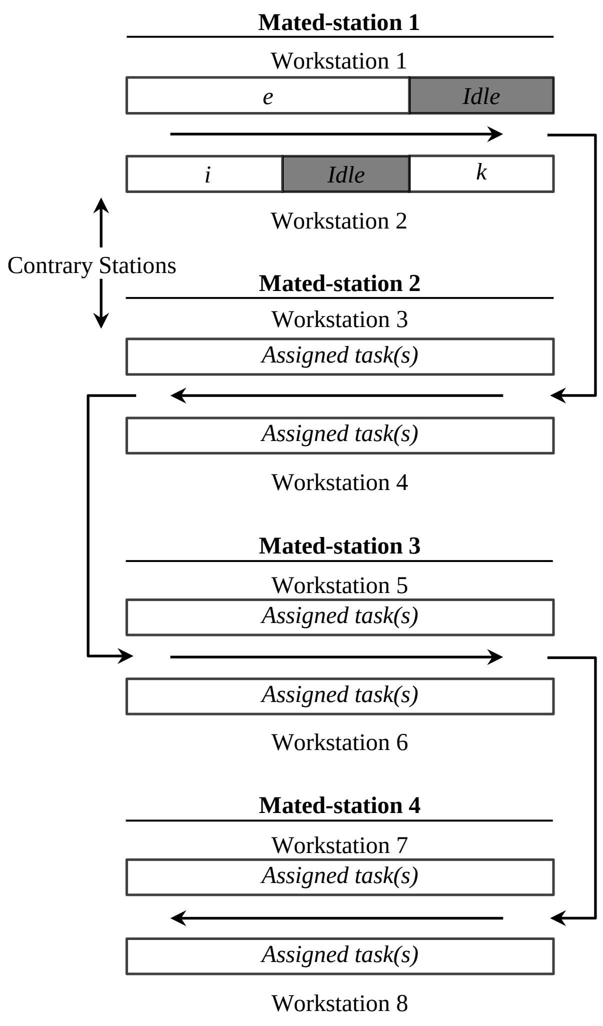


Figure 3-15. Multiple U-shaped layout proposed by Rabbani *et al.* (2012)

However, Rabbani *et al.* (2012) proposed multiple U-shaped layout to deal with mixed-model two-sided assembly line balancing problem and utilised contrary workstations. They also modelled the problem with mixed integer programme, which optimises two conflicting objectives: minimising the *(i)* number of workstations, and *(ii)* cycle time; and solved the problem with a GA-based heuristic. Nevertheless, they did not consider sequencing of the product models in their study. As illustrated in Figure 3-15 the operators work in the utilised contrary stations help each other in their idle times and processing time of a task that is handled by two operators was modified with a coefficient. Therefore, the weighted efficiency of the line was increased as using two operators instead of one leads an improvement in the processing time of this task within this period. However, as the proposed layout requires multiple turns of product models made on the line due to its unique shape, material handling costs are expected to increase with this configuration.

Apparently, the sequence of product models must be known in order to determine which product model is being performed at a specific station. Additionally, task times may vary from one product model to another and idle times of an operator in a workstation depend on the processing times of tasks assigned to that station. Using this underlying idea, model sequencing problem has been considered along with the mixed-model parallel two-sided assembly line balancing problem in this thesis. A brief background information on fundamentals of simultaneous balancing and sequencing of mixed-model assembly lines will be given in the following chapter.

## 3.3. ACO Implementations on Assembly Line Balancing Problems

Recently, there has been an increasing interest in applications of meta-heuristic approaches for solving various engineering problems. Meta-heuristics help both academics and practitioners to get not only feasible but also near optimal solutions where obtaining a solution for the relevant problem is not possible in a reasonable time using traditional optimisation techniques. The ACO algorithm is inspired from the collective behaviour of ants and one of the most efficient meta-heuristics in solving combinatorial optimisation problems. One of the main application areas of ACO algorithm is assembly line balancing problem.

In this section, the running principle of ACO algorithm is given first and then the applications of ACO based algorithms on assembly line balancing problems in the literature are reviewed. Strengths and weaknesses of proposed algorithms to solve various problem types in the literature are also discussed in this section. The main aim is to define the gap in this domain and spread the application areas of ACO techniques in various aspects of line balancing problems. Existing researches in the literature indicate that ACO methodology has a promising solution performance to solve line balancing problems, especially when integrated with other heuristic and/or meta-heuristic methodologies.

Ant algorithm, proposed by Dorigo *et al*. (1996), is one of nature inspired algorithms. They developed an ant system meta-heuristic, the initial form of ACO technique, to solve small-sized travelling salesman problem with up to 75 cities. Since then, several researchers carried out a substantial amount of research in ACO algorithm, which demonstrates a better performance than ant system.

ACO algorithm is inspired by the observation of real ant colonies in nature and their capability of finding the shortest path between the nest and food sources where each ant represents a complete solution. The foraging behaviour of ants helps them to find the shortest path by depositing a substance, called pheromone, on the ground while they are walking. In this way, a pheromone trail is formed and ants smell pheromone to choose their way in probability. Paths involving strong pheromone levels have more chance to be selected by ants (Dorigo *et al*., 1999). The pheromone trail is favourable to the succeeding ants which are intended to follow it. When a set of possible paths are given to the ants, each ant chooses one path randomly, and apparently some ants picking the shortest path will return faster. Then, there will be more pheromone on the shortest path, influencing later ants to follow this path, after their completion of one tour. By the time, the path that has high level of pheromone will be most often selected and considered as the shortest route (Leung *et al*., 2010). The famous double bridge experiment (Dorigo *et al*., 1999) depicts the selection of the shortest path by ants (see Figure 3-16).

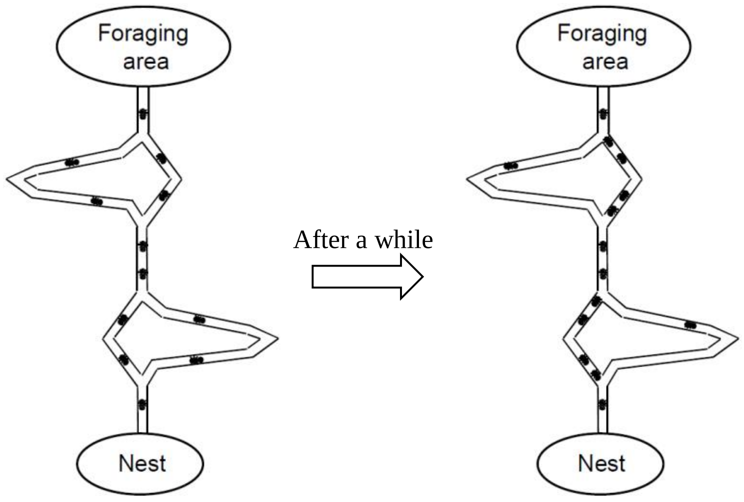


Figure 3-16. Double bridge experiment, adapted from (Dorigo *et al*., 1999)

There exist some rules in the ACO algorithms to determine:

- the amount of pheromone deposited on edges
- the edge chosen by on its way
- the pheromone evaporation speed.

The transition probability of moving from node $i$ to node $j$ for an ant located at node $i$ is computed as follows (Ilie and Badica, 2013):

$$p_{(i,j)} = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_j[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}\,, \tag{3.2}$$

where $\tau_{ij}$ is the amount of pheromone deposited on edge $(i,j)$; $\eta_{ij}$ is the weight of edge $(i,j)$ or heuristic information provided by an integrated heuristic procedure; $\alpha$ and $\beta$ are user determined parameters which control the influences of $\tau_{ij}$ and $\eta_{ij}$ respectively; and $j$ is a non-visited node reachable from node $i$.

The algorithm converges with the help of pheromone update rules. So, more pheromone is laid on each edge of a tour when a better solution is found than the best known, with cost $Cost_a$.

$$\Delta\tau_{ij} = \begin{cases} 1/Cost_{ant} & if\ edge\ (i,j)\ belongs\ to\ found\ tour \\ 0 & otherwise \end{cases}. \tag{3.3}$$

When each ant completes its tour, it will update the pheromone by laying down pheromone on the edges of the travelled path. Additionally, an amount of pheromone will be evaporated from every node either visited or not. Evaporation and pheromone updates are calculated as follows (Ilie and Badica, 2013):

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij}\,, \tag{3.4}$$

where $\rho$ is the pre-determined evaporation rate $(0 \leq \rho < 1)$.

The ACO algorithm proposed by Vilarinho and Simaria (2006), which is called ANTBAL, for mixed-model assembly line balancing problem is explored below to reflect the generic problem solving approach using ACO algorithm in the assembly line balancing field. To outline the main steps of the algorithm for a typical line balancing problem, the flowchart of the proposed algorithm is depicted in Figure 3-17. The primary objective of the algorithm was to minimise the number of operators while maintaining a smooth workload distribution among the stations, as a secondary goal. Construction of parallel workstations was allowed and controlled in such a way that parallelism is only allowed if a station is assigned a task with processing time larger than minimum replication time, which is a predefined value (Vilarinho and Simaria, 2006). Accordingly, parallel workstations are allowed if the processing time of a task assigned to a workstation is larger than the predefined minimum replication time.

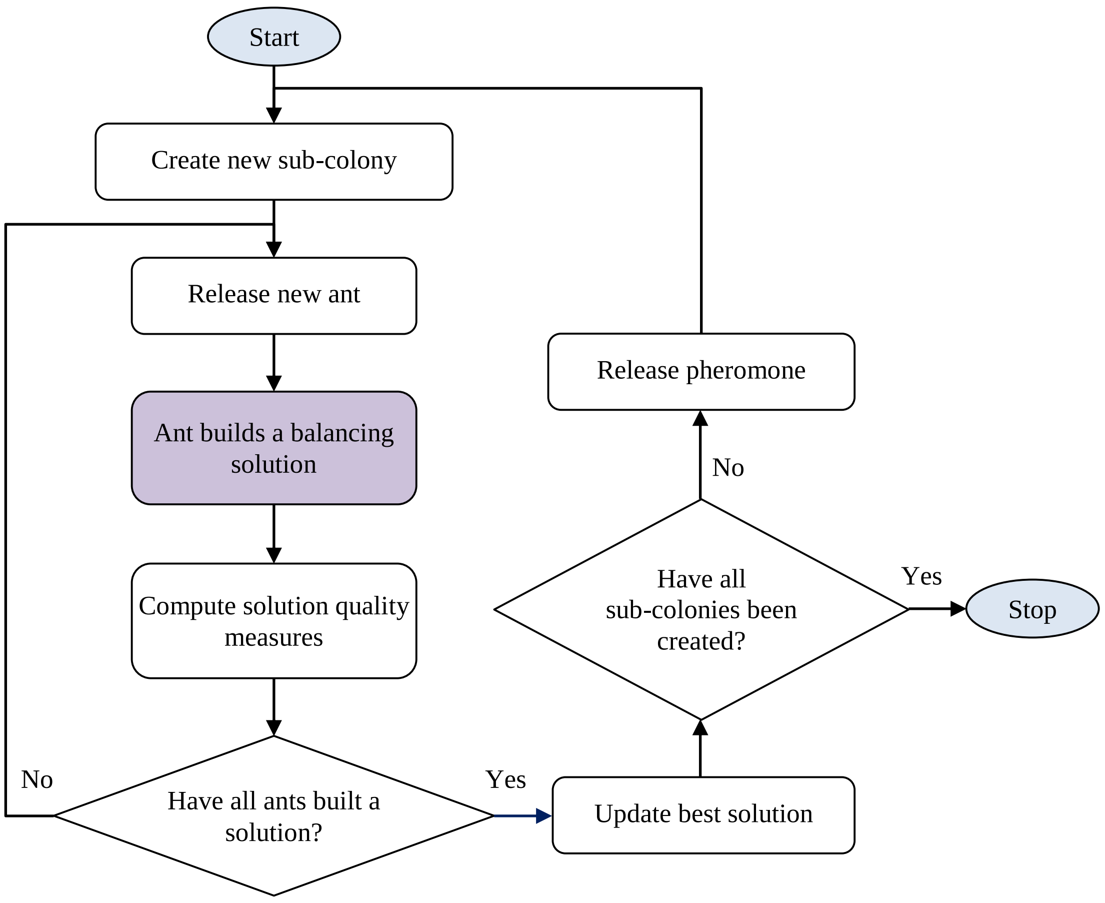

Start
Create new sub-colony
Release new ant
Ant builds a balancing solution
Compute solution quality measures
Have all ants built a solution?
No
Yes
Update best solution
Have all sub-colonies been created?
No
Yes
Release pheromone
Stop

Figure 3-17. Flowchart of ANTBAL proposed by Vilarinho and Simaria (2006)

The algorithm starts with creating a sub-colony with a pre-determined number of ants. A feasible balancing solution is obtained by each ant in the sub-colony by considering the assignment of tasks that satisfies precedence, zoning and capacity constraints. Then, the measure of solution's quality is computed for each feasible solution, according to the objective function considered (Vilarinho and Simaria, 2006). When all ants in the same sub-colony complete their tour, the pheromone is released on visited edges of the path drawn according to the quality of the solution obtained by each ant. Pheromone trails are built on the edges of the built balancing solution by each ant at the end of each sub-colony iteration. First, some amount of pheromone is evaporated in all paths; and then an amount of pheromone (where the amount is calculated in accordance with the quality of the solution) is deposited in the paths used to build the solution. Pheromone trails are kept in a task-task matrix and if task $j$ is performed immediately after task $i$, then the pheromone is released between tasks $i$ and $j$. The best solution is updated if a solution, which is better than the current best, is found and the procedure is repeated until all sub-colonies are completed within the ant colony (Vilarinho and Simaria, 2006).

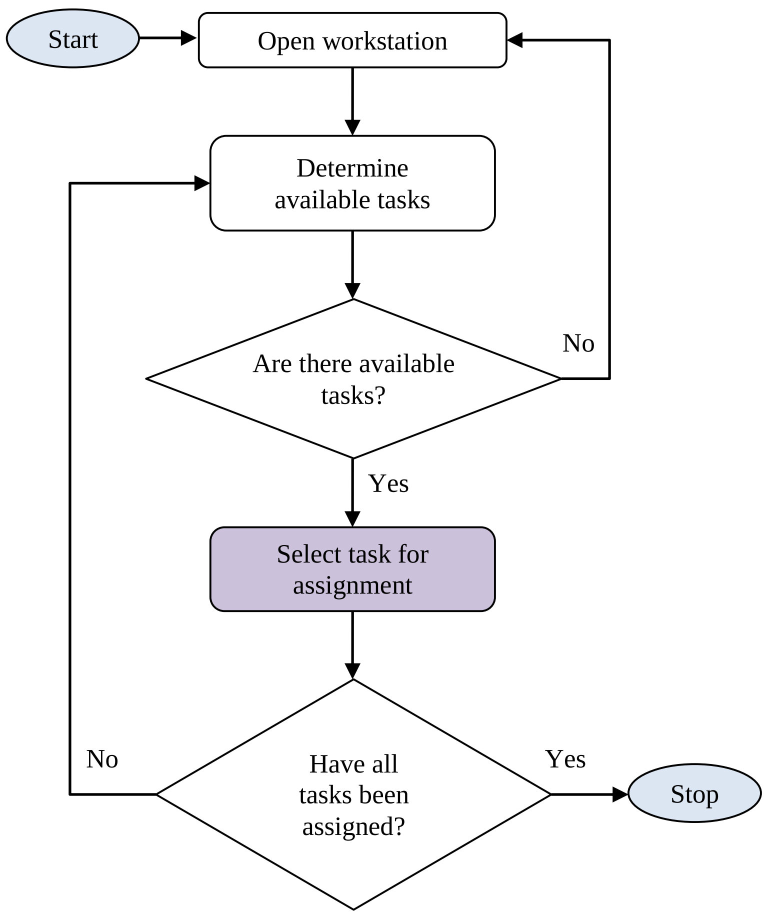

Start
Open workstation
Determine available tasks
Are there available tasks?
No
Yes
Select task for assignment
Have all tasks been assigned?
No
Yes
Stop

Figure 3-18. The procedure followed by an ant to build a feasible solution, adapted from Vilarinho and Simaria (2006)

The flowchart of the procedure carried out by an ant to build a feasible solution is given in Figure 3-18 (Vilarinho and Simaria, 2006). To process an ant, available tasks for assignment to the current workstation are determined by considering problem constraints. Then, a task is selected from the set of available tasks by an ant. If there is no capacity or there is no available task to be selected, a new workstation is opened. This cycle is repeated until all tasks are assigned. In the task selection procedure for assignment, the probability of a task being selected, from the set of available tasks, is a function of:

- the pheromone level between the last selected task and each available task
- the heuristic information provided for each available task.

The heuristic information is a priority rule that is randomly assigned to each ant. Some common priority rules for the assembly line balancing problem (*i.e.* maximum positional weight, maximum processing time –for all product models–, maximum average processing time, maximum number of direct successors, and maximum number of successors) are used in the research (Vilarinho and Simaria, 2006). Further information about ACO meta-heuristic used in this research will be given in the following chapters.

One of the main application areas of ACO algorithm is assembly line balancing problem, as mentioned. Previous researches that involve ACO techniques to solving various kinds of assembly line balancing problems are summarised in Table 3-5 and briefly extracted below. As could be seen from the table, the first technique that uses concepts derived from ACO to solve line balancing problem was implemented by McMullen and Tarasewich (2003). Then, ACO techniques have been applied to wide range of line balancing problems, from straight lines to parallel lines.

Many different performance measures were sought in these problems such as the number of workstations, cycle time, design cost, completion on time, workload smoothness, work-relatedness, and so on. However, still some types of assembly lines utilised in the industry, *i.e.* mixed-model parallel two-sided assembly lines have not been addressed by any researcher in the literature. In the study of McMullen and Tarasewich (2003), various ant system related techniques have been applied with three different objectives: *(i)* minimisation of cycle time, *(ii)* minimisation of cost, and *(iii)* completion on time. These heuristic procedures work by the selection of tasks to be added to the current workstations by artificial ants. Pheromone level determines the probability of a task being selected by an ant. Pheromone, a measure of each path's relative desirability, is deposited between the ant and candidate task. Working procedure of the algorithm includes:

- initialisation, and resetting all parameters
- determination of the need of new work center
- selection of tasks
- pheromone updating.

The global best solution is updated if a better solution is found, and this procedure is repeated until a predetermined number of iterations is exceeded.

Table 3-5. Summary of the literature review on ACO based approaches to solving assembly line balancing problems

| Research | Line Configuration | | | | Mixed-models | Parallel stations | Main obj. (min) | | | Additional constraints | | | | Additional features/keywords |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Straight | U-shaped | Parallel Lines | Two-sided | | | NS | C | O | Zoning | Positional | Synch. tasks | Space | |
| McMullen and Tarasewich (2003) | ● | | | | ● | ● | | ● | | | | | | Stochastic task times, design cost, completion on time |
| McMullen and Tarasewich (2006) | ● | | | | | ● | | | ● | | | | | Multiple objectives considered, *(i)* crew size, *(ii)* design cost, and *(iii)* probability of completing tasks on time used as the objective |
| Vilarinho and Simaria (2006) | ● | | | | ● | ● | ● | | | ● | | | | Workload smoothness, line length |
| Bautista and Pereira (2007) | ● | | | | | | ● | | | | | | ● | - |
| Zhang *et al*. (2007) | ● | | | | | | ● | | | | | | | Pheromone summation rules |
| Baykasoglu and Dereli (2008) | | | | ● | | | ● | | | ● | | | | Work-relatedness |
| Baykasoglu and Dereli (2009) | ● | ● | | | | | ● | | | | | | | - |
| Baykasoglu *et al*. (2009) | | | ● | | | | ● | | | | | | | - |
| Khaw and Ponnambalam (2009) | ● | ● | | | | | ● | | | | | | | Workload smoothness |
| Sabuncuoglu *et al*. (2009) | | ● | | | | | ● | | | | | | | - |
| Simaria and Vilarinho (2009) | | | | ● | ● | | ● | | | ● | | ● | | Workload smoothness |
| Chica *et al*. (2010) | ● | | | | | | ● | | | | | | ● | - |
| Chica *et al*. (2011) | ● | | | | | | ● | | ● | | | | ● | Multi-objective, labour cost, space cost |
| Fattahi *et al*. (2011) | ● | | | | | | ● | | | | | | | Multi-manned stations, stochastic mechanism help ants |
| Ozbakir *et al*. (2011) | | | ● | | | | ● | | | | | | | Bi-objective evaluation function |
| Sulaiman *et al*. (2011) | ● | | | | | | ● | | | | | | | Look forward ant |
| Yagmahan (2011) | ● | | | | ● | | | | | | | | | Workload smoothness |

NS: Number of workstations, C: Cycle time, O: Other

McMullen and Tarasewich (2006) solved multi-objective assembly line balancing problem by modifying previously proposed ant colony algorithm (McMullen and Tarasewich, 2003). They addressed various conflicting objectives: system utilisation, crew size, the probability of jobs being completed within a certain period and system design costs, which were thought to be needed to design a competitive assembly line. To obtain design solutions of the problem, an ACO algorithm has been developed. A composite function consisting of linear combinations of conflicting goals was constructed, and an efficient frontier was involved to address two or more objectives associated with this problem, simultaneously.

Other ACO based algorithms to solve SALBP were developed by Zhang *et al.* (2007), and Sulaiman *et al.* (2011). A new nonlinear function was used by Zhang *et al.* (2007) instead of minimising number of workstations, to solve SALBP. The objective function was maximised by transferring tasks from workstations with less workload to the ones that has more workload. They presented an ant algorithm that incorporates pheromone summation rules to increase the effectiveness of the algorithm. Pheromone was deposited between the task and its selected position. Local and global pheromone updating rules were adopted and positional weight mechanism was used to compute heuristic information about tasks.

A procedure that dynamically assigns the value of priority rule (as heuristic information) during the task selection phase was introduced by Sulaiman *et al.* (2011). They enhanced ACO algorithm with a *look forward ant* that seeks direct successors of the tasks during the task selection phase, and applied this approach to type-I SALBP. The pheromone trail was deposited between task and its position. A rank-based priority rule that provides heuristic information about candidate tasks was incorporated. The priority values vary between 1 and the total number of tasks, and are recomputed when the candidate tasks list is updated.

Mixed-model assembly line balancing problem was solved by a limited number of researchers using ACO techniques. Among those, the study of Vilarinho and Simaria (2006), which proposed an ACO algorithm called ANTBAL to solve mixed-model assembly line balancing problem, has already been described above.

Yagmahan (2011) presented an approach based on the ACO technique to address the mixed-model assembly line balancing problem as well. The proposed multi-objective

ACO algorithm sought solutions by considering both balance delay and the excursions among station times due to operation time variations. A two phased local search procedure was adopted to enhance the effectiveness of the ACO algorithm after local updating rule is completed. Initial pheromone level is obtained from modified ranked positional weight method (Helgeson and Birnie, 1961), which envisages joint precedence diagram for the given problem.

As noted in Section 2.2, Baykasoglu and Dereli (2008), and Simaria and Vilarinho (2009) solved two-sided assembly line balancing problem using ACO based techniques. The study carried out by Simaria and Vilarinho (2009) offered two-sided assembly line system to produce large-sized products. This research was also explained in detail in Section 3.2.3.

Different from conventional line balancing problems, Bautista and Pereira (2007) took 'space' into consideration as an additional constraint to reflect more realistic conditions in real production systems. They solved time and space constrained assembly line balancing problem with an ACO algorithm, which applies a diversification mechanism based on reinitiating pheromone information to avoid stagnation in unpromising areas of the solutions space. A station oriented framework was used for solution building. The pheromone trail could be used directly or accumulatively relating a task and station under construction. To build neighbourhood solutions, a local improvement procedure was also incorporated.

Then, Chica *et al*. (2010, 2011) proposed new algorithms to handle time and space constrained assembly line balancing problem. Chica *et al*. (2010) presented two new multi-objective constructive heuristics for the 1/3 (which means *time* and *space*) variant of the time and space constrained assembly line balancing problem. Proposed approaches were based on ACO technique and random greedy search method. The algorithms aimed at minimising number and area of stations in order to reflect a more realistic version of classical assembly line balancing problems. The first approach was a Pareto based multi-objective ACO algorithm which aims to find a set of best solutions, according to several conflicting objectives. The second one was a multi-objective random greedy search algorithm, based on the first stage of the greedy randomised adaptive search procedure. They also applied different configurations and parameter settings and compared them. The procedures of opening a new workstation and selection of a task were based on random priority rule to provide diversity. Two types of heuristic information values were employed; namely task operation time and required area.

Chica *et al*. (2011) handled time and space constrained assembly line balancing problem whilst entertaining the influence of user preferences. They proposed a multi-objective ACO algorithm incorporated with user preferences based on domain knowledge of a well-known automotive company. They also used six real scenarios around the world to include user preferences in the objective space. An evolutionary multi-objective optimisation based scheme was used, and two types of operational costs were considered: *(i)* labour cost, and *(ii)* space cost. While constructing the algorithm, a new mechanism was introduced to close a station, according to a probabilistic distribution. This mechanism provided more diverse solutions.

One of the first attempts to solve U-shaped assembly line balancing problem with an ACO based algorithm was carried out by Baykasoglu and Dereli  (2009). They addressed the so-called simple and U-shaped assembly line balancing problem with the objective of maximising the performance of the line (which is equivalent to minimising the number of workstations). The developed method integrates COMSOAL algorithm (Arcus, 1966) and ranked positional weight heuristic (Helgeson and Birnie, 1961) to enhance the efficiency of the applications.

Khaw and Ponnambalam (2009) presented a hybrid algorithm, multi-rule multi-objective ACO approach, to address straight and U-shaped assembly line balancing problem. They adopted the multi-rule multi-objective concept of Baykasoglu (2006) into ACO algorithm. So, proposed algorithm made use of 15 different task assignment rules. The main aim of the research was to optimise the line efficiency in conjunction with smoothness index.

Another ant colony algorithm with random search was proposed by Sabuncuoglu *et al*. (2009) to solve single-model U-type assembly line balancing problem. They applied secondary pheromone trail mechanism, not to loss information gained by pheromone accumulation after too many loops. Local and global pheromone trails were updated after each solution generated by each ant, and best solutions achieved by an ant, respectively.

Parallel line balancing problem was also considered to be solved with ACO algorithms developed by Baykasoglu *et al*. (2009), and Ozbakir *et al*. (2011). These researches were already examined in Section 2.3.

Fattahi *et al*. (2011) dealt with assembly line balancing problem with multi-manned workstations. They presented a mixed integer mathematical programming model which aims to minimise the total number of workers as a primary objective, and the number of opened multi-manned workstations, as a secondary objective. Additionally, a heuristic approach based on the ACO approach was presented to solve the medium and large-sized versions of this problem. The line manager determines the maximum number of workers that can be assigned to each workstation by considering required equipment, product size, and line design. In the proposed algorithm, a stochastic mechanism called 'temperature' has been built for the aim of reinforcing ants to escape from trapping local optima while searching. In other words, each colony has a temperature which helps ants to decide on accept or reject the workloads of poor quality solutions for a multi-manned workstation.

Aside from assembly line balancing problems, Chehade *et al*. (2008) used hybrid ACO algorithm to select machines for stations and to determine the capacity of buffers located between stations. Agrawal and Tiwari (2008) applied ACO to mixed-model U-shaped disassembly line balancing and sequencing problem.

## 3.4. Multi-agent Systems Applications on Assembly Line Balancing Problems

A multi-agent system is a distributed artificial intelligence system that includes multiple autonomous entities (Lim and Zhang, 2003). Multi-agent systems have a wide application domain from shop floor controls to air-traffic controls (Lim and Zhang, 2004, Lim and Zhang, 2012).

In the literature, several solution approaches were proposed to solve various kinds of assembly line balancing problems. However, the number of studies that apply multi-agent systems into assembly line balancing problems is extremely scarce. Only a few papers considered agent-based approaches to balance assembly lines.

Praca and Ramos (1999) proposed a multi-agent simulation based architecture to help the decision of distributing human resources in assembly lines. The proposed architecture consisted of four different kinds of agents, which communicate and cooperate with each other to reach a common goal. These agent types were *(i)* supervisor agents, *(ii)* production agents, *(iii)* resources agents, and *(iv)* an observer agent. Supervisor agents decide how many production agents are needed and distribute the operations between production agents. Production agent communicates to resources agent and if it is not possible to perform operations, production agent notifies supervisor agent to change the distribution. Observer agent is responsible for controlling the system and prepare a final report to the user.

Yokoyama *et al.* (2006) denoted a worker as an agent and developed a multi-agent system to solve SALBP. They incorporated tabu list and cooling control from tabu search and simulated annealing meta-heuristics, respectively. Then, Yokoyama *et al.* (2008) extended their previous work and proposed a multi-agent system to solve mixed-model assembly line balancing problems in a relatively short time period by distributed calculation. To have a balancing solution, agents exchange tasks between each other. An agent $(at)$ can only be communicating with its adjacent agents $(at - 1$ and $at + 1)$. If an agent receives only one proposal message, the request is always accepted. Nevertheless, if an agent receives two proposal messages, then agent selects one of them by calculating the variance of total task time of both proposals. Tabu list and cooling parameter were also utilised to provide diversity in solutions, and to avoid from local minima. The reason was that agents would always decide on the exchanges having minimum variance in total task time.

Liao *et al.* (2010) proposed a multi-agent based algorithm with two-level agent architecture. To ensure communication between machines agents, a tabu search algorithm was applied. The main objective was to minimise the number of workstations and the secondary objective was to obtain a smooth workload distribution among workstations. In the first level, a planning agent determines the ideal number of workstations for calculated cycle time based on customer demand. A balancing agent and multiple machine agents collaborate to balance the line, in the second level. A workstation presents a machine agent. If a feasible solution cannot be found by balancing agent, a message that states the need of increasing station number is sent to planning agent.

Yokoyama *et al.* (2010) modified the agent-based system proposed for SALBP in Yokoyama *et al.* (2008). They argued that the new multi-agent system was independent of the problems and the algorithmic parameters. Task exchange procedure was similar to the approach of Yokoyama *et al.* (2008). However, in the proposed approach, agents were unaware of the cycle time of production system but aware of the cycle time of immediately neighbour workers. A dynamic tabu list and a cooling parameter were also employed in this research to avoid local minima and cyclic searches. They investigated the effectiveness of the modified system by solving problems under different conditions.

A relatively different agent-based study was carried out by Kong *et al.* (2006). They proposed a new process planning method; an agent-based collaborative assembly process planning system that runs in an internet environment, to reduce the building time of an assembly line. The roles of the agents were described in the paper. The computer-based collaborative system encourages coordination and cooperation of two or more people to solve a problem. In the system, a *coordinator* is responsible for the smooth information flow between agents. Process planners are human experts who negotiate through a *negotiator*. To verify the proposed system, an internet-based application that allows geographically dispersed process planners to assign tasks on the same assembly line was experienced.

## 3.5. Chapter Summary

Since the conveyor belt system had been put into practice by Henry Ford and his colleagues in one of the leading automobile manufacturing companies, flow oriented production systems have been used in effective mass production systems of homogeneous standardised products, frequently. However, traditional assembly line configurations come up short in adapting to real-world conditions and responding to diversified market demands. So, new flexible line configurations are needed to model and solve real line balancing problems in the rapidly changing business environment. Additionally, cost efficiency should be considered while modelling such systems.

This chapter completed the survey of the literature, which was initialised in the previous chapter, on the sub-problems of mixed-model parallel two-sided assembly line balancing problem. Summary tables were composed to figure out the existing researches on concerned problem types along with ACO and multi-agent systems based approaches in the relevant literature. The main contributions have been searched and reported in tables. These tables represented the studies on divided sub-problems with proposed methodologies and their specific features. As can be understood from the tables, MPTALB/S problem has not been studied by any researcher in the literature so far. Indeed, no studies addressed balancing and sequencing problems simultaneously in the mixed-model parallel two-sided assembly line environment. That is why literature review of this research has been carried out by dividing the MPTALB/S problem into its sub-problems. The existing solution approaches for these sub-problems and key issues were investigated in detail. ACO and multi-agent system applications in assembly line balancing problems were also examined in this chapter.

Briefly, the conclusion that can be drawn from this literature review is that mixed-model parallel two-sided assembly line system is a new research domain. Although there are many studies on different types of line balancing problems, there is a gap in the literature about modelling and optimisation of mixed-model parallel two-sided assembly lines. To the best knowledge of the author, this thesis is the first study which addresses balancing and sequencing problems simultaneously in mixed-model parallel two-sided assembly lines.

The next chapter provides definition and detailed information on MPTALB/S problem and explores the main characteristics of the problem using illustrations and numerical examples.

# 4

# BALANCING AND SEQUENCING OF MIXED-MODEL PARALLEL TWO-SIDED ASSEMBLY LINES

**Contents**

## 4.1. Chapter Introduction

This chapter is considered to be the core of this thesis along with the next three chapters. Section 4.2 contextualises the need for this research by expressing the requirement of considering the line balancing problem together with the model sequencing problem. Section 4.3 describes the MPTALB/S problem and explains key benefits of the proposed mixed-model parallel two-sided assembly line layout along with its working principle. This is followed by Section 4.4 which identifies assumptions considered while describing the problem. An explanatory example is given in Section 4.5 to highlight the importance of constructing multi-line stations and necessity of taking the model-sequencing problem into consideration as well as the line balancing problem. The developed notation and mathematical model of the problem are provided with key concepts in Section 4.6 and the summary of the chapter is presented in Section 4.7.

## 4.2. Fundamentals of Simultaneous Balancing and Sequencing in Mixed-model Lines

Model sequencing is another type of most encountered problem in conjunction with the line balancing problem in mixed-model lines. Mixed-model sequencing problem is identifying sequence of order in which product models are produced on the line to meet customer demand by optimising a performance measure such as (Kara *et al*., 2007b):

- smoothing part usage
- minimising setup costs
- maximising workload smoothness across workstations.

Among the aforementioned goals, while the first is related to obtaining a uniform distribution of the usage rate of each part required by the assembly line in a just-in-time environment, the second goal aims at minimising cost of setups that may be needed while passing from one product model to another (Kara *et al*., 2007b). Setup is needed when there are considerable amount of differences between the product variants produced on the same line.

Although the first two goals have been considered in many studies, the third one has not received much attention by scholars. Maximising workload smoothness across stations minimises the deviation of operation times required by each product model. Please refer to Boysen *et al.* (2009) for a comprehensive literature review on mixed-model sequencing problem.

On mixed-model lines, while task assignment procedure is affected by the sequence of product models, the optimality of a product model sequence tightly depends on the assignment of tasks. So, there is a strong relationship between line balancing and model sequencing problems and these two problems are tightly interrelated with each other.

As will be shown in the following chapters, considering model sequencing procedure along with the line balancing problem also plays a critical role in the mixed-model parallel two-sided assembly line system proposed in this thesis. The reason is that the workloads and availability of the multi-line stations strictly depend on the product model mix being produced at these workstations. Moreover, due to the improved flexibility of producing more than one product model on a line, various combinations of the product models may exist on the lines at different times. Each of these different phases of the lines in terms of the combinations of product models across the lines is called production cycle. This will be explained in Section 4.5 in details with a numerical example.

Despite the fact above, line balancing and model sequencing problems have been considered separately in the majority of studies in the literature (Kim *et al.*, 2000a). To the best knowledge of the author, the studies that consider balancing and sequencing problems simultaneously in the literature can be divided into three groups in terms of the transport system utilised: *(i)* moving lines, *(ii)* paced lines, and *(iii)* un-paced lines. According to Merengo *et al.* (1999), the essential characteristic of a moving line that makes it different from a paced line, which is studied in this thesis, is that a transport system moves at a constant speed and transfers the items evenly distributed along the line. However, in a paced line, which is the basis of the mostly used type of conveyor systems in the automotive industry, the transport system moves periodically. Delivered items remain at the station during the cycle time, and are passed to the upstream (following) station. Different from paced lines, in un-paced lines, the operator pulls one unit from the downstream buffer located between the workstations and then moves to the upstream buffer after performing all required jobs on it. Table 4-1 exhibits the main contributions in the mixed-model assembly line balancing and sequencing literature together with methodologies/approaches and performance measures used.

Table 4-1. Overview on balancing and sequencing mixed-model lines research

| Research | Problem | Method/approach | Performance measure(s) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | **NS** | **C** | **LE** | **WLS** | **RI** | **WIP** | **PUR** | **TUW** | **CoS** | **BC** | **LBT** |
| Merengo *et al.* (1999) | B/S manual MALs | 3-phase methodology & simulation | ● | | | | ● | ● | ● | | | | |
| Kim *et al.* (2000a) | B/S MAL | Co-evolutionary algorithm | | | | | | | | ● | | | |
| Kim *et al.* (2000b) | B/S U-shaped MALs | Co-evolutionary algorithm | | | ● | | | | | | | | |
| Karabati and Sayin (2003) | B/S MAL | New mathematical model + heuristic | | ● | | | | | | | | | |
| Kim *et al.* (2006) | B/S U-shaped MAL | Endosymbiotic evolutionary algorithm | | | | ● | | | | | | | |
| Kara *et al.* (2007a) | B/S JIT U-shaped MAL | Simulated annealing | ● | | | ● | | | | | | | |
| Kara *et al.* (2007b) | B/S JIT U-shaped MAL | Simulated annealing | | | | ● | | | ● | | ● | | |
| Battini *et al.* (2009) | B/S MALs with finite buffer capacity | Branch and bound based step-by-step procedure | | | ● | | | | | | | ● | |
| Boysen *et al.* (2009) | MMS, car sequencing, level scheduling | Survey | | | | | | | | | | | |
| Hwang and Katayama (2010) | B/S straight and U-shaped MALs | Multi-objective evolutionary algorithm | ● | | | ● | | | | | | | |
| Ozcan *et al.* (2010a) | B/S parallel MALs | Simulated annealing | | | ● | ● | | | | | | | |
| Ozcan *et al.* (2011) | B/S stochastic U-shaped MAL | GA | | ● | | | | | | | | | |
| Mosadegh *et al.* (2012) | B/S MALs | Evolutionary strategies algorithm | | | | | | | | ● | | | ● |
| Hamzadayi and Yildiz (2012) | B/S U-shaped MALs | GA-based approach | ● | | | ● | | | | | | | |

B/S: Balancing and sequencing, MAL: Mixed-model line, MMS: Mixed-model sequencing, JIT: Just-in-time, NS: Number of stations, C: Cycle time, LE: Line efficiency (minimising idle time), WLS: Workload smoothness (=absolute deviations of workloads across workstations), RI: Rate of incomplete jobs, WIP: Work in process, PUR: Part usage rate, TUW: Total utility work, CoS: Cost of setups, BC: Buffer capacity, LBT: Last best time, GA: Genetic algorithm.

The studies of Merengo *et al.* (1999), Kim *et al.* (2000a), and Mosadegh *et al.* (2012) assumes a continuously *moving* transport line with a constant speed. Merengo *et al.* (1999) considered balancing and sequencing problems in manual mixed-model moving assembly lines. Minimising the rate of incomplete jobs, and reducing work in progress were common objectives for both balancing and sequencing algorithms. Moreover, the 3-phase methodology proposed considered minimising the number of stations as an additional performance measure while sequencing algorithm also provided a uniform parts usage.

Kim *et al.* (2000a) proposed a co-evolutionary algorithm to minimise the total amount of utility work in simultaneous balancing and sequencing of mixed-model assembly lines. The deviation of workload was minimised in balancing problem and the solution of the balancing problem was used as a parameter in the sequencing problem, where total utility work was considered as a performance parameter.

Mosadegh *et al.* (2012) proposed a mixed integer linear programme and an evolutionary strategies algorithm to tackle simultaneous balancing and sequencing of mixed-model assembly lines where uncompleted tasks are passed to the utility workers. Total utility work was minimised in a predetermined constant number of workstations environment, and obtained results from the proposed algorithm have been compared with the results of existing methodologies in the literature. However, one disadvantage of the algorithm was that the flexibility of assigning common tasks to different workstations would lead to extra cost of establishing special equipment.

The majority of studies in the literature, which tackle balancing and sequencing problems simultaneously, focus on *paced* mixed-model U-lines. Because, in the mixed-model U-lines, the tasks assigned to a station can be performed on the front and back of the line. In this situation, the product models produced at a station may vary in each cycle depending on the product model sequence. In other words, various product model mixes may be encountered in the stations that utilised on both front and back sides of the line. Hence, the workload of the station is strictly interrelated with the line balance and the product model sequence.

In one of the preliminary works, Kim *et al.* (2000b) applied the co-evolutionary algorithm for balancing and sequencing of mixed-model U-lines. The algorithm adopted strategies of localised interactions and steady-state reproduction and employed methods to select environmental individuals and evaluate fitness to increase diversity in the population while providing more efficient solutions. The performance measure taken into consideration in that study was workload smoothness among stations.

Karabati and Sayin (2003) addressed line balancing problem in a mixed-model sequencing environment. An exact representation of the interaction between task assignments and product model sequence has been represented with a heuristic procedure. The objective of the proposed mathematical model and the heuristic procedure was to minimise the maximum sub-cycle time for a given product model sequence.

The first study which aimed to minimise the number of stations ensuring a smooth workload across stations for just-in-time mixed-model U-line balancing and sequencing problem was carried out by Kara *et al*. (2007a). A mathematical model was proposed and a simulated annealing algorithm was developed to optimise these objectives. The ANOVA (Analysis of Variance) test that was performed within the scope of the study showed that the total number of required stations and workload smoothness index are significantly influenced by the length of the product model sequence.

In another work, Kara *et al*. (2007b) dealt with balancing and sequencing problems simultaneously for mixed-model just-in-time U-lines considering multiple objectives. The proposed multi-objective approach aimed at minimising absolute deviations of workloads across workstations, the cost of setups, and part usage rate. The proposed simulated annealing algorithm endeavoured to optimise balance dependent performance measure (absolute deviations of workloads across workstations), and sequence dependent performance measures (the cost of setups and part usage rate) in a timely manner.

Hwang and Katayama (2010) have also dealt with the line balancing and model sequencing problems in straight and U-shaped mixed-model assembly lines. For this purpose a multi-objective evolutionary algorithm with a newly developed priority based chromosome (in multi-structure), which represents not only task sequence but also sequencing of product models, was proposed. The two major performance measures of the proposed method were the number of workstations (or line efficiency) and workload variance. The results of experiments showed that, due to the increased possibilities of task combinations in U-shaped lines, results obtained for U-shaped lines outperform the results obtained for simple assembly lines.

Ozcan *et al*. (2011) introduced and characterised stochastic mixed-model U-line balancing and sequencing problem and proposed a GA approach to coping with the problem. The objective was maximising the production rate by minimising cycle time for a given number of operators (type-II).

Hamzadayi and Yildiz (2012) proposed a GA-based approach to tackling balancing and sequencing problems in mixed-model U-lines with the flexibility of constructing parallel workstations. The primary goal of the newly proposed fitness function (with simulated annealing based fitness evaluation approach) was to minimise the total number of workstations while seeking a balanced workload between/within stations as a secondary goal. It was proven in their study that product model sequence in which different product models are produced affects the objective function value and cannot be set independently of the line balance.

It can be comprehended from this review that in many cases in the literature, balancing and sequencing problems have been considered together for mixed-model U-lines, but not for mixed-model parallel two-sided assembly lines. As it has already been extracted in Section 3.2.2, the study of Ozcan *et al.* (2010a) is unique in the literature which integrates balancing and sequencing issues in parallel mixed-model lines (where the lines are considered as one-sided, not two-sided). No more information about this study will be given here since it has already been explored in details above.

As mentioned before, *un-paced* lines have also been of interest by researchers for simultaneous balancing and sequencing. A genetic approach was proposed by Kim *et al.* (2006) for mixed-model U-line balancing and sequencing problem. This study proposed an endosymbiotic evolutionary algorithm that constructs two populations, balancing population and sequencing population. In symbiotic algorithms, the individuals (called symbionts) in these populations represent task assignments and product model sequences, respectively, and are part of the entire solution. However, in their study, a new population (called balancing-sequencing population) was maintained and each endosymbiont in this population corresponded to an entire solution, which was a combination of task assignment and product model sequence. The optimal balanced workload among workstations was sought for asynchronous mixed-model U-lines.

Balancing and sequencing problem has also been adapted to un-paced lines which include buffers between workstations. Battini *et al*. (2009) proposed a branch and bound based step-by-step methodology that aimed at minimising both idle time and overload time across workstations while reducing buffer capacity requirements. The applicative case carried out within the scope of the study demonstrated the significance of considering sequencing issue in reducing downtime caused by a blockage in the buffers.

## 4.3. Definition of MPTALB/S Problem

In today's highly competitive business environment, mixed-model assembly lines provide more flexibility and capability of responding to different market demands to satisfy customised customer demands on time and to reach global markets. However, companies need to construct their production systems in an intelligent way to deal with undesirable costs caused by customisation of products.

With the solution of producing more than one product model on each adjacent line of parallel two-sided lines, a new competitive line system called *mixed-model parallel two-sided assembly lines* can be obtained. The problem of balancing these lines can be called *mixed-model parallel two-sided assembly line balancing problem*.

The mixed-model parallel two-sided assembly line balancing problem is balancing more than one mixed-model two-sided assembly line constructed in parallel to each other. The main objective is allocating tasks to the workstations optimally by considering technological priorities, capacity constraints and some other constraints like zoning or positional constraints. As will be explained in this chapter, with the integration of simultaneous model sequencing procedure with mixed-model parallel two-sided assembly line balancing problem, the problem becomes more complex to solve and turns into *MPTALB/S problem.*

The idea of constructing mixed-model parallel two-sided assembly lines is a completely new topic. It provides the flexibility of producing similar large-sized product models on parallel lines. This new type of configuration carries the combined practical advantages of mixed-model assembly lines, parallel assembly lines and two-sided assembly lines. These advantages include but are not limited to:

- shorter line lengths than traditional assembly lines
- shared use of common tools
- the flexibility of producing different product models with different throughput rates
- less material handling costs and operator movement requirements
- improved line efficiency with reduced operator requirement
- increased motivation of operators due to operation enrichment at combined workstations between two lines

- increased skill levels of operators
- improved communication skills among operators.

The precedence relationships among tasks should be considered carefully since tasks, which have precedence relationships with each other and are performed on both sides of each line, must be assigned with the consideration of completion time of previously assigned tasks. Let us consider $P_{1A9}$ as the set of predecessors of task 9 on Line I for product model A. If the precedence relationships among tasks are assumed as task $4 \in P_{1A9}$ and task $8 \in P_{1A9}$; task 9 can be initialised after the completion of tasks 4 and 8, which may be performed on the other side of the line. As mentioned earlier this phenomenon, which is called *interference* in the literature, must be handled carefully as the violation of this rule yields infeasible solutions.

The workstations can be utilised either on only one or on both adjacent two-sided lines. The common stations constructed for both adjacent lines are called *'multi-line stations'*, as mentioned by Battaïa and Dolgui (2013) for traditional parallel lines. A similar version of this structure, split workplaces, has been used by Scholl and Boysen (2009) in defining common stations on parallel assembly lines as given in Section 2.3. The utilisation of multi-line stations is one of the basic advantages of parallel assembly lines since multi-line stations help minimise the total number of required operators and thus minimise idle times. Figure 4-1 shows a typical configuration of two adjacent mixed-model parallel two-sided assembly lines with regular and multi-line workstations.

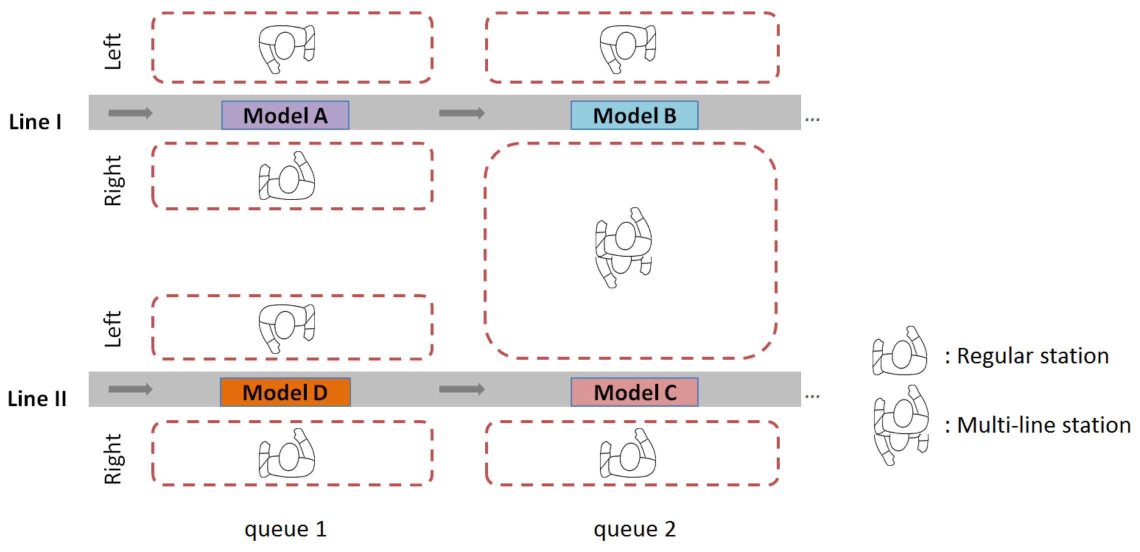


Figure 4-1. Representation of regular stations and multi-line stations on mixed-model parallel two-sided assembly lines

As can be seen from the figure, seven operators are allocated to perform tasks for all product models (A, B, C and D) in two queues. The operator allocated at the multi-line station, which is utilised between two adjacent lines in queue 2, works on both right side of Line I and left side of Line II.

More than one different product model, $m_{hj}$ $(j = 1, \dots, M_h)$, is produced on each two-sided assembly line, represented with $L_h$ $(h = 1, \dots, H)$. In the example given in Figure 4-1, product models A and B are assembled on Line I while C and D are assembled on Line II. Each product model has its own set of tasks, $t_{hji}$ $(i = 1, \dots, T_{hj})$, performed according to predefined precedence relationships. $P_{hji}$ represents the set of predecessor tasks of task $t_{hji}$ for product model $m_{hj}$ on line $L_h$. Each task $(t_{hji})$ for product model $m_{hj}$ on line $L_h$ requires a certain amount of processing time $(pt_{hji})$ to be processed and each line consists of a series of workstations, $W_{hkx}$ $(k = 1, \dots, K_h;\ x = 0, 1)$ ; where $x$ is a binary variable and '0' and '1' symbolise left side and right side of the line, respectively.

Another advantage of establishing mixed-model parallel two-sided lines is the opportunity of having different throughput rates for the lines. In other words, the cycle time $(C_h)$ of each line may be different from each other. $C_h$ is calculated according to demand over the planning horizon for each line:

$$C_h = \frac{P}{\sum_{j=1}^{M_h} D_{hj}}; \qquad h = 1, \dots, H; \tag{4.1}$$

where $D_{hj}$ represents the demand for product model $m_{hj}$ on line $L_h$ over a planning period $(P)$.

As explained, parallel two-sided assembly lines consist of a number of two-sided serial assembly lines arranged in a parallel form. It is known that running parallel lines under different cycle time conditions (and so different throughput rates) improves the flexibility of the overall assembly system. On the other hand, this situation yields a more sophisticated balancing and model sequencing problem. To tackle the complex task assignment procedure affected by the product model changes, a common cycle time should be used when different cycle times are subject to balancing and sequencing procedure. For this aim, *least common multiple* (LCM) of cycle times (Gökçen *et al.*, 2006) is adopted as common cycle time and task times are normalised according to the ratio of original cycle time to common cycle time. Common cycle time of two lines with different cycle times is calculated as follows (Gökçen et al., 2006):

- LCM of the cycle times is found.
- The integers $ld_1$ and $ld_2$ are calculated via dividing the LCM value by the cycle times of Line I and Line II ($C_1$ and $C_2$), respectively.
- Task times of the product models produced on Line I and Line II are multiplied by $ld_1$ and $ld_2$, separately.
- LCM is determined as the common cycle time ($C$) of the lines and the lines are balanced together.

The product model sequences of lines are important in determining the available times of operators that are allocated to multi-line stations, as the availability of an operator allocated between two adjacent lines depends on the sequence of product models being assembled on the lines. This issue will be exemplified in the following subsections.

The *minimum part set (MPS)* principle (Bard *et al*., 1992), which is widely accepted in mixed-model assembly line domain, is used in this study to consider the model sequencing problem integrated with the line balancing problem (Ozcan *et al*., 2010a). The $MPS_h$ (or $d_h$) is a vector which represents the smallest set having the same proportions of different product models as the demands. In other words, it is a vector which represents the mix of product models on line $L_h$, such that $d_h = (d_{h1}, \dots, d_{hM_h})$, where $d_{hj}$ is calculated as follows:

$$d_{hj} = \frac{D_{hj}}{cd_h}; \quad j = 1, \dots, M_h; \quad h = 1, \dots, H. \tag{4.2}$$

where $cd_h$ $(h = 1, \dots, H)$ is the greatest common divisor of the demands of the product models assembled on the same line $(L_h)$. $D_{hj}$ $(j = 1, \dots, M_h;\ h = 1, \dots, H)$ is the demand for product model $m_{hj}$ on line $L_h$ in a planning horizon. Obviously, the total demand is met by $cd_h$ times repetition of producing the $MPS_h$.

$MS_h$ represents the product model sequence of line $L_h$ which is independent from the sequence of other lines. $S_h$ is the length of $MS_h$ and is calculated as follows:

$$S_h = \sum_{j=1}^{M_h} d_{hj}; \quad h = 1, \dots, H. \tag{4.3}$$

For example, let us assume two product models, A and B, are assembled on the same line ($L_1$) and the demands for product models are assumed 10 and 20 units, respectively ($D_{1A} = 10,\ D_{1B} = 20$). The greatest common divisor of the demands (10 and 20) is calculated as 10 ($cd_h = 10$) and the minimum part set on $L_1$ is determined as $d_1 = (D_{1A}/cd_1, D_{1B}/cd_1) = (1,2)$. Thus, product model A is represented one time in the product model sequence of $L_1$, $MS_1$, while B is represented two times. $MS_1$ can be any combination of product models A and B in terms of the minimum part set, *e.g.* $MS_1 = ABB$, $MS_1 = BAB$ or $MS_1 = BBA$, and $S_1 = 1 + 2 = 3$.

Given a determined product model sequence, the number of different product model combinations that can be seen in a multi-line station depends on the lengths of the product model sequences on the lines ($S_h$, where $h = 1, \ldots, H$). This also regulates how many different production cycles (where a production cycle is represented with $\varphi$ and $\varphi = 1, \ldots, \phi$) the system should be split into. As will be illustrated comprehensively through a numerical example in Section 4.5, a production cycle is a different phase of the line system in terms of the configuration of product models being assembled in the workstations. For example, let us assume product models B and D are assembled in a multi-line station at a specific time. When the tasks are completed on these product models, it is possible that a new combination of product models (*e.g.* B and C) will be assembled in the same multi-line station, depending on the product model sequence previously determined. Each of these different product model combinations correspond to a different production cycle. However, these combinations will repeat in a cyclical manner. $\phi$ represents the maximum number of production cycles ($\phi = MS_{max}$) and is calculated as follows (Ozcan *et al.*, 2010a):

$$MS_{max} = \max\{S_h \times S_{(h+1)}\}; \qquad h = 1, \ldots, H-1\,. \tag{4.4}$$

Total number of possible product model sequences for a mixed-model assembly line is computed using the equation (Manavizadeh *et al.*, 2013b):

$$TS_h = \frac{\left(\sum_{j=1}^{M_h} d_{hj}\right)!}{\prod_{j=1}^{M_h}\left(d_{hj}!\right)}\,. \tag{4.5}$$

When two mixed-model lines are taken into account (as in the numerical examples and experimental tests presented in the forthcoming sections and/or chapters), the number of sequences emerging for the system could be computed by multiplying the total number of sequences belonging to the lines $(TS_1 \times TS_2)$. An example of these calculations will be provided with a numerical example in Section 4.5.

## 4.4. Assumptions

The assumptions considered in this thesis are as follows:

- More than one similar product model $(j = 1, \dots, M_h)$ is assembled on each of the two parallel two-sided assembly lines.
- Task times $(pt_{hji})$ of each product model are known and deterministic.
- The cycle time of a line is calculated according to demand over the planning horizon and can be different for different lines.
- Demand is known and deterministic for product models assembled on each line.
- Each product model has its own precedence relationships diagram, which is known.
- Common tasks between similar product models must be allocated to the same workstation. Some tasks may have different processing times for different product models. Sometimes, it is possible that the processing time of a task may equal to zero for specific product models.
- Tasks can be assigned to only a predetermined side (Left-L or Right-R) or either (E) side.
- Each task must be assigned to exactly one workstation $(W_{hkx})$; in other words tasks cannot be split into more than one workstation.
- The sum of all task times assigned to a workstation constitutes its workload, and workload of a workstation cannot exceed the predetermined cycle time of the relevant line.
- A task can only be assigned if all of its predecessors $(P_{hji})$ have been completed. That can be achieved in two alternative ways:
  - all predecessors are completed before the current queue, or
  - if some of the predecessor tasks are assigned to the current queue, then all predecessors are completed before the initialisation of the task in the same queue.
- Operators are multi-skilled and can work on any side of a line.

- Only one operator is assigned to a workstation.
- Operator travel times are ignored.
- No work in process inventory is allowed.
- Lines are paced, and starting and finishing times are same for all lines.

Relaxation of any of these assumptions may lead to an increased balancing solution which is more flexible and efficient in terms of idle times. For example, common tasks can be assigned to different workstations with considering separate precedence diagrams instead of a combined precedence diagram for different product models on the same line. However, assigning common tasks for different product models to different workstations may cause additional equipment costs. Dynamic demand is also another challenging point which manufacturers face with in real world applications. Though, it is obvious that such a relaxation makes the problem, which is already very complex, even harder to solve.

## 4.5. An Explanatory Example for MPTALB/S Problem

In this section, an illustrative example is provided to illustrate different production cycles and explain the MPTALB/S problem in detail. Another reason to give this example is to show how line balance and sequence of the product models affect the workload of a station in a cycle, caused by different product model mixes which may exist at multi-line stations utilised on two adjacent lines.

Let us assume that there is a line system, which consists of two mixed-model parallel two-sided assembly lines, as depicted in Figure 4-2. As can be seen from the figure, two product models (A and B) are executed on Line I while remaining two product models (C and D) are produced on Line II, simultaneously. Eleven workstations are utilised as illustrated in the figure, where one of the operators performs on both adjacent lines. Operator 6 performs tasks on Line I and on Line II consecutively.

The demands are considered as 10, 30, 20, and 20 for the product models A, B, C and D, respectively $(D_{1A} = 10, D_{1B} = 30, D_{2C} = 20 \text{ and } D_{2D} = 20)$ over the specified planning horizon, 480 time units. The cycle times of the lines are calculated easily using Equation (4.1) above $(C_1 = C_2 = 480\ \mathit{time\ units}/40\ \mathit{items} = 12\ \ \mathit{time\ unit}/\mathit{item})$.

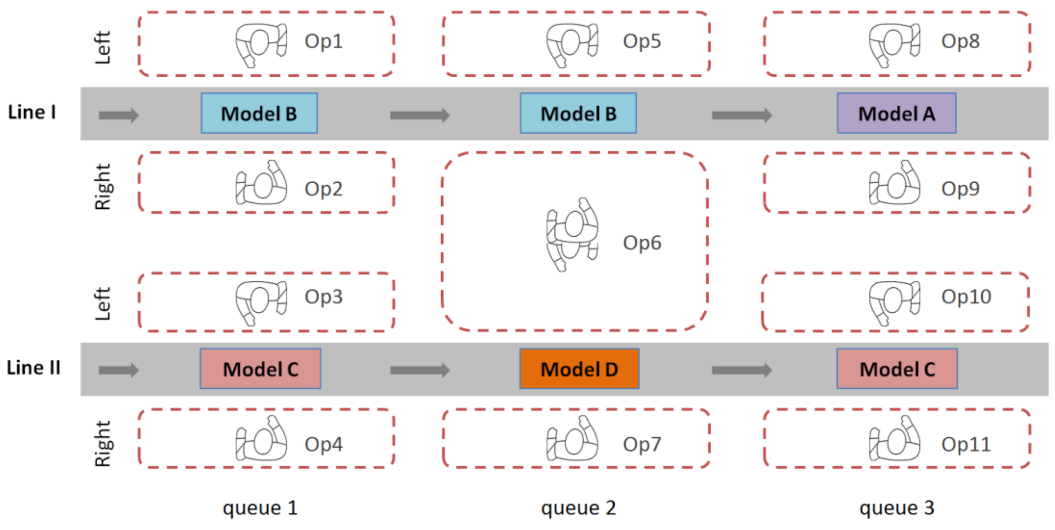


Figure 4-2. A schematic view of mixed-model parallel two-sided assembly lines

As described above, $MPS$ of each line $(MPS_h)$ is calculated by dividing total demands (10, 30, 20 and 20) of product models (A, B, C and D) by the greatest common divisor of these demands for each line. While the greatest common divisor $(cd_1)$ of $D_{1A}$ and $D_{1B}$ (10 and 30) is 10 for Line I, $cd_2$ is calculated as 20 for Line II. So, the product model mix of Line I $(MPS_1 = d_1)$ can be expressed as $d_1 = (D_{1A}/cd_1,\ D_{1B}/cd_1) = (1,3)$. Similarly, the product model mix of Line II is obtained as $d_2 = (1,1)$. Consequently, the total number of product models on the lines are: $S_1 = 4$ for $L_1$ and $S_2 = 2$ for $L_2$.

If the product model sequences are considered as $MS_1 = ABBB$ and $MS_2 = CD$ for Line I and Line II, respectively, possible product model mixes of the given example can be represented as in Table 4-2. Three different product model mixes appear at multi-line station, station 6, and same combinations are repeated by cycle 5. Therefore, there exist four different product model mixes for the sequence of product models on two adjacent lines. This situation can be illustrated as in Figure 4-3 for four production cycles.

Based on this illustration, it is obvious that product model combinations will change in workstation 6 in case of considering different product model mixes on the lines rather than $MS_1 = ABBB$ and $MS_2 = CD$. Accordingly, workload and availability of the operator who performs at this station will be affected by that change. Consequently, the model sequencing problem on the lines must also be taken into account simultaneously with the line balancing problem.

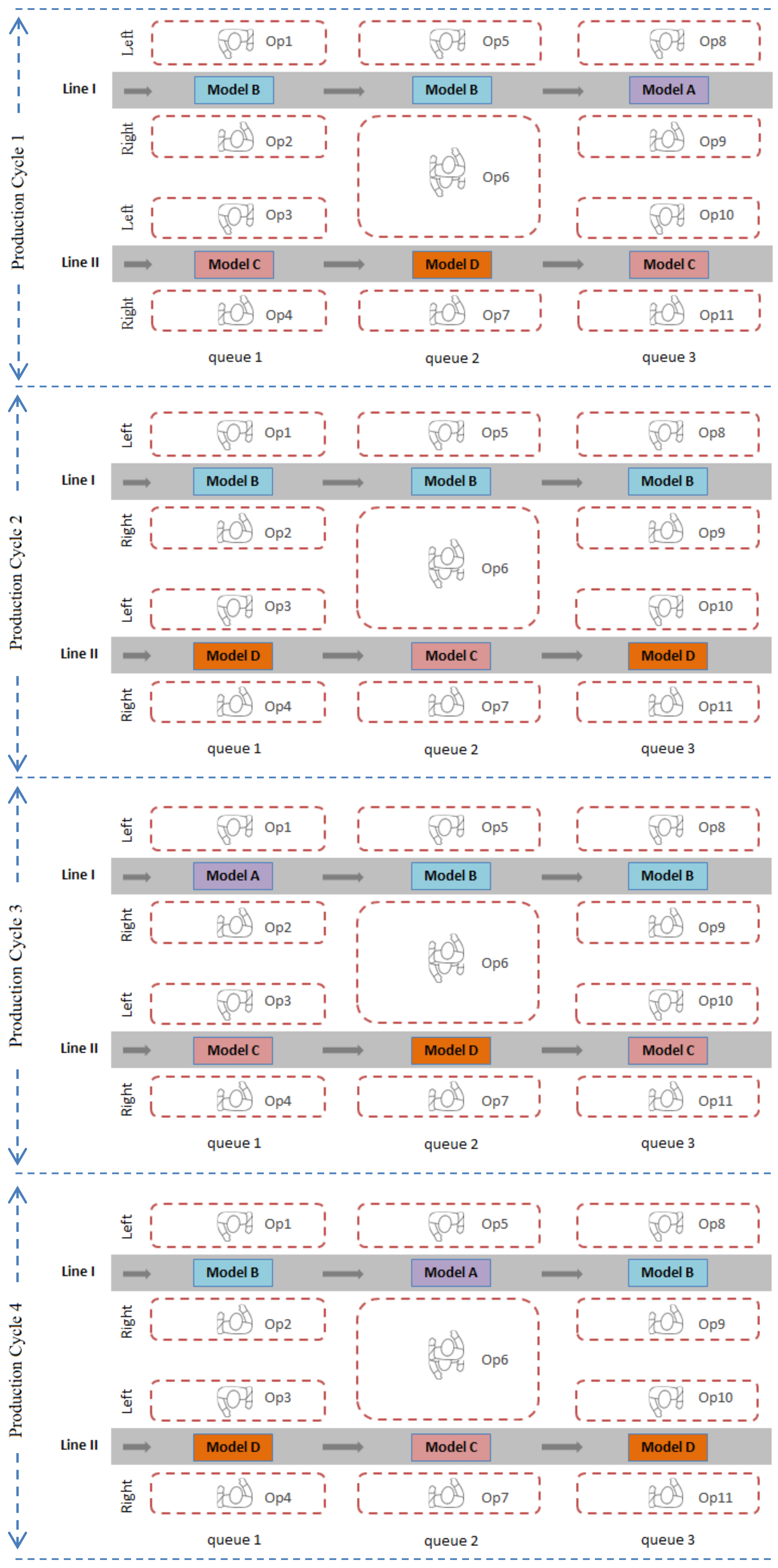


Figure 4-3. Product model mixes of the problem in the case $MS_1 = ABBB$ and $MS_2 = CD$

Table 4-2. Possible mixes of product models for given example ($MS_1 = ABBB$ and $MS_2 = CD$)

| Station no | 1 | 2 | 3 | 4 | 5 | 6 | | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cycle/Line | $\alpha_{1,1}^{\varphi}$ | $\alpha_{1,2}^{\varphi}$ | $\alpha_{2,3}^{\varphi}$ | $\alpha_{2,4}^{\varphi}$ | $\alpha_{1,5}^{\varphi}$ | $\alpha_{1,6}^{\varphi}$ | $\alpha_{2,6}^{\varphi}$ | $\alpha_{2,7}^{\varphi}$ | $\alpha_{1,8}^{\varphi}$ | $\alpha_{1,9}^{\varphi}$ | $\alpha_{2,10}^{\varphi}$ | $\alpha_{2,11}^{\varphi}$ |
| 1 | B | B | C | C | B | $\underline{B}$ | $\underline{D}$ | D | A | A | C | C |
| 2 | B | B | D | D | B | $\underline{B}$ | $\underline{C}$ | C | B | B | D | D |
| 3 | A | A | C | C | B | B | D | D | B | B | C | C |
| 4 | B | B | D | D | A | $\underline{A}$ | $\underline{C}$ | C | B | B | D | D |
| 5 | B | B | C | C | B | B | D | D | A | A | C | C |
| 6 | B | B | D | D | B | B | C | C | B | B | D | D |
| 7 | A | A | C | C | B | B | D | D | B | B | C | C |
| 8 | B | B | D | D | A | A | C | C | B | B | D | D |

$\alpha_{h,k}^{\varphi}$: The product model that is produced on line $L_h$ at station index $k$ in production cycle $\varphi$.

If the product model sequences are considered as $MS_1 = BBAB$ and $MS_2 = DC$, then all possible product model mixes that may appear on the lines can be represented as in Table 4-3. As seen, the same pattern repeats in every four cycle. Furthermore, Figure 4-4 simulates the possible production cycles based on the new product model sequence considered.

Table 4-3. Possible mixes of product models for the new combination ($MS_1 = BBAB$ and $MS_2 = DC$)

| Station no | 1 | 2 | 3 | 4 | 5 | 6 | | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cycle/Line | $\alpha_{1,1}^{\varphi}$ | $\alpha_{1,2}^{\varphi}$ | $\alpha_{2,3}^{\varphi}$ | $\alpha_{2,4}^{\varphi}$ | $\alpha_{1,5}^{\varphi}$ | $\alpha_{1,6}^{\varphi}$ | $\alpha_{2,6}^{\varphi}$ | $\alpha_{2,7}^{\varphi}$ | $\alpha_{1,8}^{\varphi}$ | $\alpha_{1,9}^{\varphi}$ | $\alpha_{2,10}^{\varphi}$ | $\alpha_{2,11}^{\varphi}$ |
| 1 | A | A | D | D | B | $\underline{B}$ | $\underline{C}$ | C | B | B | D | D |
| 2 | B | B | C | C | A | $\underline{A}$ | $\underline{D}$ | D | B | B | C | C |
| 3 | B | B | D | D | B | B | C | C | A | A | D | D |
| 4 | B | B | C | C | B | $\underline{B}$ | $\underline{D}$ | D | B | B | C | C |
| 5 | A | A | D | D | B | B | C | C | B | B | D | D |
| 6 | B | B | C | C | A | A | D | D | B | B | C | C |
| 7 | B | B | D | D | B | B | C | C | A | A | D | D |
| 8 | B | B | C | C | B | B | D | D | B | B | C | C |

$\alpha_{h,k}^{\varphi}$: The product model that is produced on line $L_h$ at station index $k$ in production cycle $\varphi$.

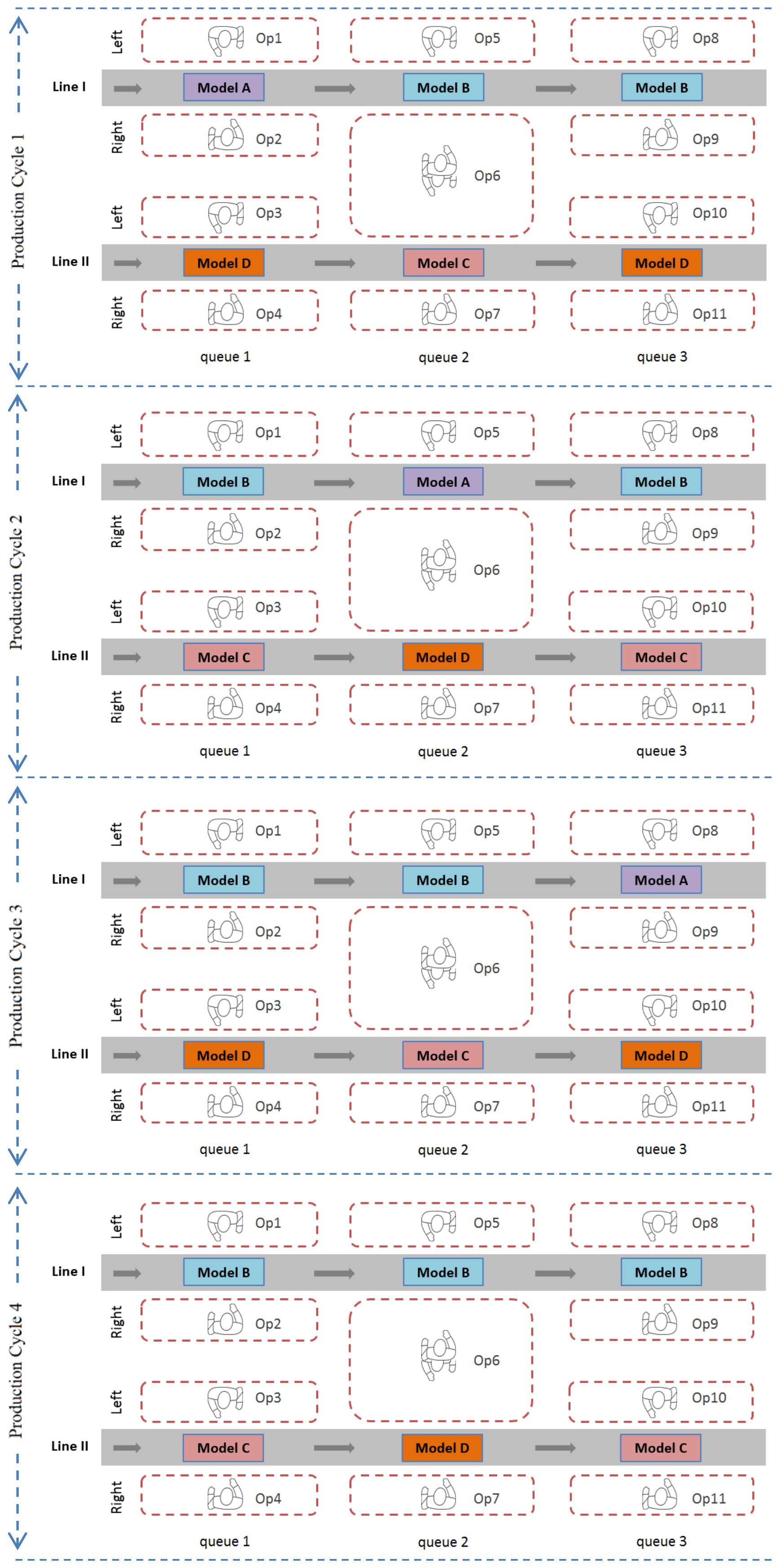


Figure 4-4. Product model mixes of the example in the case $MS_1 = BBAB$ and $MS_2 = DC$

## 4.6. The Mathematical Model of MPTALB/S Problem

The problem stated above is formulated as a mixed integer programming model that also takes into account product model sequences with the objectives of minimising weighted idle times of the lines (and so number of utilised workstations), minimising length of the lines, while maximising workload smoothness.

### 4.6.1. Nomenclature

The notation introduced in this chapter and used in the mathematical model is summarised below. Please note that some of the notations/parameters used in this study are similar with the study of Ozcan *et al.* (2010a) to provide a coherent context to the readers. In the following chapters, the additional notation will be introduced as the need arises.

#### 4.6.1.1. Indices

$L_h$ : The $h^{th}$ line $(h = 1, \dots, H)$,

$m_{hj}$ : The $j^{th}$ product model on line $L_h$ $(j = 1, \dots, M_h)$, where $M_h$ is the number of product models made on line $L_h$,

$t_{hji}$ : The $i^{th}$ task for product model $m_{hj}$ on line $L_h$ $(i = 1, \dots, T_{hj})$, where $T_{hj}$ is total number of tasks for product model $m_{hj}$ on line $L_h$,

$W_{hkx}$ : The $k^{th}$ workstation on line $L_h$ $(k = 1, \dots, K_h;\ x = 0, 1)$, where $K_h$ is total number of workstations on line $L_h$,

$x$ : Side of the line, $x = \begin{cases} 0 & \text{indicates the left side of the relevant line} \\ 1 & \text{indicates the right side of the relevant line} \end{cases}$,

$\varphi$ : Production cycle $(\varphi = 1, \dots, \phi)$, where $\phi = max\{S_h \times S_{(h+1)}\}$, $h = 1, \dots, H - 1$. The definition of $S_h$ is given below.

#### 4.6.1.2. Parameters

$pt_{hji}$ : Processing time of task $t_{hji}$ of product model $m_{hj}$ on line $L_h$,

$P$ : A pre-specified planning period,

$P_{hji}$ : Set of predecessors of task $t_{hji}$ for product model $m_{hj}$ on line $L_h$,

$D_{hj}$ : Demand, over the planning period, for product model $m_{hj}$ produced on line $L_h$,

$C_h$ : Cycle time of line $L_h$; $C_h = \frac{P}{\sum_{j=1}^{M_h} D_{hj}}$,

$C$ : Common cycle time for all lines ($C = LCM(C_1, \dots, C_H);\ h = 1, \dots, H$),

$cd_h$ : Greatest common divisor of product model demands ($D_{hj}$) for line $L_h$,

$d_{hj}$ : Normalised demand for product model $m_{hj}$ in product model mix of line $L_h$, where a normalised demand for a product model is defined as the demand in terms of greatest common divisor of the relevant line,

$MPS_h$ : Minimum part set or product model mix of line $L_h$ (which is equivalent to $d_h$ vector where $d_h = d_{h1}, \dots, d_{hM_h}$),

$MS_h$ : Product model sequence of line $L_h$,

$S_h$ : Total number of product models on line $L_h$ for one $MPS_h$ (the length of $MS_h$ for one $MPS_h$), $\left(S_h = \sum_{j=1}^{M_h} d_{hj}\right)$,

$op_{hj}$ : Overall proportion of assembled product model $m_{hj}$ on line $L_h$; $op_{hj} = \frac{D_{hj}}{\sum_{j=1}^{M_h} D_{hj}}, (h = 1, \dots, H)$,

$\gamma_1, \gamma_2, \gamma_3$: User defined weighting factors to determine the significance of performance measures, *i.e.* the weight associated with each objective function,

$PZ_{hj}$ : Set of pairs of tasks that must be assigned to the same workstation for product model $m_{hj}$ on line $L_h$ (positive zoning),

$NZ_{hj}$ : Set of pairs of tasks that must be assigned to different workstations for product model $m_{hj}$ on line $L_h$ (negative zoning),

$DL_{hj}$ : Set of left direction tasks for product model $m_{hj}$ on line $L_h$,

$DR_{hj}$ : Set of right direction tasks for product model $m_{hj}$ on line $L_h$.

#### 4.6.1.3. Decision variables

$$Y_{hjikx}^{\varphi} = \begin{cases} 1 & \text{if task } t_{hji} \text{ of } m_{hj} \text{ is assigned to station } W_{hkx} \text{ on side } x \text{ of line } L_h \text{ in cycle } \varphi \\ 0 & \text{otherwise} \end{cases},$$

$$\tau_{hjq}^{\varphi} = \begin{cases} 1 & \text{if } m_{hj} \text{ is produced in queue } q \text{ on line } L_h \text{ in production cycle } \varphi \\ 0 & \text{otherwise} \end{cases},$$

$st_{hji}^{\varphi}$ : Starting time of task $t_{hji}$ of product model $m_{hj}$ on line $L_h$ in production cycle $\varphi$,

$$U_{hkx}^{\varphi} = \begin{cases} 1 & \text{if station } W_{hkx} \text{ is utilised on side } x \text{ of line } L_h \text{ in production cycle } \varphi \\ 0 & \text{otherwise} \end{cases}.$$

### 4.6.1.4. Intermediate variables

$q_{hkx}$ : Queue number in which station $W_{hkx}$ is utilised,

$LL$ : Line length, which is equivalent to maximum number of queues ($LL = \frac{\sum_{h=1}^{H} K_h}{2H}$),

$$LS_{hkx} = \begin{cases} 0 & \text{if station } W_{hkx} \text{ is utilised on left side of the first line in production cycle } \varphi \\ 1 & \text{otherwise} \end{cases},$$

$K_{hq}^{\varphi}$ : Set of workstations located in queue $q$ on line $L_h$ in production cycle $\varphi$,

$\sigma$ : Integer variable ($\sigma = h+1, \dots, H$),

$\beta$ : Integer variable ($\beta = 0, 1$),

$$c = \begin{cases} 1 & \text{if } x = 1 \text{ and } \beta = 1 \\ 0 & otherwise \end{cases},$$

$$\mu = \begin{cases} 1 & \text{if } (\sigma - h) = 1 \; and \; \beta = 1 \\ 0 & otherwise \end{cases},$$

$$R_{hjrv}^{\varphi} = \begin{cases} 1 & \text{if tasks } r \text{ and } v \text{ for } m_{hj} \text{ on line } L_h \text{ are assigned in the same queue in cycle } \varphi \\ 0 & \text{otherwise} \end{cases}.$$

### 4.6.2. Objective function

In the literature, a large number of studies on parallel assembly line balancing problems and two-sided assembly line balancing problems consider only minimisation of total number of required workstations as the main objective while just a limited number of studies consider the minimisation of the line length (or number of mated workstations) as an additional objective. However, the line length should also be considered in MPTALB/S problem since different configurations of the lines are possible with the same number of workstations due to the nature of the parallel two-sided assembly lines. Utilisation of a multi-line station will influence the objective function as equally as a regular station as only one operator is allocated to each of those workstations.

Workload smoothness is another criterion that shows whether the lines are well balanced, especially to make a distinction between two different solutions that needs the same number of workstations. Based on this idea, Ozcan *et al.* (2010b) constructed their objective function for parallel two-sided assembly line balancing problem as in Equation (4.6) (please note that mixed-models have not been considered in their research).

$$max \sum_{k=1}^{K} \left( \sum_{i \in A_k \wedge i \in h} pt_{hi} \right)^2 , \tag{4.6}$$

where, $i$ is the task index ($i = 1, \dots, T_h$), $pt_{hi}$ is the processing time of task $i$ in line $h$, $k$ is the station number ($k = 1, \dots, K$), and $A_k$ is the set of tasks which are assigned to station $k$. Please note that the notation is changed to provide conformity with the notation used in this thesis.

Equation (4.6) represents the sum of squares of each workstation's workload. Therefore, maximising this objective function helps to reduce the number of stations. However, the mixed-model situation is not considered in that function.

Another objective function (see Equation (4.7)) developed by Ozcan *et al.* (2010a) for mixed-model parallel assembly line balancing and sequencing problem aims to maximise weighted line efficiency (WLE) and minimise weighted workload difference (EW) between stations.

$$Minimise\ E = \frac{f_1^{max}(WLE) - f_1(WLE)}{f_1^{max}(WLE) - f_1^0(WLE)} + \frac{f_2^{min}(EW) - f_2(EW)}{f_2^{min}(EW) - f_2^0(EW)}, \tag{4.7}$$

where $f_1^0(WLE)$ and $f_2^0(EW)$ are obtained from the initial solution and are the least desirable objective values of $f_1(WLE)$ and $f_2(EW)$; and $f_1^{max}(WLE)$ and $f_2^{min}(EW)$ are the target values of $WLE$ and $EW$, respectively. The target values for $WLE$ and $EW$ are 1 and 0 respectively since the aim is the maximisation of line efficiency and minimisation of workload difference among workstations.

They calculated weighted line efficiency and weighted workload balance respectively as in Equation (4.8) and Equation (4.9) to obtain a single objective function using minimum deviation method, which is applicable when partial information of the objectives is owned. *"The aim was to find the best compromise solution which minimises the sum of individual objective's fractional deviations"* (Ozcan *et al.*, 2010a)

– please note that the notation used in these equations is also changed to provide conformity with the notation used in this thesis.

$$Maximise\ f_1(WLE) = \frac{\sum_{h=1}^{H} \sum_{j=1}^{M_h} \sum_{i=1}^{T_h} \frac{d_{hj}}{S_h} \times pt_{hji}^{norm}}{K \times C}\ , \tag{4.8}$$

where $h$ is the index of each straight line $(h = 1, \dots, H)$, $j$ is the product model for a specified line $(j = 1, \dots, M_h)$, $i$ is the task index for specified line $(i = 1, \dots, T_h)$, $pt_{hji}$ is the processing time of task $i$ for product model $j$ on line $h$, $k$ is the station index $(k = 1, \dots, K)$, and $pt_{hji}^{norm}$ is the normalised task time of task $i$ for product model $j$ on line $h$. Cycle times $(C_h)$ are calculated for each line and transformed to common cycle time $(C)$ based on the least common multiple approach proposed by Gökçen *et al.* (2006); (Ozcan *et al.*, 2010a).

$$Minimise\ f_2(EW) = \sqrt{\frac{\sum_{k=1}^{K} \sum_{\varphi=1}^{\phi} (ST_k^{\varphi} - ST_{max})^2}{K \times \phi}}\ , \tag{4.9}$$

where $\phi$ is the maximum number of product model mixes which may appear at a station on two neighbouring lines, $ST_k^{\varphi}$ is the workload of station $k$ at cycle $\varphi$, and $ST_{max}$ is the maximum station time. Please refer to (Ozcan *et al.*, 2010a) for more detailed explanations and calculations of some expressions.

A modified version of the objective function developed by Ozcan *et al.* (2010a) could be used to solve MPTALB/S problems as well. However, further problem specific objectives and constraints are necessary to obtain a feasible as well as efficient assignment, for example, product models are sequenced in production cycles appropriately, demand is satisfied for each product model on each line, multi-line stations are utilised with no violation of technological constraints, and tasks are assigned to preferred operation side of the lines. For this aim, a new mixed integer mathematical programming model has been proposed and presented below in this thesis. A new objective function is developed considering weighted idle times (*WIT)* of the stations (which is also equivalent to taking total number of utilised workstations into account), workload smoothness (*WLS*), and line length (*LL*). The objective function developed for modelling the MPTALB/S problem is given in Equations (4.10) – (4.13).

$$Min\ Z = \gamma_1 WIT + \gamma_2 WLS + \gamma_3 LL\ , \tag{4.10}$$

$$WIT = \sum_{\varphi=1}^{\phi} \sum_{h=1}^{H} \sum_{k=1}^{K_h} \sum_{x\in\{0,1\}} \left( C - \sum_{j=1}^{M_h} \sum_{i=1}^{T_{hj}} op_{hj} pt_{hji} \, Y_{hjikx}^{\varphi} \right), \tag{4.11}$$

$$WLS = \sum_{\varphi=1}^{\phi} \sum_{k=1}^{K_h} \sum_{x\in\{0,1\}} \frac{\sum_{h=1}^{H} \left( \sum_{j=1}^{M_h} \sum_{i=1}^{T_{hj}} op_{hj} pt_{hji} \, Y_{hjikx}^{\varphi} - C \right)^2}{\sum_{h=1}^{H} K_h}, \tag{4.12}$$

$$LL = \frac{\sum_{h=1}^{H} K_h}{2H}. \tag{4.13}$$

The main objective of the model is to minimise weighted idle times, which is also equivalent to minimise the total number of utilised workstations, as well as to ensure a smooth workload among stations from one production cycle to another. As known, the length of the assembly lines could also be of subject to consideration based on the limitations on available space at manufacturing plants. For that reason, line length is also considered as an additional objective in the proposed model. $\gamma_1$, $\gamma_2$, and $\gamma_3$ are user defined weighting factors which allow the decision maker to decide the significance levels of the objectives.

### 4.6.3. Constraints

#### 4.6.3.1. Product model occurrence constraint

Only up to one product model $m_{hj}$ can be produced in each queue $(q)$, on each line $(L_h)$, in each production cycle $(\varphi)$ at a time. In other words, the total number of product models produced in a queue $(q)$ on line $(L_h)$ in each production cycle $(\varphi)$ at a time is smaller than or equal to 1.

$$\sum_{j=1}^{M_h} \tau_{hjq}^{\varphi} \leq 1; \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ q = 1,..,LL. \tag{4.14}$$

#### 4.6.3.2. Task occurrence constraint

In a production cycle $(\varphi)$, each task $(t_{hji})$ belonging to each product model $(m_{hj})$ can be assigned at most once to all queues $(q)$, sides $(x)$, and stations $(W_{hkx})$.

$$\sum_{q=1}^{LL} \sum_{x\in\{0,1\}} \sum_{k=1}^{K_h} \tau_{hjq}^{\varphi} \, Y_{hjikx}^{\varphi} \leq 1; \quad i = 1,..,T_{hj};\ j = 1,..,M_h;\ h = 1,..,H;$$

$$\varphi = 1, .., \phi. \tag{4.15}$$

### 4.6.3.3. Task assignment constraint for demand satisfaction

Each task must be assigned exactly $d_{hj}$ times in all production cycles. In other words, each task $(t_{hji})$ for each product model $(m_{hj})$ must be assigned exactly $d_{hj}$ times; in all production cycles $(\varphi)$, queues $(q)$, sides $(x)$, and stations $(W_{hkx})$. It is ensured that all tasks are assigned to a station exactly once.

$$\sum_{\varphi=1}^{\phi}\sum_{q=1}^{LL}\sum_{x\in\{0,1\}}\sum_{k\in K_{hq}^{\varphi}} \tau_{hjq}^{\varphi}\, Y_{hjikx}^{\varphi} = d_{hj}; \quad i = 1, .., T_{hj};\ j = 1, .., M_h;$$

$$h = 1, .., H. \tag{4.16}$$

### 4.6.3.4. Operation direction constraints

A left side task $(t_{hji} \in DL_{hj})$ for each product model $(m_{hj})$ on line $(L_h)$ must be assigned to left side stations $(x = 0)$ exactly $d_{hj}$ times; in all production cycles $(\varphi)$, queues $(q)$, and stations $(W_{hkx})$.

$$\sum_{\varphi=1}^{\phi}\sum_{q=1}^{LL}\sum_{k=1}^{K_h} \tau_{hjq}^{\varphi}\, Y_{hjik0}^{\varphi} = d_{hj}; \quad \forall t_{hji} \in DL_{hj};\ j = 1, .., M_h;\ h = 1, .., H. \tag{4.17a}$$

A right side task $(t_{hji} \in DR_{hj})$ for each product model $(m_{hj})$ on line $(L_h)$ must be assigned to right side stations $(x = 1)$ exactly $d_{hj}$ times; in all production cycles $(\varphi)$, queues $(q)$, and stations $(W_{hkx})$.

$$\sum_{\varphi=1}^{\phi}\sum_{q=1}^{LL}\sum_{k=1}^{K_h} \tau_{hjq}^{\varphi}\, Y_{hjik1}^{\varphi} = d_{hj}; \quad \forall t_{hji} \in DR_{hj};\ j = 1, .., M_h;\ h = 1, .., H. \tag{4.17b}$$

### 4.6.3.5. Precedence relationship constraints

Precedence relationship constraints ensure that the precedence relationships are not violated on the line $L_h$ precedence diagram and completion time of tasks are considered to avoid interference. These constraints must be considered for each predecessor of a task $(r \in P_{hjv})$, where $v \in T_{hj}$, in each production cycle $(\varphi)$ on each line $(L_h)$, for each product model $(m_{hj})$.

Two different situations may occur during the balancing procedure: *(i)* tasks $r$ and $v$ are assigned to different queues and *(ii)* tasks $r$ and $v$ are assigned to the same queue. The following equation is active if tasks $r$ and $v$ are assigned to different queues in a production cycle ($\varphi$) (on each line, $L_h$, for each product model, $m_{hj}$, for each predecessor of $v$):

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} q_{hkx}\left(Y_{hjrkx}^{\varphi} - Y_{hjvkx}^{\varphi}\right) \leq 0\,; \quad \forall r \in P_{hjv};\ \varphi = 1,..,\phi;\ h = 1,..,H;$$
$$j = 1,..,M_h. \tag{4.18a}$$

Following equation is active if tasks $r$ and $v$ are assigned to the same queue in the same production cycle ($\varphi$) (on each line, $L_h$, for each product model, $m_{hj}$, for each predecessor of $v$):

$$R_{hjrv}^{\varphi}\left(st_{hjr}^{\varphi} + pt_{hjr} - st_{hjv}^{\varphi}\right) \leq 0; \quad \forall r \in P_{hjv};\ \varphi = 1,..,\phi;\ h = 1,..,H;$$
$$j = 1,..,M_h. \tag{4.18b}$$

#### 4.6.3.6. Capacity constraint for regular stations

The capacity constraint ensures that the total workload of a workstation does not exceed the pre-determined cycle time. In other words, capacity constraint assures each task is executed within the cycle time.

$$Y_{hjikx}^{\varphi}\left(st_{hji}^{\varphi} + pt_{hji}\right) \leq C; \quad \varphi = 1,..,\phi;\ \ h = 1,..,H;\ \ \forall x \in \{0,1\};\ \ k = 1,..,K_h;$$
$$i = 1,..,T_{hj};\ \ j = 1,..,M_h. \tag{4.19}$$

#### 4.6.3.7. Capacity constraint for multi-line stations

If some tasks are assigned to a right side station of line $L_h$ from left side station of its adjacent line, the total workload of this multi-line station cannot exceed its capacity.

$$\sum_{j=1}^{M_h}\sum_{i=1}^{T_{hj}} pt_{hji}\, Y_{hjikx}^{\varphi} + LS_{hkx}\left(\sum_{j=1}^{M_h}\sum_{i=1}^{T_{hj}} pt_{(h+1)ji}\, Y_{(h+1)jik(x-1)}^{\varphi}\right) \leq C\, U_{hkx}^{\varphi};$$
$$\varphi = 1,..,\phi;\ \ h = 1,\dots,H-1;\ \ k = 1,..,K_h;\ \ \forall x \in \{0,1\}. \tag{4.20}$$

#### 4.6.3.8. Assigning to multi-line stations constraints

This constraint defines whether any task is assigned to workstation $W_{hkx}$ from its adjacent line.

$$\sum_{i=1}^{T_{hj}} Y_{hjikx}^{\varphi} - T_{hj} U_{hkx}^{\varphi} \leq 0\,; \quad \varphi = 1,..,\phi;\ \ h = 1,..,H;\ \ j = 1,..,M_h;$$

$$k = 1,..,K_h;\ \forall x \in \{0,1\}. \qquad (4.21)$$

#### 4.6.3.9. Valid zone constraints for multi-line stations

A multi-line station can only perform tasks from its adjacent line and side. Following constraints ensure that an operator working at station $W_{hkx}$ can only perform task(s) additionally from only one adjacent line and side; unless station $W_{hkx}$ is utilised on left side of the first line or on right side of the last line. For example, if an operator is located on right side of the first line $(L_h = 1, x = 1)$, that operator can perform additional tasks from only left side of the second line $(L_h = 2, x = 0)$ along with his/her main job. The operator cannot perform any job from left side of the first line $(L_h = 1, x = 0)$, or right side of the second line $(L_h = 2, x = 1)$, since it is not possible a direct communication with those tasks assigned to these stations.

Following constraint controls utilising multi-line station for the lines from 1 to $H - 1$ $(h = 1, \dots, H - 1)$.

$$(1 - x)\left(U_{hk\beta}^{\varphi} + U_{(\sigma-\mu)kx}^{\varphi}\right) + x\left(U_{hkc}^{\varphi} + U_{\sigma kx}^{\varphi}\right) = 1; \quad \varphi = 1,..,\phi;\ \ k = 1, \dots, K_h;$$

$$h = 1, \dots, H - 1;\ \ \forall x \in \{0,1\};\ \ \sigma = h + 1, \dots, H;\ \ \forall \beta \in \{0,1\}. \qquad (4.22a)$$

Following constraint restricts utilising multi-line station for the right side of the last line $(h = H)$.

$$U_{Hk0}^{\varphi} + U_{Hk1}^{\varphi} = 1; \quad \varphi = 1,..,\phi;\ \ k = 1,..,K_h. \qquad (4.22b)$$

#### 4.6.3.10. Positive and negative zoning constraints

Some tasks may need to be processed in the same workstation for some specific reasons that may originate from the work environment or tool requirements. In the literature, this condition is called positive zoning constraint and it is ensured that those tasks are assigned to the same workstation. $PZ_{hj}$ is the set of pairs of tasks that must be assigned to the same workstation for product model $m_{hj}$ on line $L_h$.

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} k\left(Y_{hjakx}^{\varphi}-Y_{hjbkx}^{\varphi}\right)=0;\quad \forall(a,b)\in PZ_{hj};\ \ \varphi=1,..,\phi;\ \ h=1,..,H;$$
$$j=1,..,M_h. \tag{4.23a}$$

On the other hand, some tasks must be performed in different workstations due to safety reasons or processing obligations. This is controlled by negative zoning constraint, which ensures that those tasks are assigned to different workstations. $NZ_{hj}$ is the set of pairs of tasks that must be assigned to different workstations for product model $m_{hj}$ on line $L_h$.

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} k\left(Y_{hjakx}^{\varphi}-Y_{hjbkx}^{\varphi}\right)\neq 0;\quad \forall(a,b)\in NZ_{hj};\ \ \varphi=1,..,\phi;\ \ h=1,..,H;$$
$$j=1,..,M_h. \tag{4.23b}$$

### 4.6.3.11. Variable constraints

Decision variable and indicator variable constraints are as follows:

$$Y_{hjikx}^{\varphi}\in\{0,1\},\quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ j=1,..,M_h;\ \ i=1,..,T_{hj};\ \ k=1,..,K_h;$$
$$\forall x\in\{0,1\}. \tag{4.24}$$

$$U_{hkx}^{\varphi}\in\{0,1\},\quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ k=1,..,K_h;\ \ \forall x\in\{0,1\}. \tag{4.25}$$

$$\tau_{hjq}^{\varphi}\in\{0,1\},\quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ j=1,..,M_h;\ \ q=1,..,LL. \tag{4.26}$$

$$st_{hji}^{\varphi}\geq 0,\quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ j=1,..,M_h;\ \ i=1,..,T_{hj}. \tag{4.27}$$

$$LS_{hkx}\in\{0,1\},\quad h=1,..,H;\ \ k=1,..,K_h;\ \ \forall x\in\{0,1\}. \tag{4.28}$$

$$q_{hkx}\geq 0,\quad h=1,..,H;\ \ k=1,..,K_h;\ \ \forall x\in\{0,1\}. \tag{4.29}$$

$$R_{hjrv}^{\varphi}\in\{0,1\},\quad \varphi=1,..,\phi;\ \ \forall r\in P_{hjv}. \tag{4.30}$$

$$\sigma=h+1,\dots,H. \tag{4.31}$$

$$\beta,c,\mu\in\{0,1\}. \tag{4.32}$$

$$\mathrm{LL}>0. \tag{4.33}$$

### 4.6.4. Complete model

The complete mathematical model of MPTALB/S problem is given as follows.

**Objective Function:**

$$Min\, Z = \gamma_1 WIT + \gamma_2 WLS + \gamma_3 LL;$$

$$WIT = \sum_{\varphi=1}^{\phi} \sum_{h=1}^{H} \sum_{k=1}^{K_h} \sum_{x \in \{0,1\}} \left( C - \sum_{j=1}^{M_h} \sum_{i=1}^{T_{hj}} op_{hj} pt_{hji}\, Y_{hjikx}^{\varphi} \right);$$

$$WLS = \sum_{\varphi=1}^{\phi} \sum_{k=1}^{K_h} \sum_{x \in \{0,1\}} \frac{\sum_{h=1}^{H} \left( \sum_{j=1}^{M_h} \sum_{i=1}^{T_{hj}} op_{hj} pt_{hji}\, Y_{hjikx}^{\varphi} - C \right)^2}{\sum_{h=1}^{H} K_h};$$

$$LL = \frac{\sum_{h=1}^{H} K_h}{2H}.$$

**Subject to:**

$$\sum_{j=1}^{M_h} \tau_{hjq}^{\varphi} \le 1; \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ q = 1,..,LL.$$

$$\sum_{q=1}^{LL} \sum_{x \in \{0,1\}} \sum_{k=1}^{K_h} \tau_{hjq}^{\varphi}\, Y_{hjikx}^{\varphi} \le 1; \quad i = 1,..,T_{hj};\ j = 1,..,M_h;\ h = 1,..,H;$$

$$\varphi = 1,..,\phi.$$

$$\sum_{\varphi=1}^{\phi} \sum_{q=1}^{LL} \sum_{x \in \{0,1\}} \sum_{k \in K_{hq}^{\varphi}} \tau_{hjq}^{\varphi}\, Y_{hjikx}^{\varphi} = d_{hj}; \quad i = 1,..,T_{hj};\ j = 1,..,M_h;$$

$$h = 1,..,H.$$

$$\sum_{\varphi=1}^{\phi} \sum_{q=1}^{LL} \sum_{k=1}^{K_h} \tau_{hjq}^{\varphi}\, Y_{hjik0}^{\varphi} = d_{hj}; \quad \forall t_{hji} \in DL_{hj};\ j = 1,..,M_h;\ h = 1,..,H.$$

$$\sum_{\varphi=1}^{\phi} \sum_{q=1}^{LL} \sum_{k=1}^{K_h} \tau_{hjq}^{\varphi}\, Y_{hjik1}^{\varphi} = d_{hj}; \quad \forall t_{hji} \in DR_{hj};\ j = 1,..,M_h;\ h = 1,..,H.$$

**Subject to (continued):**

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} q_{hkx}\left(Y_{hjrkx}^{\varphi}-Y_{hjvkx}^{\varphi}\right)\leq 0\,; \quad \forall r\in P_{hjv};\ \varphi=1,..,\phi;\ h=1,..,H;$$

$$j=1,..,M_h.$$

$$R_{hjrv}^{\varphi}\left(st_{hjr}^{\varphi}+pt_{hjr}-st_{hjv}^{\varphi}\right)\leq 0; \quad \forall r\in P_{hjv};\ \varphi=1,..,\phi;\ h=1,..,H;$$

$$j=1,..,M_h.$$

$$Y_{hjikx}^{\varphi}\left(st_{hji}^{\varphi}+pt_{hji}\right)\leq C; \quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ \forall x\in\{0,1\};\ \ k=1,..,K_h;$$

$$i=1,..,T_{hj};\ \ j=1,..,M_h.$$

$$\sum_{j=1}^{M_h}\sum_{i=1}^{T_{hj}} pt_{hji}\,Y_{hjikx}^{\varphi}+LS_{hkx}\left(\sum_{j=1}^{M_h}\sum_{i=1}^{T_{hj}} pt_{(h+1)ji}\,Y_{(h+1)jik(x-1)}^{\varphi}\right)\leq C\,U_{hkx}^{\varphi};$$

$$\varphi=1,..,\phi;\ \ h=1,\dots,H-1;\ \ k=1,..,K_h;\ \ \forall x\in\{0,1\}.$$

$$\sum_{i=1}^{T_{hj}} Y_{hjikx}^{\varphi}-T_{hj}U_{hkx}^{\varphi}\leq 0\,; \quad \varphi=1,..,\phi;\ \ h=1,..,H;\ \ j=1,..,M_h;$$

$$k=1,..,K_h;\ \forall x\in\{0,1\}.$$

$$(1-x)\left(U_{hk\beta}^{\varphi}+U_{(\sigma-\mu)kx}^{\varphi}\right)+x\left(U_{hkc}^{\varphi}+U_{\sigma kx}^{\varphi}\right)=1; \quad \varphi=1,..,\phi;\ \ k=1,\dots,K_h;$$

$$h=1,\dots,H-1;\ \ \forall x\in\{0,1\};\ \ \sigma=h+1,\dots,H;\ \ \forall\beta\in\{0,1\}.$$

$$U_{Hk0}^{\varphi}+U_{Hk1}^{\varphi}=1; \quad \varphi=1,..,\phi;\ \ k=1,..,K_h.$$

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} k\left(Y_{hjakx}^{\varphi}-Y_{hjbkx}^{\varphi}\right)=0; \quad \forall(a,b)\in PZ_{hj};\ \ \varphi=1,..,\phi;\ \ h=1,..,H;$$

$$j=1,..,M_h.$$

$$\sum_{x\in\{0,1\}}\sum_{k=1}^{K_h} k\left(Y_{hjakx}^{\varphi}-Y_{hjbkx}^{\varphi}\right)\neq 0; \quad \forall(a,b)\in NZ_{hj};\ \ \varphi=1,..,\phi;\ \ h=1,..,H;$$

$$j=1,..,M_h.$$

**Subject to (continued):**

$Y^{\varphi}_{hjikx} \in \{0,1\}, \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ j = 1,..,M_h;\ i = 1,..,T_{hj};$

$\quad k = 1,..,K_h;\ \forall x \in \{0,1\}.$

$U^{\varphi}_{hkx} \in \{0,1\}, \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ k = 1,..,K_h;\ \forall x \in \{0,1\}.$

$\tau^{\varphi}_{hjq} \in \{0,1\}, \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ j = 1,..,M_h;\ q = 1,..,LL.$

$st^{\varphi}_{hji} \geq 0, \quad \varphi = 1,..,\phi;\ h = 1,..,H;\ j = 1,..,M_h;\ i = 1,..,T_{hj}.$

$LS_{hkx} \in \{0,1\}, \quad h = 1,..,H;\ k = 1,..,K_h;\ \forall x \in \{0,1\}.$

$q_{hkx} \geq 0, \quad h = 1,..,H;\ k = 1,..,K_h;\ \forall x \in \{0,1\}.$

$R^{\varphi}_{hjrv} \in \{0,1\}, \quad \varphi = 1,..,\phi;\ \forall r \in P_{hjv}.$

$\sigma = h + 1, \dots, H - 1.$

$\beta, c, \mu \in \{0,1\}.$

$\mathrm{LL} > 0.$

## 4.7. Chapter Summary

Growing interests from customers in customised products and increasing competitions among peers necessitate companies to configure their manufacturing systems more effectively than ever before. In this concept, mixed-model lines are replacing single-model lines in an industrial inquiry.

In mixed-model production lines, line balancing and model sequencing problems are reciprocally connected with each other and cannot be handled independently. This chapter commences with providing some evidence from the literature to support this argument. However, the number of studies, which consider balancing and sequencing problems simultaneously, is limited. Furthermore, extracted studies do not cover a wide range of line types; instead, the majority of them focus on mixed-model U-shaped lines. Interestingly, no studies addressed balancing and sequencing issues simultaneously in mixed-model parallel two-sided assembly lines.

The core of this chapter was characterised by the introduction of the simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines problem studied in this thesis. Thus, the MPTALB/S problem, which has yet been studied academically so far, has been defined comprehensively. Assumptions of the studied problem have been explained first, followed by an explanatory example which shows changing product model combinations across the lines through different production cycles. A new mathematical model has been developed to simultaneously sequence product models and balance operations on the mixed-model parallel two-sided assembly lines considering all possible production cycles. The proposed mathematical model aimed at minimising key performance measures: *(i)* weighted idle times (and so the total number of workstations utilised), *(ii)* workload variations among workstations, and *(iii)* length of the lines.

It is obvious that considering model sequencing problem simultaneously with line balancing problem makes the entire problem more complex and requires substantially efficient solution techniques to produce optimal or approximate (near-optimal) solutions in a reasonable amount of time. For that purpose, novel solution techniques will be developed based on the characteristics of the MPTALB/S problem and described with illustrations in the following chapters. The running mechanisms of the three variants of the agent-based ACO techniques developed will be explained with flowcharts and illustrated through examples in Chapters 5, 6, and 7 respectively. Also, some test results will be reported to measure the performances of the algorithms proposed.

# 5

# GENERALISED SOLUTION OF THE MIXED-MODEL PARALLEL TWO-SIDED ASSEMBLY LINE BALANCING PROBLEMS

**Contents**

## 5.1. Chapter Introduction

This chapter proposes an agent-based ACO approach (referred to as ABACO hereafter), which produces general solutions convenient for any product model sequence being launched. To start with, Section 5.2 briefly addresses how natural ant systems work and why an agent-based ACO algorithm is developed. Section 5.3 describes the developed ABACO approach for the generalised solution of the MPTALB/S problem, which was defined in the previous chapter. To show the solution building mechanism of ABACO, a numerical example is given in Section 5.4. Computational tests are conducted in Section 5.5 to make a comparison between the proposed system and traditional one, and the chapter is concluded with a summary given in Section 5.6.

## 5.2. Brief Background

As shown by Wee and Magazine (1982), SALBP is an NP-hard class of combinatorial optimisation problem. Since many complex characteristics of line configurations are involved in MPTALB/S problem along with the model sequencing problem, it is clear that the MPTALB/S problem is a much more sophisticated version of the SALBP and is a very complex NP-hard problem. In the literature, researchers usually utilise heuristic, meta-heuristic, and other approaches to find approximate solutions for such complex problems.

The ACO technique is one of those meta-heuristics consulted to solve complex engineering problems. It is inspired from the collective behaviour of ants and is one of the most efficient meta-heuristics in solving combinatorial optimisation problems. The idea underlying behind the ACO algorithms have already been explained in details in Section 3.3. Ant algorithms, initially proposed by Dorigo *et al.* (1996), belong to the category of nature-inspired algorithms. The initial form of ACO techniques, called the ant system, was developed to solve small-sized travelling salesman problem with up to 75 cities. Since then, several researchers have carried out a substantial amount of research in ACO and have developed algorithms which demonstrate better performance than the original ant system. Different variants of ACO have been developed and applied in many fields, especially for the combinatorial optimisation problems (*i.e.* see Dorigo and Blum (2005), Yu *et al.* (2009), Deng and Lin (2011), Mavrovouniotis and Yang (2013), Venkata Narasimha *et al.* (2013), and Thepphakorn (2013)). Dorigo and

Stützle (2004) described all available ACO algorithms and the review of the literature (Ting and Chen, 2013).

Aside from meta-heuristics, agent-based solution techniques have become popular in this context recently and have been used in many fields of engineering optimisation problems; such as manufacturing operations, production planning and scheduling problems. For example, Anussornnitisarn *et al*. (2005), Mes *et al*. (2007), Anosike and Zhang (2009), Bearzotti *et al*. (2012), Amini *et al*. (2012), and He *et al*. (2014) developed agent-based techniques to solve problems in modelling, management, and optimisation of manufacturing and transportation processes. Nevertheless, the applications of agent-based systems on the manufacturing and industrial scheduling problems are still very scarce in the literature. In the agent-based systems, a network of problem solvers collaborate with each other to find solutions for problems that are beyond their individual capabilities (Goh and Zhang, 2003).

The procedures of the developed agent-based ACO technique, called ABACO, are explained with flowcharts and illustrated through examples in the following sub-sections.

## 5.3. Description of the Proposed ABACO Approach

In ABACO, the ant colony solution algorithm is built over an agent-based platform. The platform consists of different agents with different tasks and one of the agents uses ACO algorithm to build solutions benefiting from success principle of collaborative ants in nature. The algorithm developed here does not include a sequencing procedure since the model sequencing problem is not considered along with the line balancing problem in this method.

The three-level ABACO system constructed for MPTALB/S problem is outlined in Figure 5-1. As can be seen from the figure, it has three agents, which are Facilitator Agent (FA), Planning Agent (PA), and Balancing Agent (BA). These agents are programme scripts interacting with each other to obtain a complete balancing solution and constitute the three levels of the computational system. Facilitator agent is the first level and fulfils the initialisation steps of the ABACO. FA reads input data including task times, precedence relationships, and product model demands and normalises it to be accessed easily and be used by other agents.

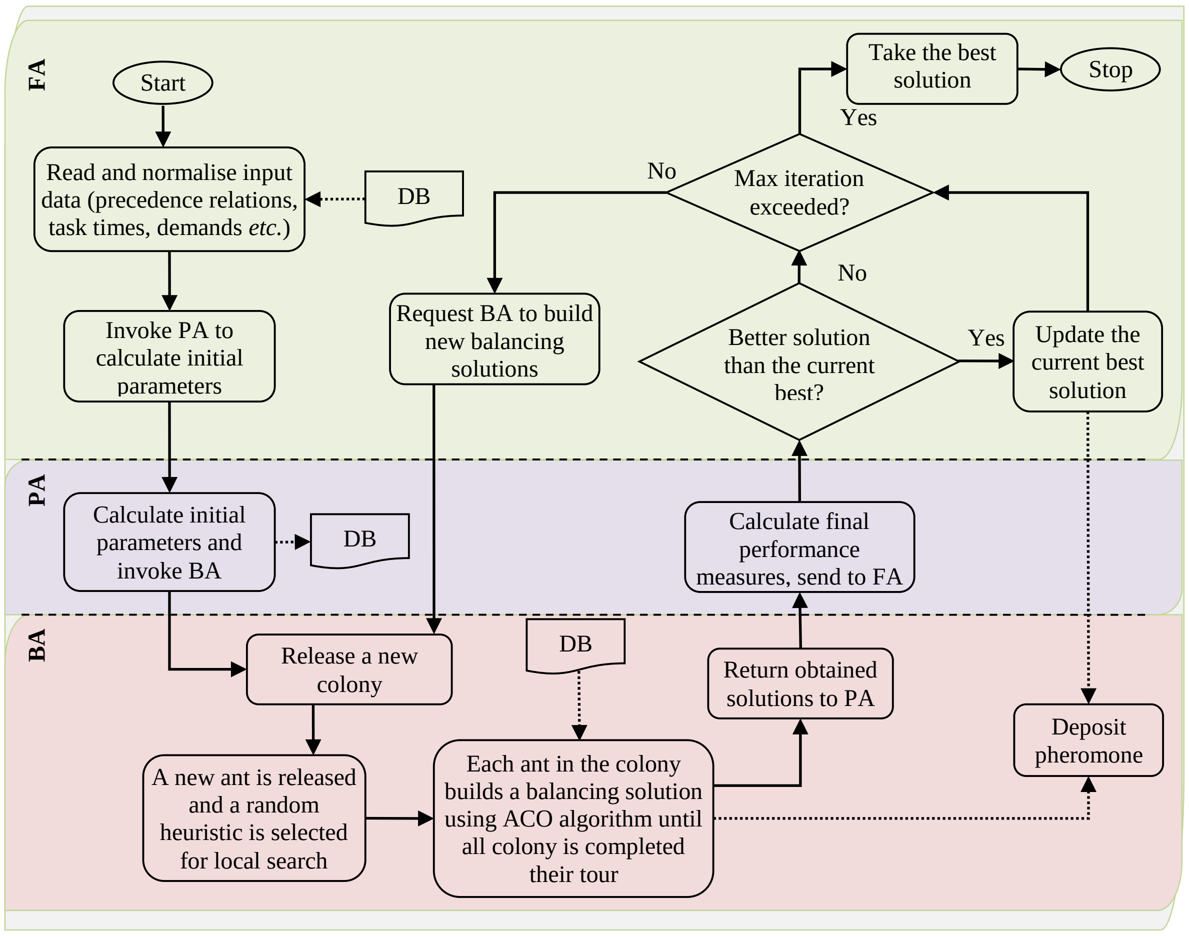

FA
Start
Read and normalise input data (precedence relations, task times, demands etc.)
DB
Invoke PA to calculate initial parameters
Request BA to build new balancing solutions
No
Max iteration exceeded?
Yes
Take the best solution
Stop
No
Better solution than the current best?
Yes
Update the current best solution
PA
Calculate initial parameters and invoke BA
DB
Calculate final performance measures, send to FA
BA
Release a new colony
DB
Return obtained solutions to PA
Deposit pheromone
A new ant is released and a random heuristic is selected for local search
Each ant in the colony builds a balancing solution using ACO algorithm until all colony is completed their tour

Figure 5-1. Outline of the proposed ABACO approach

PA is invoked by FA to perform planning and computing of parameter values. The cycle times of lines are determined according to the product model demands produced on each line and a common cycle time is designated for the lines. For this aim, the LCM based approach proposed by Gökçen *et al*. (2006) is used (as explained in the previous chapter) and task times are revised for each line according to the ratio of designated common cycle time to the original one.

Afterwards, BA is appointed to obtain a balancing solution using ACO algorithm built over the 2-ANTBAL algorithm proposed by Simaria and Vilarinho (2009). The ACO algorithm used in this thesis is enhanced by the following 10 heuristics, which are mostly popular in the line balancing literature, to search the solution space more effectively:

- Computer Method of Sequencing Operations for Assembly Lines - COMSOAL (Arcus, 1966): COMSOAL is a heuristic technique run by a computer to generate random solutions to the assembly line balancing problem. A list of available tasks which are feasible to be assigned is generated and the next task to be assigned is chosen, at random, from this list. Tasks are not prioritised and all tasks in the available tasks list have the same probability to be selected. To appear on the available tasks list, a task must have all predecessor activities completed and there must be enough capacity to perform the job (DePuy and Whitehouse, 2000). New stations are opened when required. A new balancing solution is obtained when all tasks are assigned to exactly one workstation. This procedure is repeated until the predefined number of iterations is exceeded and solutions which are worse than the current best solution are discarded.

- Ranked Positional Weight Method – RPWM (Helgeson and Birnie, 1961): The positional weights of each task is computed through taking the cumulative processing time of the associated task and its successors. In the mixed-model lines, the average processing times of tasks are used and average processing time of a task is considered as the sum of processing times of task for each product model weighted by the respective production share (Akpinar and Bayhan, 2011). Tasks with higher positional weights will have more chance to be assigned to the lowest numbered feasible workstation.
- Reverse Ranked Positional Weight Method – RRPWM (produced from RPWM): The positional weights of tasks are computed as in RPWM but different from RPWM, tasks with lower positional weights will be most likely assigned to the lowest numbered feasible workstation, if possible.
- Longest Processing Time – LPT (Talbot and Patterson, 1984): When a task will be assigned, available tasks are listed in a decreasing order in terms of their average processing times and tasks with the larger average processing time are most likely selected to be assigned first.
- Shortest Processing Time – SPT (Baykasoglu, 2006): As contrary to the LPT, tasks are listed in an increasing order based on their average processing times and tasks with lower average processing times have more chance to be selected and assigned to the lowest numbered stations.
- Smallest Task Number – STN (Arcus, 1963): Tasks are listed in an increasing order based on their task numbers, and tasks with the smallest number are made more favourable to be assigned to the lowest numbered stations when this technique is used.

- Maximum Number of Predecessors – MNP (produced from Maximum Number of Immediate Predecessors technique proposed by Baykasoglu (2006)): This technique applies a rule based on the total number of preceding tasks of a task. Tasks with the maximum number of predecessors have more chance to be assigned to the earliest stations.
- Least Number of Predecessors – LNP, (produced from Maximum Number of Immediate Predecessors technique proposed by Baykasoglu (2006)): As reverse to the MNP, tasks with the least number of predecessors have more chance to be assigned to the earliest stations.
- Maximum Number of Successors – MNS (produced from Maximum Number of Immediate Successors technique proposed by Tonge (1960)): Tasks are prioritised based on their total number of successors. Tasks which have more successors are more favourable to be assigned to the lowest numbered stations.
- Least Number of Successors – LNS (produced from Maximum Number of Immediate Successors technique proposed by Tonge (1960)): As contrary to the MNS, tasks which have less number of successors are more favourable to be assigned to the lowest numbered stations.

A new colony is released upon receiving a request from FA or PA and each ant in the colony picks up a random heuristic from the list of heuristics (which are given with detailed explanations above) to build a balancing solution. Ants start building individual balancing solution from any line or side randomly. Available tasks are determined very carefully through a set of procedures to make sure:

- The task has not already been assigned before.
- Precedence relationship constraints are not violated.
- Remaining capacity of the current workstation is enough to perform the largest processing time of associated task among all product models.
- Operation side constraints are satisfied.

Afterwards, an ant selects a task among the available ones in probability. The selection probability of a task is calculated using Equation (5.1).

$$p_{ik} = \frac{[\tau_{ik}]^{\alpha}[\eta_i]^{\beta}}{\sum_{y \epsilon Z_i}[\tau_{iy}]^{\alpha}[\eta_i]^{\beta}}, \tag{5.1}$$

where $i$, $k$, and $Z_i$ indicate task, current workstation, and the list of candidate tasks when task $i$ is selected, respectively. $\tau_{ik}$ is the amount of virtual pheromone between task – workstation, $\eta_i$ is the heuristic information of task $i$ that comes from the randomly selected heuristic by each ant and $\alpha$ and $\beta$ are user determined parameters. This probability is calculated by each ant and every time when a new task is being selected, and tasks with more probability will be favourable to be selected by ants.

Upon all ants in the colony completes their tour, in the pheromone matrix, an amount of pheromone is deposited between tasks and workstations that the tasks are assigned in. Then, results are sent to FA through PA and a proportional amount of pheromone is evaporated from all task-workstation pairs in the matrix using Equation (5.2).

$$\tau_{ik} \leftarrow (1-\rho)\tau_{ik} + \sum_{ant=1}^{CS} \Delta\tau_{ik}^{ant} \ , \tag{5.2}$$

where $CS$ is colony size, $\Delta\tau_{ik}^{ant} = \frac{Q}{Performance\ measure}$ and is calculated based on the quality of the solution found by each ant; $\rho$ is evaporation rate, and $Q$ is a user determined parameter.

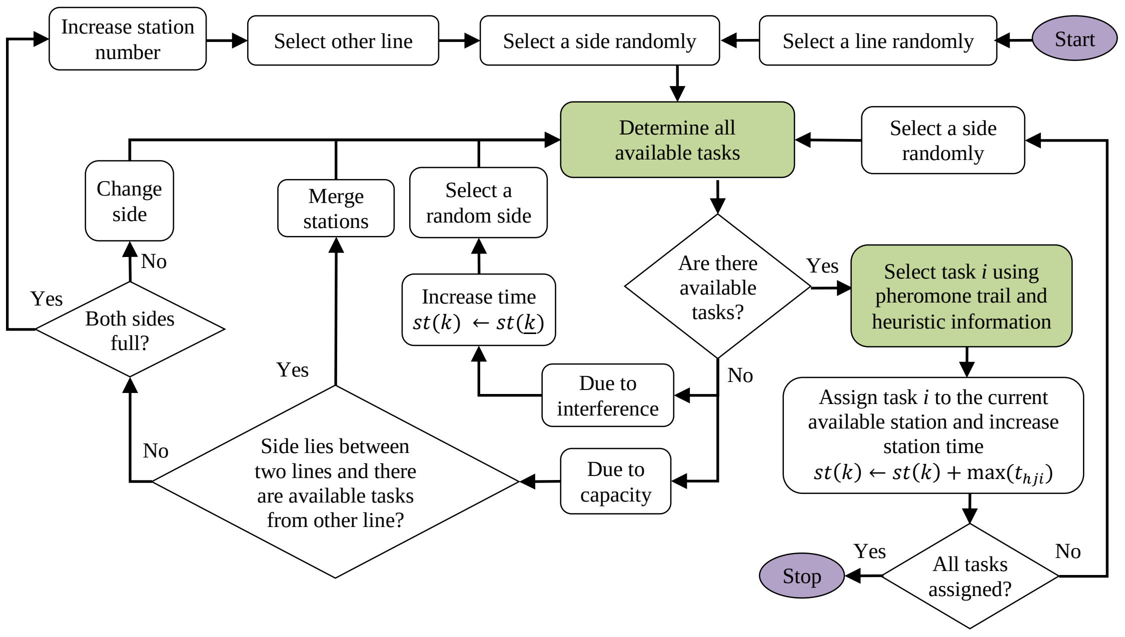

Start
Select a line randomly
Select a side randomly
Select other line
Increase station number
Determine all available tasks
Select a side randomly
Change side
Merge stations
Select a random side
Are there available tasks?
Yes
Select task $i$ using pheromone trail and heuristic information
No
Yes
Both sides full?
Increase time $st(k) \leftarrow st(k)$
Yes
Due to interference
No
Assign task $i$ to the current available station and increase station time $st(k) \leftarrow st(k) + \max(t_{hji})$
No
Side lies between two lines and there are available tasks from other line?
Due to capacity
Yes
No
Stop
All tasks assigned?

Figure 5-2. Building a balancing solution procedure of ABACO algorithm

If the solution obtained from the colony is better than the current best, then the current best solution is updated and double amount of pheromone is laid on the path of the new best solution referred to as global best. A new solution request is sent to BA and this cycle continues until a predetermined maximum iteration number is exceeded. The procedure of building a balancing solution is outlined in Figure 5-2.

The total number of utilised workstations $(NS)$ is usually considered as the unique performance measure for most of the line balancing problems. However, the length of the lines $(LL)$ is another key factor as well as the number of workstations while balancing two-sided lines. This is especially important if the manufacturing space is limited in the manufacturing plant. So, line length is also considered in the objective function given in Equation (5.3) – Equation (5.5) and importance of these two factors could be controlled by the user via assigning values to weighting parameters $\gamma_1$ and $\gamma_2$.

$$Min\ Z = \gamma_1 LL + \gamma_2 NS; \tag{5.3}$$

$$NS = \sum_{h=1}^{H} \sum_{k=1}^{K_h} \sum_{x \in \{0,1\}} U_{hkx}; \tag{5.4}$$

$$LL = max\{q_{hkx}\}; \qquad k = 1, \dots, K_h; \quad h = 1, \dots, H; \quad x \in \{0,1\}; \tag{5.5}$$

where $U_{hkx}$ equals 1 if workstation $W_{hkx}$ is utilised on side $x$ of line $L_h$, 0 otherwise; and $q_{hkk}$ indicates the queue number in which workstation $W_{hkx}$ is utilised in.

## 5.4. Illustrative Example

A numerical example is given here to show the solution building procedure used in ABACO algorithm. For this aim, two precedence relationship diagrams are taken from the two well-known test problems P16 (Lee *et al.*, 2001) and P24 (Kim *et al.*, 2000c) for Line I and Line II, respectively. Then, task times are generated between zero and the maximum task time of the original problem, as given in Table 5-1. In the table, *Side* column presents the preferred operation direction of each task; where L, R and E denote left, right and either side, respectively.

As ABACO generates more flexible solutions regardless from the sequences of product models being assembled on the lines, individual product model demands are not important. However, cumulative demands of individual product models on each line determine the cycle time of each line, which are assumed 16 and 18 time units for Line I and Line II, respectively. As the cycle times of the lines are different, the LCM based approach (Gökçen *et al.*, 2006, Ozcan *et al.*, 2010b) explained in Chapter 2 is used to maintain a common cycle time.

Table 5-1. Data for the numerical example

| **Line I** | | | | | | **Line II** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Task no | Side | A | B | C | Immediate predecessors | Task no | Side | D | E | F | Immediate predecessors |
| 1 | E | 6 | 7 | 6 | - | 1 | L | 3 | 3 | 0 | - |
| 2 | E | 5 | 2 | 0 | - | 2 | L | 7 | 0 | 2 | - |
| 3 | L | 2 | 5 | 9 | 1 | 3 | R | 7 | 1 | 1 | - |
| 4 | E | 9 | 2 | 8 | 1 | 4 | R | 5 | 0 | 0 | - |
| 5 | R | 8 | 9 | 5 | 2 | 5 | L | 4 | 6 | 1 | 2 |
| 6 | L | 4 | 8 | 0 | 3 | 6 | E | 3 | 5 | 1 | 2, 3 |
| 7 | E | 7 | 8 | 9 | 4, 5 | 7 | R | 4 | 8 | 5 | 3 |
| 8 | E | 4 | 6 | 3 | 6, 7 | 8 | E | 3 | 0 | 7 | 5 |
| 9 | R | 5 | 0 | 8 | 7 | 9 | E | 6 | 4 | 4 | 6 |
| 10 | R | 4 | 4 | 7 | 7 | 10 | E | 4 | 2 | 9 | 7 |
| 11 | E | 6 | 5 | 7 | 8 | 11 | L | 4 | 8 | 3 | 1 |
| 12 | L | 5 | 6 | 6 | 9 | 12 | L | 3 | 1 | 1 | 8, 9 |
| 13 | E | 6 | 4 | 9 | 9, 10 | 13 | E | 3 | 5 | 3 | 9 |
| 14 | E | 4 | 2 | 7 | 11 | 14 | R | 9 | 4 | 3 | 9, 10 |
| 15 | E | 3 | 6 | 9 | 11, 12 | 15 | R | 5 | 1 | 4 | 4 |
| 16 | E | 4 | 8 | 8 | 13 | 16 | L | 9 | 1 | 2 | 11 |
| - | - | - | - | - | - | 17 | E | 2 | 7 | 3 | 12 |
| - | - | - | - | - | - | 18 | E | 7 | 4 | 4 | 13 |
| - | - | - | - | - | - | 19 | E | 9 | 2 | 1 | 13, 14 |
| - | - | - | - | - | - | 20 | R | 9 | 1 | 1 | 15 |
| - | - | - | - | - | - | 21 | L | 8 | 9 | 7 | 16, 17 |
| - | - | - | - | - | - | 22 | E | 8 | 7 | 9 | 18 |
| - | - | - | - | - | - | 23 | R | 9 | 9 | 5 | 19, 20 |
| - | - | - | - | - | - | 24 | E | 9 | 3 | 5 | 20 |

For this aim, line divisors ($ld_h$) are calculated as $ld_1 = LCM(C_1, C_2)/C_1 = 144/16 = 9$ and $ld_2 = LCM(C_1, C_2)/C_2 = 144/18 = 8$. Then, processing time of each task for product models on Line I and Line II are multiplied by $ld_1$ and $ld_2$, respectively. $LCM(C_1, C_2) = 144$ is accepted as the common cycle time ($C$). These normalised task times (presented in Table 5-2) and common cycle time are used while balancing the lines.

A simple example solution building procedure of ABACO is shown in Figure 5-3. The balancing procedure starts from a randomly selected line and a randomly selected side (right side of Line I in this example). Then, available tasks are determined and assigned to the workstations one-by-one using the procedure explained in the previous section. Arrows show the task assignment order for this example using ABACO.

Table 5-2. Data with normalised task processing times

| **Line I** | | | | | | **Line II** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Task no | Side | A | B | C | Immediate predecessors | Task no | Side | A | B | C | Immediate predecessors |
| 1 | E | 54 | **63** | 54 | - | 1 | L | **24** | **24** | 0 | - |
| 2 | E | **45** | 18 | 0 | - | 2 | L | **56** | 0 | 16 | - |
| 3 | L | 18 | 45 | **81** | 1 | 3 | R | **56** | 8 | 8 | - |
| 4 | E | **81** | 18 | 72 | 1 | 4 | R | **40** | 0 | 0 | - |
| 5 | R | 72 | **81** | 45 | 2 | 5 | L | 32 | **48** | 8 | 2 |
| 6 | L | 36 | **72** | 0 | 3 | 6 | E | 24 | **40** | 8 | 2, 3 |
| 7 | E | 63 | 72 | **81** | 4, 5 | 7 | R | 32 | **64** | 40 | 3 |
| 8 | E | 36 | **54** | 27 | 6, 7 | 8 | E | 24 | 0 | **56** | 5 |
| 9 | R | 45 | 0 | **72** | 7 | 9 | E | **48** | 32 | 32 | 6 |
| 10 | R | 36 | 36 | **63** | 7 | 10 | E | 32 | 16 | **72** | 7 |
| 11 | E | 54 | 45 | **63** | 8 | 11 | L | 32 | **64** | 24 | 1 |
| 12 | L | 45 | **54** | **54** | 9 | 12 | L | **24** | 8 | 8 | 8, 9 |
| 13 | E | 54 | 36 | **81** | 9, 10 | 13 | E | 24 | **40** | 24 | 9 |
| 14 | E | 36 | 18 | **63** | 11 | 14 | R | **72** | 32 | 24 | 9, 10 |
| 15 | E | 27 | 54 | **81** | 11, 12 | 15 | R | **40** | 8 | 32 | 4 |
| 16 | E | 36 | **72** | **72** | 13 | 16 | L | **72** | 8 | 16 | 11 |
| - | - | - | - | - | - | 17 | E | 16 | **56** | 24 | 12 |
| - | - | - | - | - | - | 18 | E | **56** | 32 | 32 | 13 |
| - | - | - | - | - | - | 19 | E | **72** | 16 | 8 | 13, 14 |
| - | - | - | - | - | - | 20 | R | **72** | 8 | 8 | 15 |
| - | - | - | - | - | - | 21 | L | 64 | **72** | 56 | 16, 17 |
| - | - | - | - | - | - | 22 | E | 64 | 56 | **72** | 18 |
| - | - | - | - | - | - | 23 | R | **72** | **72** | 40 | 19, 20 |
| - | - | - | - | - | - | 24 | E | **72** | 24 | 40 | 20 |

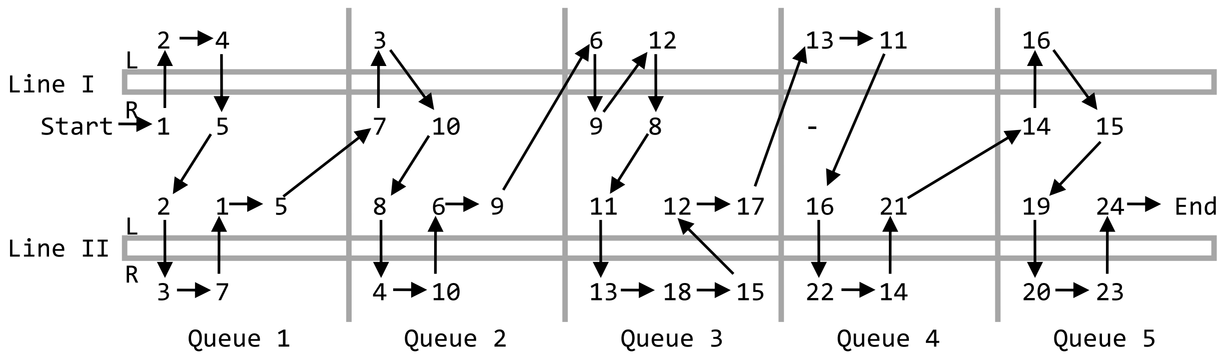


Figure 5-3. An example for the solution building procedure

Figure 5-4 presents the detailed balancing solution obtained for the numerical example problem. Assignment configuration of tasks to the workstations can be clearly seen from this figure where lengths of the bars correspond to the processing times of tasks given in those bars. By this way, the workload of each workstation can be easily seen

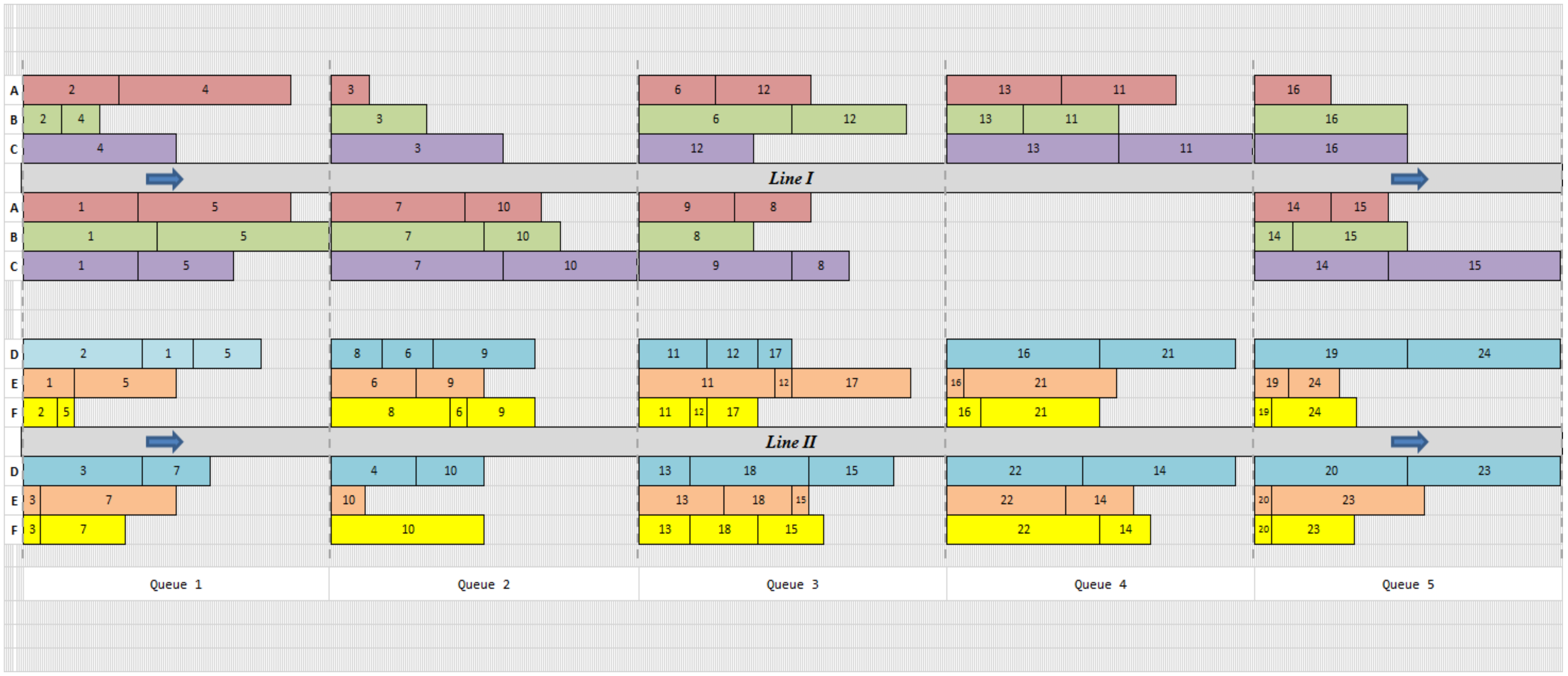


Figure 5-4. Detailed balancing solution and assignment of tasks for the given example

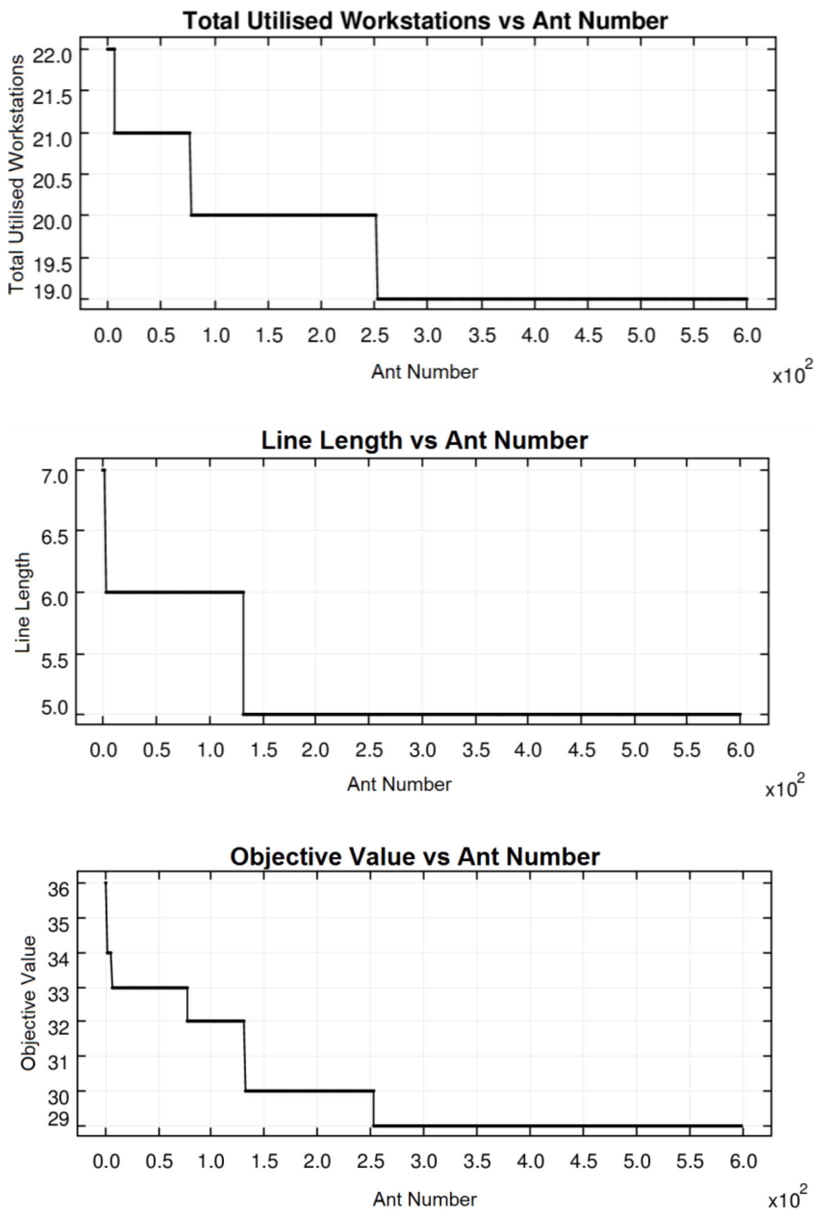


Figure 5-5. Convergence of the performance measures for the example problem

for each particular product model and line. As mentioned earlier, the maximum processing times of tasks among three product models (given in bold in Table 5-2) are considered to ensure the capacity constraint is satisfied for any product model being assembled in the system. To give an example about this, let us assume that we are assigning tasks to the right side workstation with a remaining capacity of 63 time units on Line I in queue 2 and all predecessors of tasks 9 and 10 have already been assigned. Only task 10 will be available to be assigned to the current position since processing time of task 9 for product model C (72 time units) exceeds the remaining capacity of the workstation, which is 63 time units.

The convergence of the objective value, which is calculated using the total number of workstations utilised and the line length, is depicted in Figure 5-5. In accordance with the weighting parameters used, the change in the length of the line affects the objective function value as double of that of the total number of utilised workstations ($\gamma_1 = 2$, $\gamma_2 = 1$).

## 5.5. Computational Tests

In order to analyse the efficiency of the proposed approach, 24 test cases are solved using ABACO and the obtained results are compared. Since there is no related study and so published results in the literature to make comparison, the same test cases are solved using six common heuristic approaches (COMSOAL, RPWM, RRPWM, LPT, LNP, and MNS) given in Section 5.3 and the results of the developed ABACO are compared with those obtained from six heuristics. The original versions of these conventional heuristics address only the SALBP, where only one product model of a product is assembled on a one-sided line and no parallel lines are considered. Whereas, the addressed problem here (MPTALB/S problem) is a much more complex version of the SALBP. Therefore, these techniques are adapted to solve the MPTALB/S problem and the same procedure is used with the ABACO when determining available tasks.

The heuristics and the proposed ABACO algorithm are coded in Java$^{TM}$ Standard Edition 7u4 and run on a 3.1 GHz Intel Core i5-2400 CPU computer. For the heuristics, the algorithm is terminated after 100 iterations and the best solution is taken after one run for each test case. That means, each test case is solved by each heuristic 100 times and the final heuristic solution for each test case represents the best among those results obtained. For ABACO, parameters are chosen experimentally as shown in Table 5-3 to acquire a high-quality solution in an acceptable period of time. As could be seen from the table, parameters may differ from one test case to another. That is to increase the capacity of the algorithm as search space enlarges with the increasing number of tasks.

Two parallel two-sided lines are assumed to be balanced with three product models on each of the lines. Required data was generated using the seven original test problems from the literature; namely, P9, P12 and P24 from Kim *et al.* (2000c); P16, A65, and A205 from Lee *et al.* (2001); and B148 from Bartholdi (1993). B148 was then modified by Lee *et al.* (2001). Also, processing times of tasks 1, 131, 132, and 133 for the problem A205 were modified to fit the rest of the data. Precedence relationship diagrams taken from these studies are considered as the common diagram for all three product models on the same line, but new task times are generated randomly (between zero and maximum task time of the original problem) for the missing product models. Table 5-4 provides the data used for test cases in this research. Detailed data on task times and precedence relationship diagrams of test problems used for experimental purposes are provided in Appendices.

Table 5-3. Parameters of the ABACO

| Problem no | Alpha | Beta | Evaporation rate | Initial pheromone | Colony size | Number of iterations |
|---|---|---|---|---|---|---|
| 1-6 | 0.1 | 0.2 | 0.1 | 10 | 10 | 10 |
| 7-14 | 0.1 | 0.2 | 0.1 | 15 | 20 | 30 |
| 15-24 | 0.1 | 0.2 | 0.1 | 20 | 30 | 60 |

Table 5-4. Data for test cases

| Problem scale | Test case | Problem | | Cycle time | | Minimum part set (MPS) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | *Line I* | | | *Line II* | | |
| | | *Line I* | *Line II* | *Line I* | *Line II* | *A* | *B* | *C* | *D* | *E* | *F* |
| Small | 1 | P9 | P9 | 4 | 7 | 4 | 2 | 1 | 2 | 1 | 1 |
| | 2 | P9 | P9 | 6 | 5 | 2 | 2 | 1 | 1 | 2 | 1 |
| | 3 | P9 | P12 | 5 | 8 | 2 | 1 | 1 | 2 | 2 | 1 |
| | 4 | P9 | P12 | 7 | 6 | 1 | 1 | 2 | 2 | 1 | 4 |
| | 5 | P12 | P12 | 4 | 5 | 2 | 1 | 2 | 1 | 2 | 1 |
| | 6 | P12 | P12 | 6 | 5 | 2 | 1 | 2 | 2 | 1 | 1 |
| Medium | 7 | P12 | P16 | 9 | 12 | 1 | 2 | 1 | 1 | 1 | 1 |
| | 8 | P12 | P16 | 10 | 12 | 1 | 1 | 1 | 1 | 2 | 2 |
| | 9 | P16 | P16 | 12 | 15 | 1 | 2 | 2 | 2 | 1 | 1 |
| | 10 | P16 | P16 | 16 | 14 | 1 | 4 | 2 | 2 | 1 | 1 |
| | 11 | P16 | P24 | 14 | 16 | 2 | 1 | 1 | 4 | 2 | 1 |
| | 12 | P16 | P24 | 16 | 18 | 1 | 3 | 2 | 1 | 2 | 1 |
| | 13 | P24 | P24 | 15 | 20 | 2 | 1 | 1 | 1 | 1 | 1 |
| | 14 | P24 | P24 | 25 | 20 | 1 | 2 | 1 | 1 | 2 | 2 |
| Large | 15 | A65 | A65 | 300 | 480 | 2 | 1 | 1 | 2 | 1 | 2 |
| | 16 | A65 | A65 | 420 | 360 | 1 | 1 | 2 | 2 | 4 | 1 |
| | 17 | A65 | B148 | 405 | 810 | 2 | 1 | 1 | 2 | 2 | 1 |
| | 18 | A65 | B148 | 675 | 540 | 2 | 1 | 1 | 1 | 2 | 2 |
| | 19 | B148 | B148 | 255 | 510 | 1 | 2 | 1 | 1 | 2 | 2 |
| | 20 | B148 | B148 | 425 | 340 | 2 | 1 | 1 | 2 | 2 | 1 |
| | 21 | B148 | A205 | 510 | 1020 | 2 | 1 | 3 | 1 | 2 | 2 |
| | 22 | B148 | A205 | 600 | 1200 | 2 | 1 | 1 | 1 | 1 | 1 |
| | 23 | A205 | A205 | 1200 | 1200 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 24 | A205 | A205 | 1000 | 2000 | 3 | 1 | 2 | 1 | 1 | 1 |

Table 5-5. Computational results when lines are balanced separately ($\gamma_1 = 2, \gamma_2 = 1$)

| **Test case** | **COMSOAL** | | | **RPWM** | | | **RRPWM** | | | **LPT** | | | **LNP** | | | **MNS** | | | **ABACO** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* |
| 1 | **4** | **12** | **20** | 5 | 14 | 24 | 5 | 13 | 23 | 6 | 12 | 24 | **4** | **12** | **20** | 5 | 13 | 23 | **4** | **12** | **20** |
| 2 | **3** | **11** | **17** | 4 | 11 | 19 | 4 | 11 | 19 | 4 | 11 | 19 | 4 | 12 | 20 | **3** | **11** | **17** | **3** | **11** | **17** |
| 3 | **3** | **10** | **16** | 4 | 10 | 18 | 4 | 10 | 18 | 4 | 11 | 19 | 3 | 11 | 17 | **3** | **10** | **16** | **3** | **10** | **16** |
| 4 | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** |
| 5 | 5 | 16 | 26 | 5 | 16 | 26 | 5 | 15 | 25 | 5 | 15 | 25 | **4** | **15** | **23** | 5 | 15 | 25 | **4** | **15** | **23** |
| 6 | 4 | 12 | 20 | 4 | 13 | 21 | 4 | 12 | 20 | 4 | 12 | 20 | 4 | 13 | 21 | **3** | **12** | **18** | **3** | **12** | **18** |
| 7 | **8** | **18** | **34** | **8** | **18** | **34** | 10 | 19 | 39 | 9 | 19 | 37 | 9 | 19 | 37 | **8** | **18** | **34** | **8** | **18** | **34** |
| 8 | 8 | 18 | 34 | 8 | 18 | 34 | 10 | 19 | 39 | 9 | 18 | 36 | 9 | 19 | 37 | 8 | 18 | 34 | ***8*** | ***17*** | ***33*** |
| 9 | **8** | **25** | **41** | **8** | **25** | **41** | 8 | 26 | 42 | **8** | **25** | **41** | 9 | 26 | 44 | **8** | **25** | **41** | **8** | **25** | **41** |
| 10 | 7 | 22 | 36 | **7** | **21** | **35** | 8 | 23 | 39 | 7 | 23 | 37 | 7 | 24 | 38 | 7 | 22 | 36 | **7** | **21** | **35** |
| 11 | 7 | 26 | 40 | 7 | 26 | 40 | 7 | 26 | 40 | 7 | 26 | 40 | 7 | 26 | 40 | 7 | 26 | 40 | ***7*** | ***25*** | ***39*** |
| 12 | 6 | 21 | 33 | **5** | **20** | **30** | 6 | 21 | 33 | 6 | 21 | 33 | 6 | 22 | 34 | **5** | **20** | **30** | **5** | **20** | **30** |
| 13 | 8 | 25 | 41 | 8 | 25 | 41 | 8 | 25 | 41 | 8 | 25 | 41 | 8 | 25 | 41 | 8 | 25 | 41 | ***7*** | ***24*** | ***38*** |
| 14 | 5 | 18 | 28 | 5 | 18 | 28 | 5 | 18 | 28 | 5 | 18 | 28 | 5 | 18 | 28 | 5 | 18 | 28 | ***5*** | ***17*** | ***27*** |
| 15 | 16 | 54 | 86 | **16** | **51** | **83** | **16** | **51** | **83** | **16** | **51** | **83** | 17 | 53 | 87 | **16** | **51** | **83** | **16** | **51** | **83** |
| 16 | 14 | 52 | 80 | 13 | 48 | 74 | 14 | 51 | 79 | 13 | 49 | 75 | 14 | 50 | 78 | 13 | 49 | 75 | ***13*** | ***47*** | ***73*** |
| 17 | 12 | 44 | 68 | 12 | 42 | 66 | 12 | 46 | 70 | 12 | 44 | 68 | 12 | 43 | 67 | 12 | 42 | 66 | ***11*** | ***41*** | ***63*** |
| 18 | 15 | 44 | 74 | 14 | 42 | 70 | 15 | 46 | 76 | 15 | 44 | 74 | 14 | 42 | 70 | 14 | 43 | 71 | ***14*** | ***41*** | ***69*** |
| 19 | 31 | 92 | 154 | 27 | 89 | 143 | 32 | 92 | 156 | 31 | 93 | 155 | 30 | 90 | 150 | 29 | 87 | 145 | ***26*** | ***86*** | ***142*** |
| 20 | 23 | 82 | 128 | 21 | 77 | 121 | 23 | 85 | 131 | 23 | 84 | 130 | 23 | 81 | 127 | 22 | 79 | 123 | ***21*** | ***76*** | ***118*** |
| 21 | 25 | 81 | 131 | 25 | 78 | 128 | 27 | 85 | 139 | 27 | 85 | 139 | 26 | 81 | 133 | 25 | 80 | 130 | ***24*** | ***77*** | ***125*** |
| 22 | 23 | 71 | 117 | 21 | 67 | 109 | 23 | 74 | 120 | 22 | 71 | 115 | 22 | 69 | 113 | 22 | 69 | 113 | ***21*** | ***66*** | ***108*** |
| 23 | 23 | 89 | 135 | **21** | **83** | **125** | 24 | 92 | 140 | 23 | 90 | 136 | 23 | 87 | 133 | 22 | 87 | 131 | **21** | **83** | **125** |
| 24 | 27 | 82 | 136 | 26 | 79 | 131 | 27 | 87 | 141 | 27 | 84 | 138 | 26 | 81 | 133 | 26 | 79 | 131 | ***25*** | ***76*** | ***126*** |

Test cases with generated data are solved under various cycle time constraints and obtained results are compared with respect to line length (*LL*), number of workstations (*NS*), and objective function value (*OBJ*). Table 5-5 and Table 5-6 present the computational results when the lines are balanced separately and together, respectively. In together balancing, the utilisation of multi-line stations is allowed between two lines. As can be seen from Table 5-5, where the experimental results are reported for separate balancing condition, ABACO finds better solutions than all of the six test heuristics (namely COMSOAL, RPWM, RRPWM, LPT, LNP, and MNS) for 12 out of 24 test cases, *i.e.* test cases #8, #11, #13, #14, #16–#22 and #24 (please see italicised *OBJ* values given in ABACO column).

Table 5-6. Computational results when lines are balanced together ($\gamma_1 = 2, \gamma_2 = 1$)

| **Test case** | **COMSOAL** | | | **RPWM** | | | **RRPWM** | | | **LPT** | | | **LNP** | | | **MNS** | | | **ABACO** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* |
| 1 | **4** | **12** | **20** | 5 | 13 | 23 | **4** | **12** | **20** | 6 | 12 | 24 | **4** | **12** | **20** | 4 | 14 | 22 | **4** | **12** | **20** |
| 2 | **3** | **11** | **17** | 4 | 11 | 19 | 4 | 12 | 20 | 4 | 11 | 19 | 4 | 12 | 20 | 3 | 12 | 18 | **3** | **11** | **17** |
| 3 | **3** | **10** | **16** | 4 | 10 | 18 | 3 | 11 | 17 | 4 | 10 | 18 | 3 | 11 | 17 | **3** | **10** | **16** | **3** | **10** | **16** |
| 4 | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** | **3** | **10** | **16** |
| 5 | 4 | 16 | 24 | 5 | 16 | 26 | 4 | 16 | 24 | 5 | 15 | 25 | 4 | 15 | 23 | 5 | 15 | 25 | ***4*** | ***14*** | ***22*** |
| 6 | 4 | 12 | 20 | 4 | 13 | 21 | 4 | 12 | 20 | 4 | 12 | 20 | 4 | 12 | 20 | 3 | 12 | 18 | ***3*** | ***11*** | ***17*** |
| 7 | 8 | 18 | 34 | 8 | 18 | 34 | 7 | 19 | 33 | **7** | **18** | **32** | 8 | 18 | 34 | 8 | 18 | 34 | **7** | **18** | **32** |
| 8 | 7 | 19 | 33 | 8 | 18 | 34 | 8 | 19 | 35 | 8 | 19 | 35 | 8 | 19 | 35 | 8 | 18 | 34 | ***7*** | ***18*** | ***32*** |
| 9 | 8 | 25 | 41 | 8 | 25 | 41 | 8 | 26 | 42 | 8 | 25 | 41 | 8 | 27 | 43 | 8 | 25 | 41 | ***7*** | ***25*** | ***39*** |
| 10 | 7 | 22 | 36 | **7** | **21** | **35** | 7 | 23 | 37 | 7 | 22 | 36 | 7 | 23 | 37 | **7** | **21** | **35** | **7** | **21** | **35** |
| 11 | 7 | 26 | 40 | **7** | **25** | **39** | 7 | 27 | 41 | 7 | 26 | 40 | 7 | 26 | 40 | 7 | 26 | 40 | **7** | **25** | **39** |
| 12 | 6 | 20 | 32 | 5 | 20 | 30 | 6 | 20 | 32 | 6 | 21 | 33 | 6 | 22 | 34 | 5 | 20 | 30 | ***5*** | ***19*** | ***29*** |
| 13 | 7 | 25 | 39 | 7 | 25 | 39 | 8 | 25 | 41 | 7 | 25 | 39 | 7 | 25 | 39 | 7 | 25 | 39 | ***7*** | ***24*** | ***38*** |
| 14 | 5 | 18 | 28 | 5 | 18 | 28 | 5 | 18 | 28 | **5** | **17** | **27** | **5** | **17** | **27** | **5** | **17** | **27** | **5** | **17** | **27** |
| 15 | 15 | 54 | 84 | **14** | **49** | **77** | 14 | 52 | 80 | 15 | 51 | 81 | 15 | 52 | 82 | 14 | 50 | 78 | **14** | **49** | **77** |
| 16 | 14 | 50 | 78 | 13 | 48 | 74 | 13 | 51 | 77 | 13 | 49 | 75 | 13 | 49 | 75 | 13 | 49 | 75 | ***13*** | ***46*** | ***72*** |
| 17 | 12 | 43 | 67 | 11 | 41 | 63 | 12 | 44 | 68 | 12 | 44 | 68 | 12 | 43 | 67 | 11 | 41 | 63 | ***11*** | ***40*** | ***62*** |
| 18 | 14 | 45 | 73 | **12** | **41** | **65** | 13 | 46 | 72 | 14 | 45 | 73 | 13 | 43 | 69 | **12** | **41** | **65** | **12** | **41** | **65** |
| 19 | 28 | 93 | 149 | 26 | 89 | 141 | 28 | 93 | 149 | 29 | 90 | 148 | 29 | 91 | 149 | 25 | 86 | 136 | ***24*** | ***85*** | ***133*** |
| 20 | 22 | 81 | 125 | 20 | 78 | 118 | 23 | 83 | 129 | 22 | 83 | 127 | 21 | 79 | 121 | 21 | 77 | 119 | ***20*** | ***77*** | ***117*** |
| 21 | 24 | 83 | 131 | 23 | 81 | 127 | 25 | 87 | 137 | 24 | 85 | 133 | 24 | 81 | 129 | 23 | 80 | 126 | ***22*** | ***79*** | ***123*** |
| 22 | 21 | 71 | 113 | 20 | 68 | 108 | 21 | 75 | 117 | 21 | 73 | 115 | 20 | 70 | 110 | 20 | 68 | 108 | ***19*** | ***67*** | ***105*** |
| 23 | 23 | 87 | 133 | 21 | 82 | 124 | 23 | 89 | 135 | 22 | 87 | 131 | 22 | 87 | 131 | 21 | 82 | 124 | ***21*** | ***81*** | ***123*** |
| 24 | 25 | 83 | 133 | 22 | 77 | 121 | 25 | 83 | 133 | 26 | 82 | 134 | 25 | 79 | 129 | 22 | 77 | 121 | ***22*** | ***76*** | ***120*** |

For the remaining ones (*i.e.* #1–#7, #9–#10, #12, #15 and #23), the results obtained by ABACO are either the same with or better than some of the other approaches. The solution building capacities of the heuristics get worse when the problem size gets bigger with the increasing number of tasks. Although no test heuristic finds any better solutions than ABACO for any of the test cases, MNS and RPWM find the same solutions as ABACO for eight and seven out of 24 test cases, respectively; *i.e.* see the results of test cases #2–#4, #6, #7, #9, #12 and #15 in MNS column and see the results of test cases #4, #7, #9, #10, #12, #15 and #23 in RPWM column. For better understanding, best results for each test problem are given in bold font in the table. While the COMSOAL heuristic finds the same solutions as ABACO for six test cases (*i.e.* #1–#4, #7, #9), RRPWM performs the worst as it finds for only two test cases (*i.e.* #4 and #15). The LPT and LNP heuristics also find the same solutions as ABACO for only three test cases.

The solution building capability of ABACO over six test heuristics improves in together balancing condition, for which the experimental test results are presented in Table 5-6. ABACO performs better than those test heuristics for 14 test cases out of 24, *i.e.* see the test cases #5, #6, #8, #9, #12, #13, #16, #17 and 19–#24. To increase readability, ABACO results for these test cases are given in italic font in Table 5-6. Among six test heuristics, RPWM and MNS are the two heuristics, which find closer results to the same test cases solved by ABACO (*i.e.* see test cases #4, #10, #11, #15 and #18 for RPWM and #3, #4, #10, #14 and #18 for MNS. The COMSOAL, LPT, LNP and RRPWM find four, three, three and two same results as ABACO, respectively. Although RPWM and MNS show modest advantage over other four heuristics (namely COMSOAL, RRPWM, LPT and LNP), all of six test heuristics fail to investigate good-quality solutions when the problem size grows.

Overall, these results indicate that ABACO outperforms all other test heuristics in terms of the sought performance measures while balancing lines either separately or together. However, another significant outcome of the experimental tests performed in this section is that the objective function value is improved when the lines are balanced together. If the ABACO results presented in Table 5-5 are compared to ABACO results provided in Table 5-6, the advantage of balancing lines together can be observed clearly. The positive effect of allowance to establish multi-line stations between two adjacent lines on minimising objective function can be distinguished easily in majority of the test cases, *i.e.* see *OBJ* values of test cases #5–#9, 12 and #15–#24 in both tables. As the number of tasks increases, the difference between the obtained objective function values of separate balancing and together balancing conditions increases. A set of statistical tests will be conducted in Chapter 8 for a more comprehensive analysis of the results reported in this chapter.

## 5.6. Chapter Summary

The main benefit of the proposed assembly line system is its flexibility to produce more than one product model on the same line with less workforce. This is because constructing multi-line stations on more than one assembly line minimises operator requirements. However, the complexity of the problem increases dramatically with the consideration of various product models, which have different precedence relationships, task times, and sequences on the lines. Moreover, obtaining an optimal solution becomes more difficult when the problem size increases. The reason is that the solution space grows exponentially as the number of tasks increases due to the nature of the NP-hard problems. It is the major reason why; *(i)* a considerable amount of researches in the literature strives to develop heuristics and meta-heuristics instead of exact algorithms to solve assembly line balancing problems, and *(ii)* an agent-based ACO algorithm is developed in this study for the MPTALB/S problem.

Obviously, considering different cycle times for each of the parallel lines contributes to the complexity of the problem as the product models produced on the lines will change at different production cycles. This situation yields requirement of a comprehensive study to determine the availability of operators working in multi-line stations and to assign tasks by fulfilling all other associated constraints.

The developed agent-based technique in this chapter, called ABACO, dealt with such a complex problem and built acceptable solutions in a considerable amount of time. In the proposed approach, agents interact with each other to build feasible and efficient line balancing solutions and use ACO algorithm enhanced with several heuristics.

With the ABACO algorithm, it was aimed at minimising the total number of utilised workstations as well as the length of the line, as different from the common tendency in the literature. So that the user can easily define the importance of these two performance measures via the novel user interface of the algorithm. Also, the ACO related parameters (*i.e.* $\alpha$ $\beta$, $Q$, initial pheromone, evaporation rate, colony size, and maximum number of iterations), the problem-specific parameters (*i.e.* demands for the product models, planning horizon, and the names of input data files), and preferences about utilising multi-line stations (or merging stations, in other words) could be easily tuned thanks to the developed interface. Please note that this interface also includes some other features related to model sequencing options and GA parameters which will be explained in the following chapters.

Different numerical examples were generated using the test problems available in the literature, and solution building steps of the algorithm and related calculations were explained through these examples. The convergence of the objective value across the number of ants created was exhibited along with the total number of workstations and line length.

To check the performance of the proposed approach, 24 test cases were generated by modifying the existing test problems and solved with ABACO and other six heuristics. ABACO generated high-quality solutions in comparison with other six heuristics. In accordance with the results obtained, ABACO outperformed all heuristics for the majority of the problems solved. For the rest of the problems, ABACO found the same solution with other(s). Even more importantly, it is obvious that balancing lines together reduced the number of required operators at first glance. These results will be analysed and discussed comprehensively in Chapter 8. As an alternative to the given method in this chapter, a new algorithm which employs different model sequencing procedures is proposed for the solution of the problem in the next chapter.

# 6

# COPING WITH THE CHANGING MODEL COMBINATIONS IN THE MIXED-MODEL PARALLEL TWO-SIDED ASSEMBLY LINE BALANCING PROBLEMS

**Contents**

## 6.1. Chapter Introduction

The solutions obtained using the proposed ABACO approach in the previous chapter were independent of launched product model and convenient for any product model sequence. However, it is possible to have more powerful solutions for the same problem by taking product model changes at each production cycle into account and seeking balancing solutions which satisfy those product model sequences. For that purpose, an improved version of ABACO, called ABACO/S (where ‘S’ refers to the inclusion of model sequencing procedure), is developed and presented in this chapter. Procedures of the developed ABACO/S approach and the running principle of agents and other programme components are given in detail in Section 6.2. The multi-agent architecture of the agents is also presented in this section while a comprehensive numerical example is provided in Section 6.3. In Section 6.4, the performance of the developed algorithm is tested through a set of test problems and comparisons are made with the heuristic solutions of the same problems. The chapter is concluded with a summary in Section 6.5.

## 6.2. Procedures of the Developed ABACO/S Approach

This section describes the ABACO/S approach to solving the MPTALB/S problem. It starts explaining ABACO/S from the outermost level and continues with the basic programming components level-by-level.

In the previous chapter, the framework proposed to solve the MPTALB/S problem was used. However, in the algorithm developed in the previous chapter, the model sequencing problem, *i.e.* product model variation in each different production cycle, was ignored. Instead, a balancing solution, which would be feasible for any sequence or combination of the product models, was sought. For this aim, the processing time of a task was assumed to be the maximum time among the product models on the same line and the lines were balanced using maximum task times, like single-model lines. There is no doubt that such an approach is faster in terms of computational time but yields weak solutions as a generalised solution, which is independent of the launched sequence and product model combinations in production cycles, is obtained.

Different from the procedures and solution building mechanism presented in the previous chapter, this chapter takes the launched product model sequences and product model combinations in each production cycle into account. Therefore, line balancing problem is dealt with according to which product model is assembled on the workstation at a particular time. Moreover, the algorithm has a capability of sequencing product models through two different procedures; *(i)* combinatorial model sequencing, and *(ii)* random model sequencing (this will be explained later).

ABACO/S consists of four-level agents: *facilitator agent (FA), planning agent (PA), sequencing agent (SA)*, and *balancing agent (BA)*. These agents are programme scripts interacting with each other to solve the problem collectively. The outline and multi-agent architecture of the ABACO/S are displayed in Figure 6-1 and Figure 6-2, respectively.

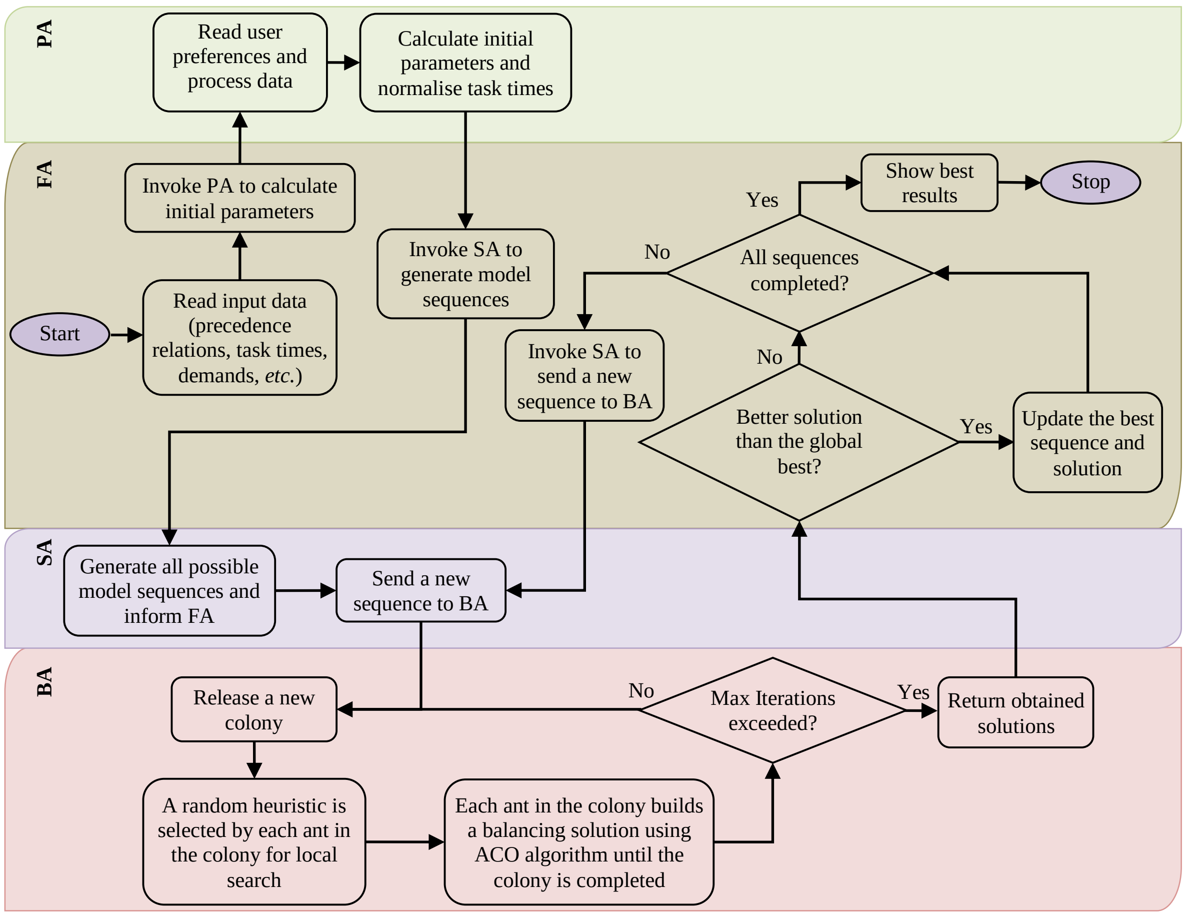

PA
Read user preferences and process data
Calculate initial parameters and normalise task times
FA
Invoke PA to calculate initial parameters
Show best results
Stop
Yes
No
All sequences completed?
Invoke SA to generate model sequences
Start
Read input data (precedence relations, task times, demands, *etc.*)
Invoke SA to send a new sequence to BA
No
Better solution than the global best?
Yes
Update the best sequence and solution
SA
Generate all possible model sequences and inform FA
Send a new sequence to BA
BA
Release a new colony
No
Max Iterations exceeded?
Yes
Return obtained solutions
A random heuristic is selected by each ant in the colony for local search
Each ant in the colony builds a balancing solution using ACO algorithm until the colony is completed

Figure 6-1. The outline of the ABACO/S algorithm

The algorithm starts with reading input data and related parameters by the FA. Then, FA invokes PA to process data and calculate initial parameters (*e.g.* normalising task times, calculating MPSs, *etc*.). Afterwards, PA stimulates FA and FA asks SA to generate product model sequences. SA generates all possible product model sequences and sends a product model sequence to BA to build a balancing solution based on the product model sequence sent. BA employs a colony of ants to find solutions, and ants in the colony build solutions until the maximum number of iterations is exceeded. Results are returned to FA and the global best solution is updated if a solution which is better than the current best solution is found. BA is sent another product model sequence and another colony find solutions for the launched product model sequence. This cycle continues until a pre-defined number of product model sequences are tried. When this is achieved, the programme shows the best product model sequence and the balancing solution found.

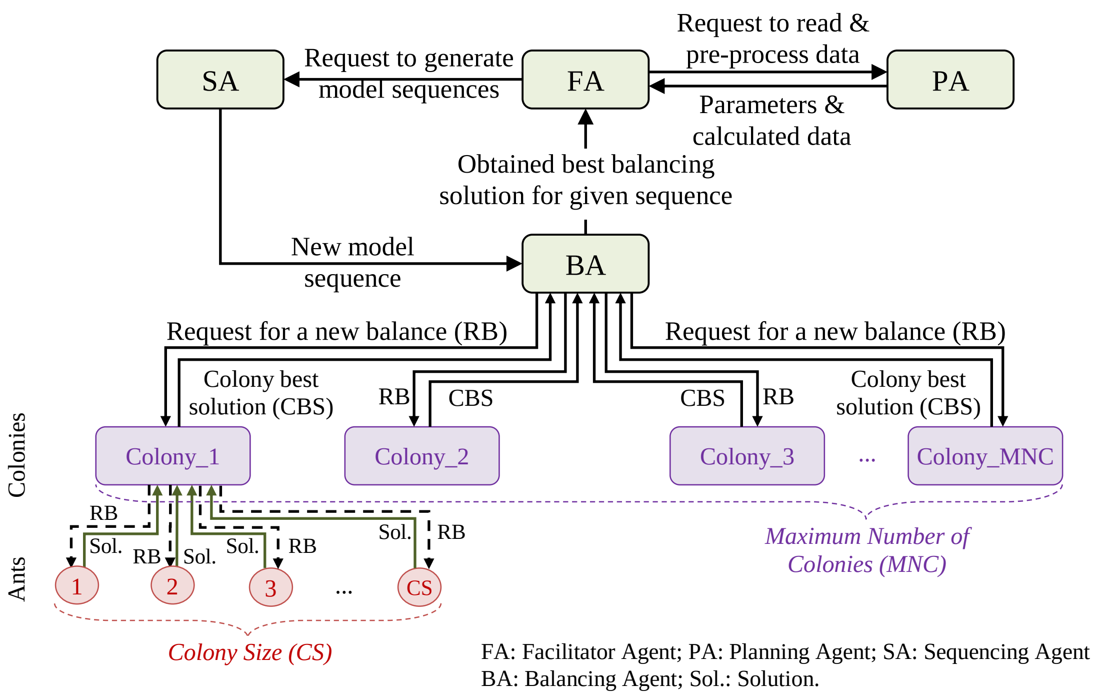

SA
Request to generate model sequences
FA
Request to read & pre-process data
PA
Parameters & calculated data
Obtained best balancing solution for given sequence
New model sequence
BA
Request for a new balance (RB)
Request for a new balance (RB)
Colony best solution (CBS)
RB
CBS
CBS
RB
Colony best solution (CBS)
Colonies
Colony_1
Colony_2
Colony_3
...
Colony_MNC
Maximum Number of Colonies (MNC)
Ants
RB
Sol.
RB
Sol.
Sol.
RB
Sol.
RB
1
2
3
...
CS
Colony Size (CS)
FA: Facilitator Agent; PA: Planning Agent; SA: Sequencing Agent
BA: Balancing Agent; Sol.: Solution.

Figure 6-2. The multi-agent architecture of the ABACO/S algorithm

It should be mentioned here that the algorithm has the capability of using combinatorial model sequencing and random model sequencing procedures. If the user prefers combinatorial model sequencing, the algorithm tries all possible product model sequences generated by the SA. If random model sequencing is preferred, the algorithm tries random product model sequences from possible product model sequences generated by the SA until a user defined number of trials are achieved. Additionally, it is also supported to find a balancing solution for a given product model sequence by the user.

The procedures of the ACO algorithm are exhibited in Figure 6-3. Each ant in the colony comes up with a solution and the performance measures are evaluated according to the quality of the obtained solution. Then, an amount of pheromone is laid on the edges of the path drawn (between task and workstation) in accordance with the performance measures. If a solution is better than the best solution in the colony, double amount of pheromone is laid on the edges of the solution to make the path favourable to be selected by other ants. A constant amount of pheromone is evaporated from all edges and the cycle continues until the colony is completed.

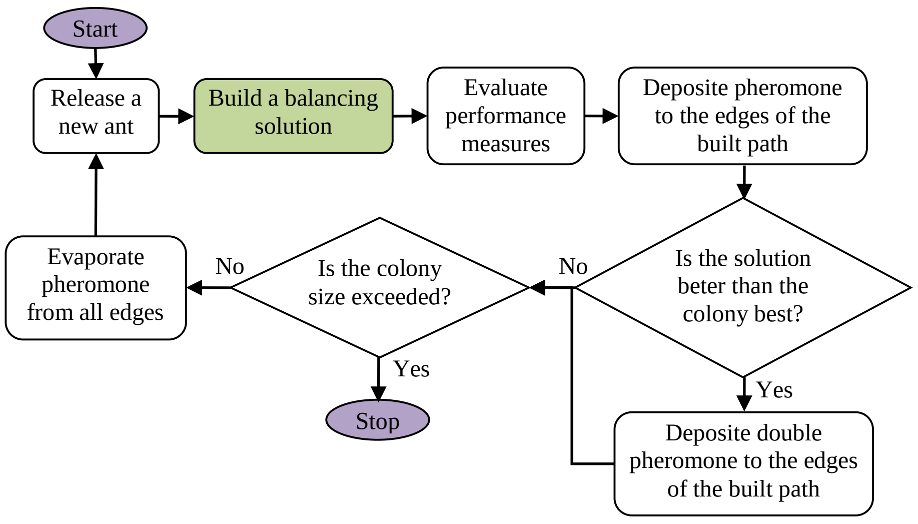

Start
Release a new ant
Build a balancing solution
Evaluate performance measures
Deposite pheromone to the edges of the built path
Is the solution beter than the colony best?
No
Yes
Deposite double pheromone to the edges of the built path
Is the colony size exceeded?
No
Yes
Evaporate pheromone from all edges
Stop

Figure 6-3. Procedures of the ACO

The same pheromone update rule with ABACO is used for ABACO/S:

$$\tau_{ik} \leftarrow (1-\rho)\tau_{ik} + \sum_{ant=1}^{CS} \Delta\tau_{ik}^{ant} \ , \qquad (6.1)$$

where $CS$ is colony size, $\Delta\tau_{ik}^{ant} = \frac{Q}{Performance\ measure}$ and is calculated based on the quality of the solution found by each ant; $\rho$ is evaporation rate, and $Q$ is a user determined parameter that affects the amount of pheromone deposited.

To find better solutions for the complex problem defined in Chapter 4, ABACO/S is enhanced with 10 different heuristics, most of them are commonly used in the literature. These heuristics will not be repeated here as they have already been explained in Chapter 5.

The topology of the proposed ant colony and the procedure of building a balancing solution are illustrated in Figure 6-4 and Figure 6-5. To provide a lean representation, only the first layer of paths and possible choices after task 1 are illustrated in Figure 6-4.

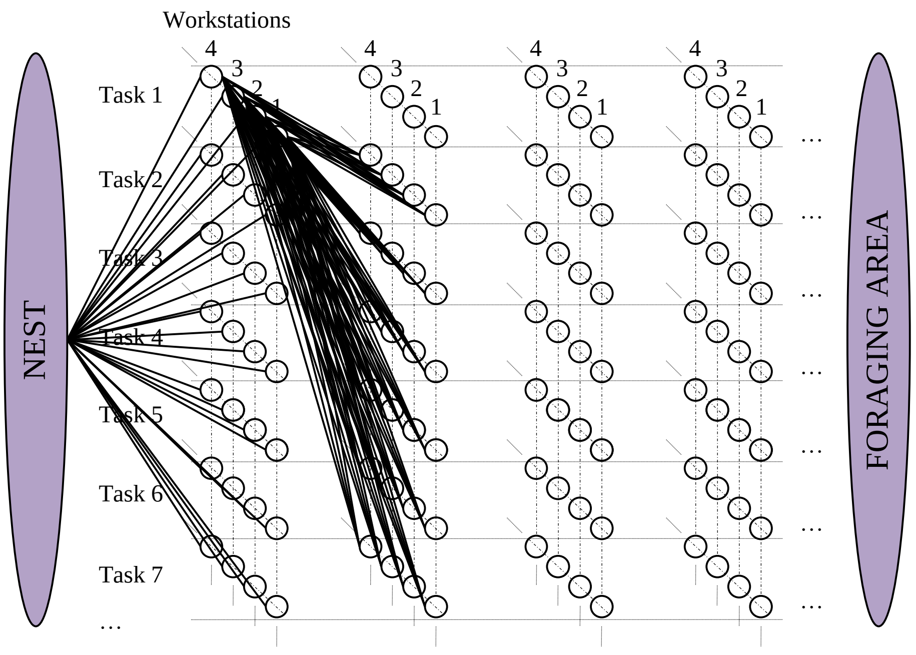


Figure 6-4. The topology of the proposed ACO algorithm

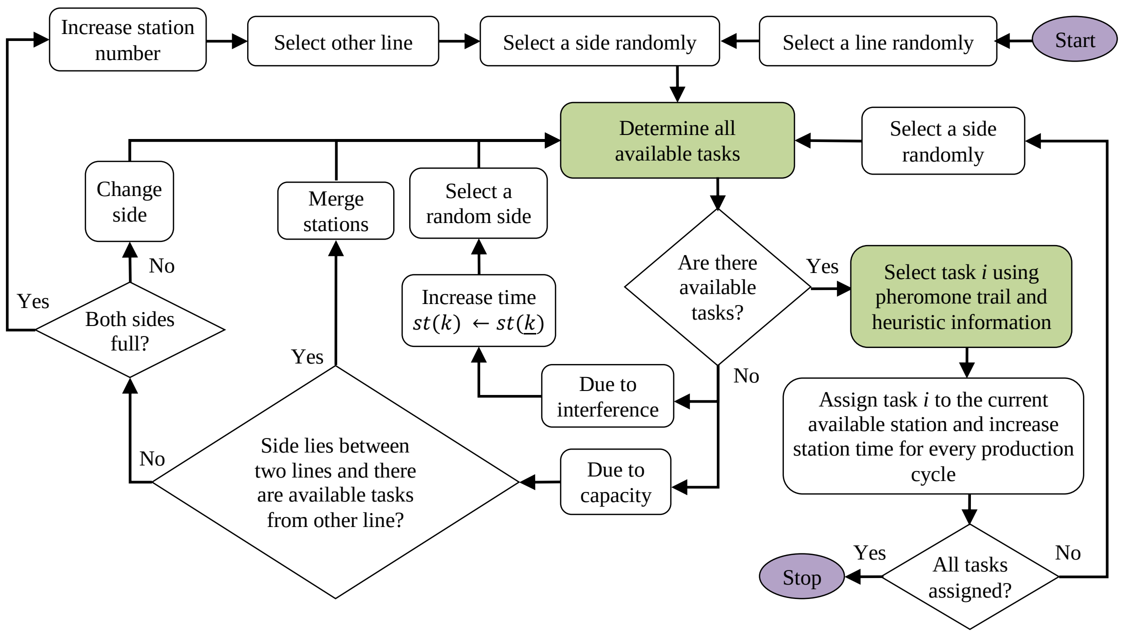

Start
Select a line randomly
Select a side randomly
Select other line
Increase station number
Determine all available tasks
Select a side randomly
Are there available tasks?
Yes
Select task *i* using pheromone trail and heuristic information
Assign task *i* to the current available station and increase station time for every production cycle
All tasks assigned?
Yes
Stop
No
No
Due to interference
Due to capacity
Increase time $st(k) \leftarrow st(\underline{k})$
Select a random side
Merge stations
Side lies between two lines and there are available tasks from other line?
Yes
No
Both sides full?
Yes
No
Change side

Figure 6-5. Building a balancing solution procedure of ABACO/S algorithm

Each ant in the colony builds a balancing solution according to the procedure given in Figure 6-5 (where $st(k)$ and $st(\underline{k})$ mean station time of the current station and its mated-station, respectively). Each ant starts from a random line and side and forwards by assigning tasks from the available tasks list to the current position. The selection probability of a task by an ant is calculated using the following equation:

$$p_{ik} = \frac{[\tau_{ik}]^{\alpha}[\eta_i]^{\beta}}{\sum_{y \epsilon Z_i}[\tau_{iy}]^{\alpha}[\eta_i]^{\beta}}\,, \tag{6.2}$$

where $i$, $k$, and $Z_i$ indicate task, current workstation, and the list of candidate tasks when task $i$ is selected, respectively. $\tau_{ik}$ and $\eta_i$ are the amount of virtual pheromone between task – workstation, and the heuristic information of task $i$ that comes from the randomly selected heuristic by each ant. This probability is calculated by each ant every time when a new task will be selected, and tasks having higher probability will most likely be selected and assigned by ants, primarily.

To increase the possibility of obtaining well-balanced solutions, ants can change their sides at any time. But, it is not allowed to forward to another line or queue without filling the capacity of current mated-stations as long as there are available tasks. Also, it is allowed to assign tasks from the contrary side of the adjacent line if the station lies between two lines. If there is not any available task (caused by the inadequate capacity, interference, *etc.*), a solution is sought depending on the reason as seen in Figure 6-5.

The flowchart of determining available tasks procedure is given in Figure 6-6. This process plays a critical role in the overall balancing and sequencing system. The reason is that the solution that will be obtained at the end of the balancing and sequencing procedure must be feasible in terms of different product model sequences, which change at every production cycle. That is why workloads of workstations and earliest starting times of tasks (caused by precedence relationships) must be recorded for every single production cycle. This data is used when determining whether a task is available or not, and processing time of a candidate task is considered according to the actual product model at the relevant cycle. Therefore, the processing time of a task for the relevant product model must be equal to or less than the remaining capacity at every production cycle. Also, earliest starting times of tasks must be considered carefully as they may differ from one cycle to another caused by the tasks' processing time differences depending on the assembled product model on the line.

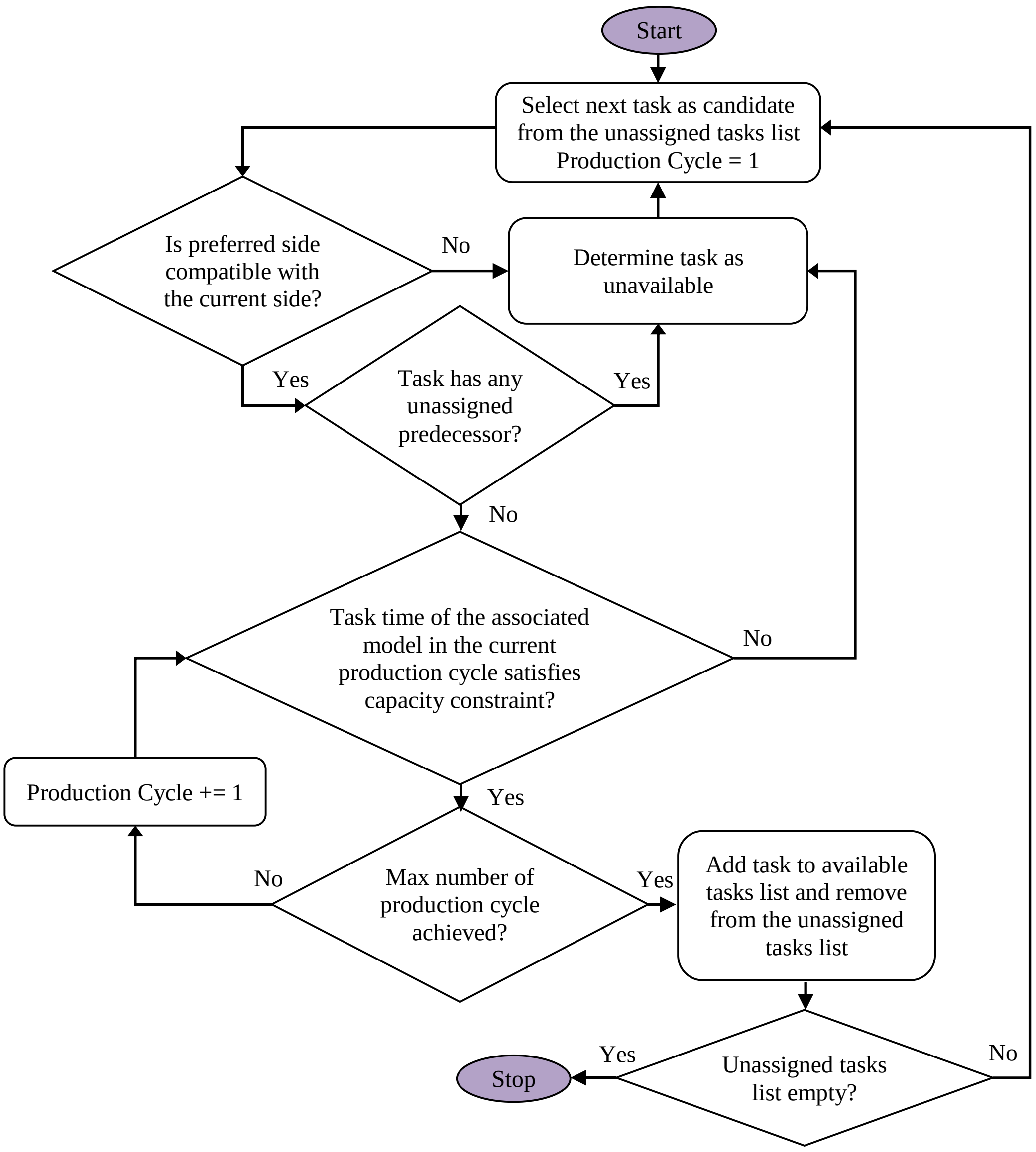

Start
Select next task as candidate from the unassigned tasks list Production Cycle = 1
Is preferred side compatible with the current side?
No
Determine task as unavailable
Yes
Task has any unassigned predecessor?
Yes
No
Task time of the associated model in the current production cycle satisfies capacity constraint?
No
Production Cycle += 1
Yes
No
Max number of production cycle achieved?
Yes
Add task to available tasks list and remove from the unassigned tasks list
Yes
Stop
Unassigned tasks list empty?
No

Figure 6-6. The procedure of determining available tasks

## 6.3. Numerical Example

This section briefly explains the simultaneous balancing and model sequencing procedure through a numerical example. Common precedence relationship diagrams are used between different product models on the same line. Two precedence relationship diagrams, P12 and P16 are taken from Kim *et al.* (2000c) and Lee *et al.* (2001), respectively. Task times are generated randomly between zero and 10 as given in Table 6-1 with the original preferred operation directions (where L means Left, R means Right, and E means Either side) and immediate predecessor tasks. If processing time of a task equals ‘0’ time units, it means that this task is not required to be performed for this product model.

Table 6-1. Task times (before normalisation) and relevant data for the numerical example

| Task no | Line I (P12) | | | | | Line II (P16) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | Side | Immediate predecessors | D | E | F | Side | Immediate predecessors |
| 1 | 4 | 2 | 6 | L | - | 5 | 7 | 4 | E | - |
| 2 | 8 | 10 | 7 | R | - | 0 | 4 | 0 | E | - |
| 3 | 3 | 5 | 3 | E | - | 5 | 10 | 7 | L | 1 |
| 4 | 4 | 2 | 0 | L | 1 | 4 | 8 | 2 | E | 1 |
| 5 | 3 | 1 | 2 | E | 2 | 3 | 4 | 8 | R | 2 |
| 6 | 1 | 6 | 0 | L | 3 | 1 | 2 | 3 | L | 3 |
| 7 | 2 | 0 | 2 | E | 4, 5 | 7 | 1 | 6 | E | 4, 5 |
| 8 | 5 | 6 | 6 | R | 5 | 4 | 4 | 5 | E | 6, 7 |
| 9 | 4 | 4 | 2 | E | 5, 6 | 2 | 2 | 1 | R | 7 |
| 10 | 2 | 5 | 0 | E | 7, 8 | 3 | 3 | 4 | R | 7 |
| 11 | 2 | 9 | 5 | E | 9 | 5 | 7 | 4 | E | 8 |
| 12 | 3 | 5 | 1 | R | 11 | 1 | 6 | 5 | L | 9 |
| 13 | - | - | - | - | - | 4 | 4 | 6 | E | 9, 10 |
| 14 | - | - | - | - | - | 5 | 2 | 3 | E | 11 |
| 15 | - | - | - | - | - | 0 | 4 | 1 | E | 11, 12 |
| 16 | - | - | - | - | - | 5 | 3 | 5 | E | 13 |

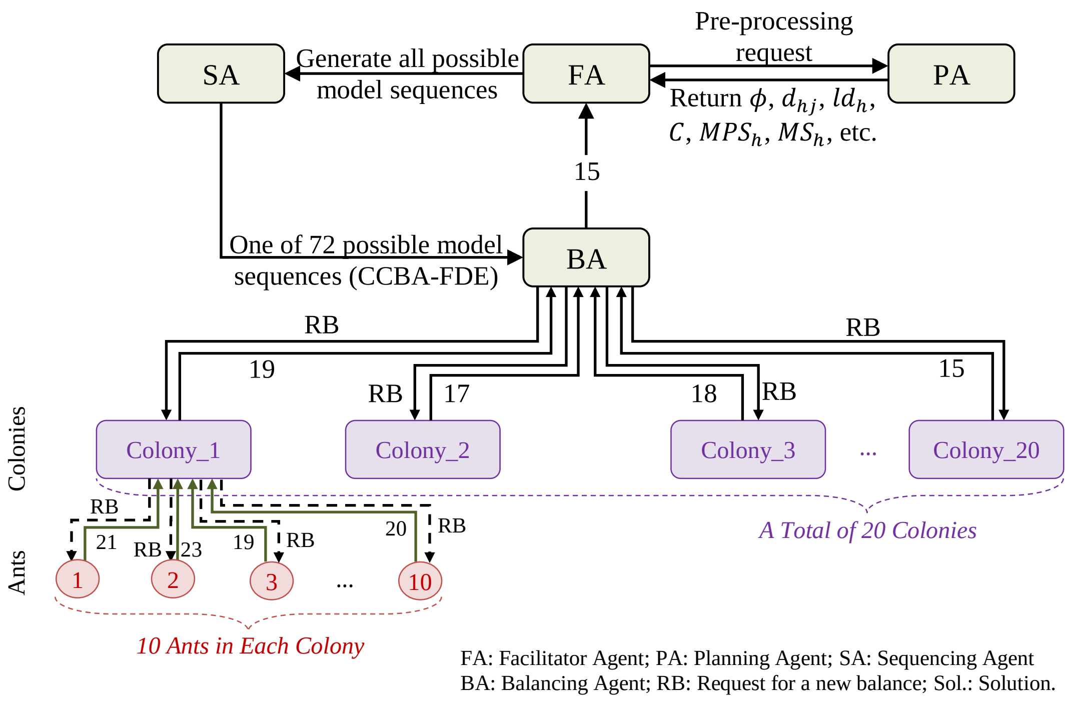
Product models A, B and C are assembled on Line I while product models D, E and F are assembled on Line II. If demands are assumed $D_{1A} = 8$, $D_{1B} = 8$, $D_{1C} = 16$, $D_{2D} = 8$, $D_{2E} = 8$ and $D_{2F} = 8$ for a planning horizon of 480 time units; cycle times are calculated as $C_1 = 480/(8+8+16) = 15$ and $C_2 = 480/(8+8+8) = 20$ time units; and minimum part sets are calculated as $MPS_1 = (8/8,\ 8/8,\ 16/8) = (1,1,2)$ and $MPS_2 = (8/8,\ 8/8,\ 8/8) = (1,1,1)$ for Line I and Line II, respectively. This means 12 different production cycles are subject to consideration for each product model sequence. The number of possible product model sequences for Line I is ${}^{4!}/_{(1! \times 1! \times 2!)} = 12$ and for Line II is ${}^{3!}/_{(1! \times 1! \times 1!)} = 6$. Thus, $12 \times 6 = 72$ different combinations of product model sequences must be tried in the case of combinatorial sequencing being selected. If random model sequencing is selected by the user, assuming that the number of sequences is determined as 18, 18 arbitrarily selected product model sequences will be tried out of 72 different combinations.

As the cycle times of the lines are different, the LCM based approach (Gökçen *et al.*, 2006, Ozcan *et al.*, 2010b) addressed in Chapter 2 is used. Line divisors ($ld_h$) are obtained as $ld_1 = LCM(C_1, C_2)/C_1 = 4$ and $ld_2 = LCM(C_1, C_2)/C_2 = 3$. Then, task times of product models on Line I and Line II are multiplied by $ld_1$ and $ld_2$, respectively, and $LCM(C_1, C_2)$ is accepted as the common cycle time ($C$). These normalised task times and common cycle time are used while balancing the lines.

As in the previous chapter, to provide more compact and easily understandable results, the following objective function is used when solving the example problem in this section and the test cases in the following sub-section:

$$Min\ Z = \gamma_1 LL + \gamma_2 NS\,, \tag{6.3}$$

where $LL$ and $NS$ indicate the *length of the lines* and the *total number of workstations*, respectively, for which the calculations have already been presented in Section 5.3.



Pre-processing request
SA
Generate all possible model sequences
FA
PA
Return $\phi$, $d_{hj}$, $ld_h$, $C$, $MPS_h$, $MS_h$, etc.
15
One of 72 possible model sequences (CCBA-FDE)
BA
RB
19
RB
17
18
RB
RB
15
Colonies
Colony_1
Colony_2
Colony_3
...
Colony_20
A Total of 20 Colonies
Ants
RB
21
RB
23
19
RB
20
RB
1
2
3
...
10
10 Ants in Each Colony
FA: Facilitator Agent; PA: Planning Agent; SA: Sequencing Agent
BA: Balancing Agent; RB: Request for a new balance; Sol.: Solution.

Figure 6-7. Interactions between agents on the multi-agent architecture of the proposed method

The algorithm was run for 20 random product model sequences, and 20 iterations for each product model sequence with 10 ants in a colony for the given example (the user defined weighting parameters are assumed $\gamma_1 = 2$, and $\gamma_2 = 1$). Agent behaviours and interactions between the agents are briefly shown in Figure 6-7 through the multi-agent architecture of the proposed method. This sample illustration is made for only the first colony of the first product model sequence $(CCBA - FDE)$ as it is not possible to show all of the steps of the solution procedure on a single figure. Facilitator agent invokes the SA until twenty product model sequences are completed and solution with the best performance measure is designated as the solution of the problem at the end of this process.

Task assignment order of the algorithm is shown in Figure 6-8 for the best solution found with product model sequences of $MS_1 = (ACCB)$ and $MS_2 = (EFD)$ on Line I and Line II, respectively. Arrows symbolise the assignment order of tasks across the lines. An ant starts assigning tasks from a randomly selected line and side (Line I side R in this example), and selects a task from the list of available tasks to assign to the current position. Then, the ant stays in the current side or changes side based on a random decision and selects another task from the updated list of available tasks for assignment. If the capacity is full or there is not any available task to assign for the current line, the line is changed and tasks are assigned to the new workstations on the new line using a similar procedure. As could be seen from the figure, task 12 (highlighted with asterisk$^*$) belonging to the product models assembled on Line I is assigned to a multi-line station on Line II in the given sample solution of the problem.

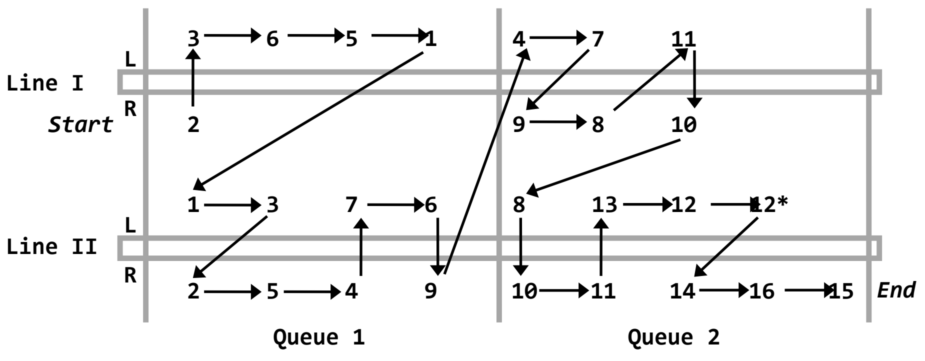


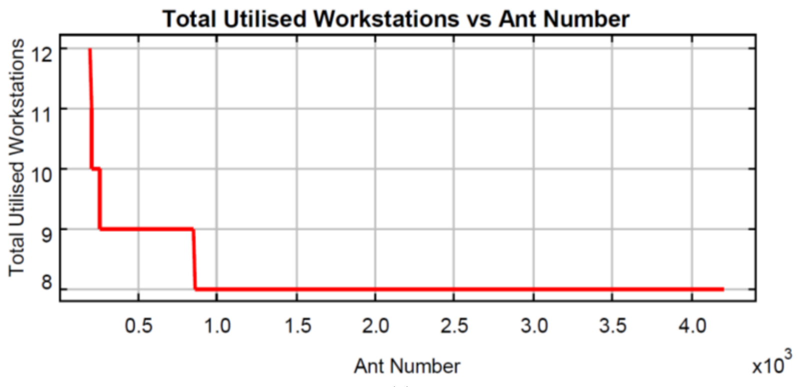
Figure 6-8. Assignment order of tasks for the given example

The different solution values obtained for different product model sequences of the example problem are also given in Table 6-2 according to the sequence (where ‘Objective’ means objective value). As could be seen from the table, the algorithm finds 15 as the objective value for the majority of the sequences, such as CCBA-FDE, CACB-FED, ACBC-DFE, *etc.* An objective value of 14 is found for the sequence CCAB-FDE and finally 12 is found for ACCB-EFD. It is obvious that better solutions could be investigated if the algorithm was run for more than 20 product model sequences. However, it was considered enough as this is just for an illustration.

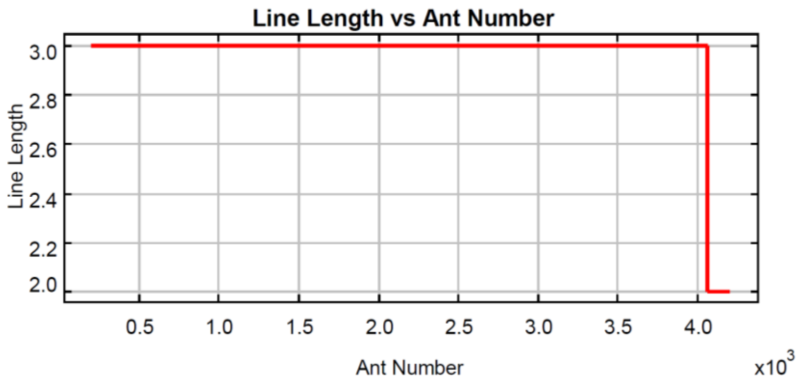
Table 6-2. Obtained solutions with different product model sequences for the given example

| # | Product model sequences | | Best solution | | | Average objective | Maximum objective |
|---|---|---|---|---|---|---|---|
| | Line I | Line II | LL | NS | Objective | | |
| 1 | CCBA | FDE | 3 | 9 | 15 | 17.85 | 23 |
| 2 | CACB | FED | 3 | 9 | 15 | 17.76 | 22 |
| 3 | ACBC | DFE | 3 | 9 | 15 | 17.62 | 22 |
| 4 | CCAB | FDE | 3 | 8 | 14 | 17.75 | 22 |
| 5 | CABC | FED | 3 | 9 | 15 | 18.00 | 22 |
| 6 | ABCC | EFD | 3 | 9 | 15 | 17.72 | 22 |
| 7 | CABC | DFE | 3 | 9 | 15 | 17.63 | 23 |
| 8 | CBCA | FED | 3 | 9 | 15 | 17.76 | 22 |
| 9 | CBCA | FDE | 3 | 9 | 15 | 17.73 | 24 |
| 10 | BACC | DFE | 3 | 9 | 15 | 17.75 | 22 |
| 11 | BACC | FDE | 3 | 9 | 15 | 17.72 | 23 |
| 12 | ABCC | EDF | 3 | 9 | 15 | 17.89 | 22 |
| 13 | CACB | DEF | 3 | 9 | 15 | 17.78 | 22 |
| 14 | CBAC | FDE | 3 | 9 | 15 | 17.73 | 23 |
| 15 | CBAC | EDF | 3 | 9 | 15 | 17.81 | 23 |
| 16 | CBCA | DEF | 3 | 9 | 15 | 17.73 | 23 |
| 17 | CCAB | EFD | 3 | 9 | 15 | 17.92 | 23 |
| 18 | CABC | EFD | 3 | 9 | 15 | 17.87 | 23 |
| 19 | CBCA | EDF | 3 | 9 | 15 | 17.82 | 22 |
| 20 | ACCB | EFD | 2 | 8 | 12 | 17.63 | 23 |

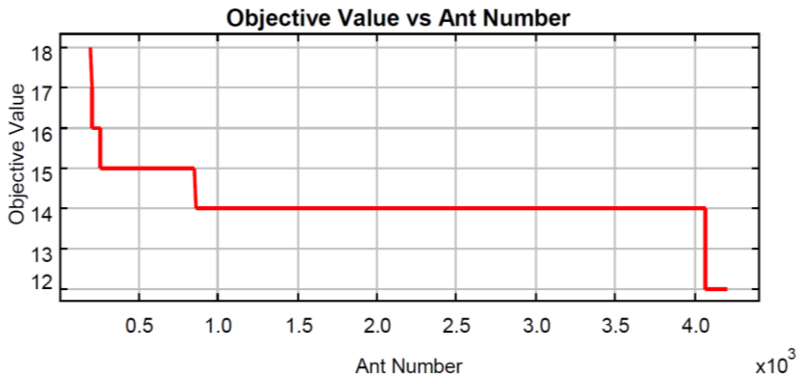
Figure 6-9 depicts the convergence of the performance measures, *i.e.* the total number of utilised workstations, line length and objective value, for the given example (where $\gamma_1 = 2$, and $\gamma_2 = 1$). As could be seen from Figure 6-9a, the total number of utilised workstations reduces from 12 to 9 dramatically with the first product model sequence (CCBA-FDE) given in Table 6-2, and reduces to 8 consequently. However, the line length remains the same (3 units) until the last as well as the best product model sequence (ACCB-EFD) among the obtained solutions when it reduces to 2 (Figure 6-9b). The curve of the objective function is mainly affected by the total number of utilised workstations till the last product model sequence and reaches the minimum with the reduction in the length of the line (Figure 6-9c). These graphs exhibit the effect of model sequencing on the quality of the obtained line balance, once again.



(a)



(b)



(c)

Figure 6-9. The convergence of the performance measures for the given example

## 6.4. Computational Experiments

The performance of the developed approach is compared against three heuristics gathered from the literature. Test cases are solved using the developed ABACO/S algorithm to evaluate the performance of the proposed approach. The test cases generated in Chapter 5 from the combination of commonly used test problems are solved using ABACO/S in this chapter. Product model demands and cycle times of the lines, which were generated in accordance with the original problem data in the previous chapter, are given in Table 6-3. To remind, no change is made for the precedence relationships diagrams and they are considered as the same as the original problems (*e.g.* P9, P24, *etc.*).

There is no comparable result available in the literature. Therefore, three heuristics (COMSOAL, Ranked Positional Weight Method - RPWM, and Maximum Number of Successors - MNS), which are commonly used in different formats in the literature to solve various line balancing problems, are also developed to solve the same test cases with ABACO/S. The heuristics designed here use the same balancing/sequencing procedures with ABACO/S. The only difference is that each heuristic applies its own rule to select and assign tasks to the workstations.

Table 6-3. Data for test cases

| **Test case** | **Problem** | | **Cycle time** | | **Demands Line I** | | | **Demands Line II** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Line I | Line II | Line I | Line II | A | B | C | D | E | F |
| 1 | P9 | P9 | 4 | 7 | 40 | 20 | 10 | 20 | 10 | 10 |
| 2 | P9 | P9 | 6 | 5 | 20 | 20 | 10 | 15 | 30 | 15 |
| 3 | P9 | P12 | 5 | 8 | 40 | 20 | 20 | 20 | 20 | 10 |
| 4 | P9 | P12 | 7 | 6 | 15 | 15 | 30 | 20 | 10 | 40 |
| 5 | P12 | P12 | 4 | 5 | 20 | 10 | 20 | 10 | 20 | 10 |
| 6 | P12 | P12 | 6 | 5 | 20 | 10 | 20 | 30 | 15 | 15 |
| 7 | P12 | P16 | 9 | 12 | 10 | 20 | 10 | 10 | 10 | 10 |
| 8 | P12 | P16 | 10 | 12 | 20 | 20 | 20 | 10 | 20 | 20 |
| 9 | P16 | P16 | 12 | 15 | 10 | 20 | 20 | 20 | 10 | 10 |
| 10 | P16 | P16 | 16 | 14 | 10 | 40 | 20 | 40 | 20 | 20 |
| 11 | P16 | P24 | 14 | 16 | 40 | 20 | 20 | 40 | 20 | 10 |
| 12 | P16 | P24 | 16 | 18 | 15 | 45 | 30 | 20 | 40 | 20 |
| 13 | P24 | P24 | 15 | 20 | 20 | 10 | 10 | 10 | 10 | 10 |
| 14 | P24 | P24 | 25 | 20 | 10 | 20 | 10 | 10 | 20 | 20 |
| 15 | A65 | A65 | 300 | 480 | 40 | 20 | 20 | 20 | 10 | 20 |
| 16 | A65 | A65 | 420 | 360 | 15 | 15 | 30 | 20 | 40 | 10 |
| 17 | A65 | B148 | 405 | 810 | 10 | 5 | 5 | 4 | 4 | 2 |
| 18 | A65 | B148 | 675 | 540 | 20 | 10 | 10 | 10 | 20 | 20 |
| 19 | B148 | B148 | 255 | 510 | 5 | 10 | 5 | 2 | 4 | 4 |
| 20 | B148 | B148 | 425 | 340 | 20 | 10 | 10 | 20 | 20 | 10 |
| 21 | B148 | A205 | 510 | 1020 | 10 | 5 | 15 | 3 | 6 | 6 |
| 22 | B148 | A205 | 600 | 1200 | 6 | 3 | 3 | 2 | 2 | 2 |
| 23 | A205 | A205 | 1200 | 1200 | 10 | 10 | 10 | 10 | 10 | 10 |
| 24 | A205 | A205 | 1000 | 2000 | 15 | 5 | 10 | 5 | 5 | 5 |

A 3.1 GHz Intel Core$^{TM}$ i5-2400 CPU computer is used to run ABACO/S and three heuristics coded in JAVA$^{TM}$ Standard Edition 7u4 environment. The parameters of ABACO/S are chosen through a set of experimental tests for a high-quality solution and may differ from one problem to another. This is why the search space grows exponentially and the complexity increases with the increasing number of tasks. For test heuristics (COMSOAL, RPWM and MNS), 15, 20 and 40 randomly selected product model combinations were tried for test cases #1–#6, #7–#14 and #15–#24, respectively. The best solution was taken after the algorithm was run 20 times for each product model sequence for the test cases #1–#10; and 30 times for each product model sequence for the test cases #11–#24. For ABACO/S, initial pheromone level, colony size, the number of iterations, the number of random product model sequences tried and other parameters were set in accordance with the increasing problem size (see Table 6-4).

Table 6-4. Parameters of the ABACO/S

| Test case | $\alpha$ | $\beta$ | $\rho$ | $Q$ | Initial pheromone | Colony size | Number of iterations | Number of sequences |
|---|---|---|---|---|---|---|---|---|
| 1-6 | 0.1 | 0.2 | 0.1 | 50 | 10 | 10 | 10 | 15 |
| 7-14 | 0.1 | 0.2 | 0.1 | 50 | 15 | 20 | 20 | 20 |
| 15-24 | 0.1 | 0.2 | 0.1 | 50 | 20 | 30 | 30 | 40 |

Table 6-5 exhibits the results obtained by ABACO/S and three heuristics for test cases (the user defined parameters in the objective function are considered as $\gamma_1 = 2$, and $\gamma_2 = 1$). Test cases are solved under various cycle time constraints with different product model demands and obtained results are compared with respect to line lengths (*LL*), number of workstations (*NS*), and objective values (*OBJ*). As can be seen from the results table, it is possible to have balancing solutions with the same performance measures for different product model sequences. For example, all four approaches find balancing solutions with the same performance measures ($LL = 3$, $NS = 9$, and $OBJ = 15$) for different product model sequences in test case #2.

According to the computational results, COMSOAL finds good solutions for small-sized test cases but not large-sized ones. The overall performances of RPWM and MNS vary while MNS finds a better solution than others for the test case #17. The algorithms find similar solutions, even same for some cases such as #2 and #6–#9.

Table 6-5. Computational results

| | **COMSOAL** | | | | **RPWM** | | | | **MNS** | | | | **ABACO/S** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *#* | *LL* | *NS* | *OBJ* | *Model sequence Line I – Line II* | *LL* | *NS* | *OBJ* | *Model sequence Line I – Line II* | *LL* | *NS* | *OBJ* | *Model sequence Line I – Line II* | *LL* | *NS* | *OBJ* | *Model sequence Line I – Line II* |
| 1 | **3** | **10** | **16** | **CBAAAAB-EDDF** | 4 | 11 | 19 | BABACAA-EDFD | 3 | 11 | 17 | AAABCAB-EDDF | **3** | **10** | **16** | **AABAABC-FDED** |
| 2 | 3 | 9 | 15 | AACBB-DEEF | 3 | 9 | 15 | BACAB-EDEF | 3 | 9 | 15 | BBAAC-EDFE | 3 | 9 | 15 | CAABB-EDFE |
| 3 | **3** | **8** | **14** | **BACA-EEDDF** | 3 | 9 | 15 | BAAC-DEEFD | **3** | **8** | **14** | **CBAA-DDFEE** | **3** | **8** | **14** | **AABC-DEFDE** |
| 4 | **2** | **8** | **12** | **BCAC-FEDFDFF** | 3 | 9 | 15 | CCAB-FFDEDFF | 3 | 9 | 15 | BACC-FDDFFEF | **2** | **8** | **12** | **BCAC-DFFFFED** |
| 5 | 4 | 13 | 21 | BACAC-DFEE | 4 | 13 | 21 | CCABA-DEEF | 4 | 13 | 21 | CCAAB-FEED | **4** | **12** | **20** | **CCAAB-EDEF** |
| 6 | 3 | 10 | 16 | ABACC-EDFD | 3 | 10 | 16 | CBCAA-FDDE | 3 | 10 | 16 | ACBAC-DFED | 3 | 10 | 16 | ABACC-DFDE |
| 7 | 7 | 17 | 31 | BCAB-DEF | 7 | 17 | 31 | ABCB-DFE | 7 | 17 | 31 | CABB-FDE | 7 | 17 | 31 | ACBB-CBA |
| 8 | 7 | 17 | 31 | ABC-FEEFD | 7 | 17 | 31 | CAB-EFDFE | 7 | 17 | 31 | BCA-FEFDE | 7 | 17 | 31 | BCA-DEFFE |
| 9 | 7 | 23 | 37 | ACCBB-EDDF | 7 | 23 | 37 | ABBCC-FDED | 7 | 23 | 37 | ACCBB-FEDD | 7 | 23 | 37 | CABCB-EFDD |
| 10 | 7 | 21 | 35 | ABBCCBB-EDDF | **7** | **20** | **34** | **ABCBBCB-DEDF** | 7 | 21 | 35 | ACBCBBB-DEDF | **7** | **20** | **34** | **CCBBABB-DEFD** |
| 11 | 7 | 22 | 36 | BCAA-DEFDDED | 6 | 24 | 36 | BCAA-EEDFDDD | 7 | 23 | 37 | BACA-DDFDEDE | **6** | **22** | **34** | **ABAC-EFDEDDD** |
| 12 | 5 | 18 | 28 | BCBACB-EDFE | 5 | 18 | 28 | CBABBC-FDEE | **5** | **17** | **27** | **BCBCAB-FEDE** | **5** | **17** | **27** | **BBCCBA-FEDE** |
| 13 | 6 | 20 | 32 | ACAB-FED | 6 | 21 | 33 | AACB-DFE | 6 | 20 | 32 | CABA-DEF | **5** | **19** | **29** | **ABCA-EFD** |
| 14 | 4 | 15 | 23 | BACB-FEFDE | 4 | 15 | 23 | CBBA-FFEDE | **4** | **14** | **22** | **CABB-FDFEE** | **4** | **14** | **22** | **CABB-EFEFD** |
| 15 | 12 | 43 | 67 | ABAC-DFDFE | 11 | 39 | 61 | BACA-DFEFD | 12 | 40 | 64 | BCAA-DEDFF | **11** | **38** | **60** | **BACA-EFFDD** |
| 16 | 10 | 40 | 60 | ACCB-FEEDDEE | 10 | 38 | 58 | CCAB-DEDEFEE | 10 | 38 | 58 | CABC-FEEEDDE | **10** | **37** | **57** | **CBCA-FDEEDEE** |
| 17 | 9 | 33 | 51 | CAAB-DDFEE | 9 | 32 | 50 | AABC-EDEFD | **9** | **31** | **49** | **BACA-FDEDE** | 9 | 32 | 50 | AABC-EEDFD |
| 18 | 10 | 35 | 55 | AACB-FDFEE | **9** | **31** | **49** | **ABAC-EEFDF** | 10 | 32 | 52 | AABC-FFEED | **9** | **31** | **49** | **ACBA-EFFED** |
| 19 | 23 | 72 | 118 | CABB-EFEDF | 21 | 69 | 111 | ABBC-FEEFD | 21 | 68 | 110 | BABC-EFFED | **18** | **65** | **101** | **ABCB-EEFDF** |
| 20 | 17 | 62 | 96 | ABAC-DEDFE | 16 | 60 | 92 | BCAA-DDEEF | 16 | 59 | 91 | ACBA-FDEED | **15** | **58** | **88** | **BACA-DEDEF** |
| 21 | 19 | 67 | 105 | ABCCAC-EDEFF | 18 | 61 | 97 | CACACB-EEFFD | 18 | 62 | 98 | CCAABC-FEFED | **17** | **61** | **95** | **BCAACC-EFEFD** |
| 22 | 17 | 57 | 91 | BACA-EFD | 15 | 54 | 84 | BAAC-FED | 16 | 53 | 85 | CAAB-FDE | **15** | **53** | **83** | **AACB-FDE** |
| 23 | 18 | 70 | 106 | ABC-FED | 16 | 63 | 95 | BAC-EDF | 16 | 63 | 95 | CBA-EDF | **16** | **62** | **94** | **ACB-FED** |
| 24 | 19 | 64 | 102 | ACCBAA-FDE | 17 | 61 | 95 | BACCAA-DEF | 17 | 60 | 94 | BACAAC-FED | **17** | **59** | **93** | **CBCAAA-FED** |

Table 6-6. Comparison of results obtained by ABACO and ABACO/S

| | ***Test case*** | ***1*** | ***2*** | ***3*** | ***4*** | ***5*** | ***6*** | ***7*** | ***8*** | ***9*** | ***10*** | ***11*** | ***12*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ABACO** | *LL* | 4 | 3 | 3 | 3 | 4 | 3 | 7 | 7 | 7 | 7 | 7 | 5 |
| | *NS* | 12 | 11 | 10 | 10 | 14 | 11 | 18 | 18 | 25 | 21 | 25 | 19 |
| | *OBJ* | 20 | 17 | 16 | 16 | 22 | 17 | 32 | 32 | 39 | 35 | 39 | 29 |
| | ***Test case*** | ***13*** | ***14*** | ***15*** | ***16*** | ***17*** | ***18*** | ***19*** | ***20*** | ***21*** | ***22*** | ***23*** | ***24*** |
| | *LL* | 7 | 5 | 14 | 13 | 11 | 12 | 24 | 20 | 22 | 19 | 21 | 22 |
| | *NS* | 24 | 17 | 49 | 46 | 40 | 41 | 85 | 77 | 79 | 67 | 81 | 76 |
| | *OBJ* | 38 | 27 | 77 | 72 | 62 | 65 | 133 | 117 | 123 | 105 | 123 | 120 |
| **ABACO/S** | ***Test case*** | ***1*** | ***2*** | ***3*** | ***4*** | ***5*** | ***6*** | ***7*** | ***8*** | ***9*** | ***10*** | ***11*** | ***12*** |
| | *LL* | 3 | 3 | 3 | 2 | 4 | 3 | 7 | 7 | 7 | 7 | 6 | 5 |
| | *NS* | 10 | 9 | 8 | 8 | 12 | 10 | 17 | 17 | 23 | 20 | 22 | 17 |
| | *OBJ* | 16 | 15 | 14 | 12 | 20 | 16 | 31 | 31 | 37 | 34 | 34 | 27 |
| | ***Test case*** | ***13*** | ***14*** | ***15*** | ***16*** | ***17*** | ***18*** | ***19*** | ***20*** | ***21*** | ***22*** | ***23*** | ***24*** |
| | *LL* | 5 | 4 | 11 | 10 | 9 | 9 | 18 | 15 | 17 | 15 | 16 | 17 |
| | *NS* | 19 | 14 | 38 | 37 | 32 | 31 | 65 | 58 | 61 | 53 | 62 | 59 |
| | *OBJ* | 29 | 22 | 60 | 57 | 50 | 49 | 101 | 88 | 95 | 83 | 94 | 93 |

With the increasing task numbers of the test cases, ABACO/S finds better solutions than others; see test cases #5, #11, #13, #15, #16 and #19–#24. Therefore, it is observed that ABACO/S finds solutions better than or equal to those found by the three heuristics except one test case (test case #17, in which the best solution is investigated by the MNS for).

Consequently, the results indicate that the developed algorithm has a good solution capacity for the MPTALB/S problem and outperforms the three heuristics for the experimented test cases in this research. The effect of solving model sequencing and line balancing problems simultaneously could also be distinguished easily when the results obtained by ABACO/S in this chapter are compared to those obtained by ABACO in Chapter 5.

Table 6-6 gives the comparison of situations when the model sequencing problem is simultaneously considered (see ABACO/S column) and when it is not considered (see ABACO column), for the same test cases. In the table, the ABACO row denotes the results obtained in the previous chapter where only the line balancing problem is solved by utilising multi-line stations. On the other hand, the ABACO/S row reports results from the ABACO/S where the model sequencing problem was integrated to the line balancing problem by considering multi-line stations. At first glance, there is no doubt that the solutions found by ABACO/S are better than those obtained by ABACO for the same test cases. Although the solutions of ABACO have more flexibility and may suit any product model sequence launched, more efficiency is provided and operator requirements are minimised in the solutions of ABACO/S. Please note that a more comprehensive analysis and discussion of these results will be provided in Chapter 8.

## 6.5. Chapter Summary

This chapter proposed a novel algorithm to cope with the changing product model sequences at every production cycle of the lines. For that purpose, a new agent was deployed to generate all possible product model sequences and to launch a new one among them when requested, unlike the method used in the previous chapter (for ABACO). Existing balancing agent was modified to carefully consider the processing time of the particular task which belongs to the product model being assembled on the line at that time.

The performance of the algorithm was enhanced with the integrated heuristics, each of which is a commonly used individual technique in this domain. To build a complete solution, different programme scripts (called agents) collaborate with each other. Initialisation and associated calculations are carried out by facilitator agent and planning agents. The sequencing agent generates different model sequencing patterns and balancing agent releases ant colonies to build solutions in accordance with the generated patterns. A numerical example was given to explain the calculation of initial parameters and solution building procedures of the algorithm. A couple of results were provided with different product model sequences for the same example problem. The multi-agent architecture of the ABACO/S algorithm was also presented with an example.

To compare the performance of the proposed algorithm in solving the MPTALB/S problem, 24 test problems were solved with ABACO/S and with three heuristics, respectively, and the results were reported. The results indicated that ABACO/S outperforms other heuristics in terms of the performance measures sought.

Moreover, a comparison was provided between balancing the mixed-model parallel two-sided assembly lines with and without the simultaneous model sequencing problem. It was found that considering the model sequencing problem together with the line balancing problem provides advantages, such as minimising the required number of operators and line length. Thus, it was demonstrated that the sequence of product models is a significant factor that affects the efficiency of the lines as well as task sequencing. As processing times of tasks may vary from one product model to another, the sequence of product models on the line influences the availability of operators, who perform their jobs in multi-line workstations. This is another practical finding of the research carried out within the scope of this research.

# 7

# INTEGRATION OF A GA-BASED MODEL SEQUENCING PROCEDURE TO ABACO/S

**Contents**

## 7.1. Chapter Introduction

This chapter proposes ways to robustly solve the MPTALB/S problem and to increase the quality of the solutions obtained further on the developed ABACO/S approach, which considers changing product model combinations from one production cycle to another. A GA-based model sequencing procedure is depicted to be integrated with the ABACO/S and to enhance its model sequencing mechanism. A brief background on the hybridisation of GA and ACO algorithms is given in Section 7.2. The proposed ABACO/S-GA solution method is depicted in Section 7.3 and Section 7.4 by illustrating its component architecture and the main module characteristics. A numerical example is given in Section 7.5 to simulate the solution building procedure of the algorithm and the evolution of chromosomes while the quantitative assessment of the performance of the developed algorithm through test problems is provided in Section 7.6. Finally, Section 7.7 concludes the chapter with a brief summary.

## 7.2. Brief Background

In the ABACO/S presented in Chapter 6, the user is allowed to choose between two different schemes provided for the determination of product model sequences: *(i)* combinatorial sequencing and *(ii)* random sequencing. If combinatorial sequencing is chosen, the algorithm generates all possible product model sequences for assembly lines and tries each product model sequence combination one-by-one. If the second option (random sequencing) is chosen by the user, the algorithm tries a user defined number of random product model sequences. While the former option increases the possibility of obtaining a well-balanced solution, the latter one returns solutions faster than the other. Therefore, a model sequencing mechanism is needed to find a balance between these two options and to obtain good quality solutions in a reasonable computational time. For this aim, a GA-based model sequencing procedure is integrated to the ABACO/S and a new hybrid *agent based ant colony optimisation – genetic algorithm* (ABACO/S-GA) approach is proposed for the solution of MPTALB/S problem in this chapter.

ACO and GA are well-known and widely applied meta-heuristics in solving combinatorial optimisation problems, and engineering optimisation problems in particular. While ACO mimics the foraging behaviour of real ant colonies in nature, GA mimics the natural selection and evolution of individuals (Costa *et al.*, 2014, Ugarte *et al.*, 2011). The basic principles of ACO and GA are laid down by Dorigo *et al.* (1996) and Holland (1975), respectively. Since then, these two meta-heuristics attracted researchers and have been enhanced in terms of solution capacity and efficiency. Recent advances could be found from Cordon *et al.* (2002) and Blum (2005) for the ACO technique; and Srinivas and Patnaik (1994) and Li *et al.* (2011b) for the GA technique. Moreover, Tasan and Tunali (2008) reviewed current applications of GAs in assembly line balancing domain.

ACO and GA have been hybridised successfully for a wide range of problems, such as Lee *et al.* (2008), Chen and Chien (2011), Li *et al.* (2011a), and Akpinar *et al.* (2013). Among these hybridisation attempts, the method of Akpinar *et al.* (2013) was used to solve mixed-model assembly line balancing problem with sequence dependent setup times between tasks. The algorithm proposed by Akpinar *et al.* (2013) aimed at enhancing the solution building procedure of ACO by incorporating GA as a local search strategy.

Based on this motivation, GA is integrated to the ABACO/S to enhance the capability of the algorithm while decreasing the needed computational effort. In the following sub-sections, the ABACO/S-GA approach developed in this research will be described.

## 7.3. The outline of Proposed Approach

ABACO/S-GA consists of four-level agents: *planning agent (PA)*, *facilitator agent (FA)*, *sequencing agent (SA)*, and *balancing agent (BA)*, which interact with each other to solve the problem collectively. The algorithm systematically explores less number of workstations and shorter line length to find a balancing solution for different candidate product model sequences generated by the integrated GA mechanism in SA. For each candidate product model sequence, ACO algorithm attempts to find the best task assignment. The outline of the developed algorithm is depicted in Figure 7-1.

The input data (*e.g.* task times, precedence relationships, product model demands, *etc.*), algorithmic parameters (*e.g.* ACO parameters such as $\alpha$, $\beta$, $\rho$, *etc.*; GA parameters such as population size, crossover rate, mutation rate, *etc.*), and user preferences (such as objective function weights) are read by the FA and the algorithm is initialised.

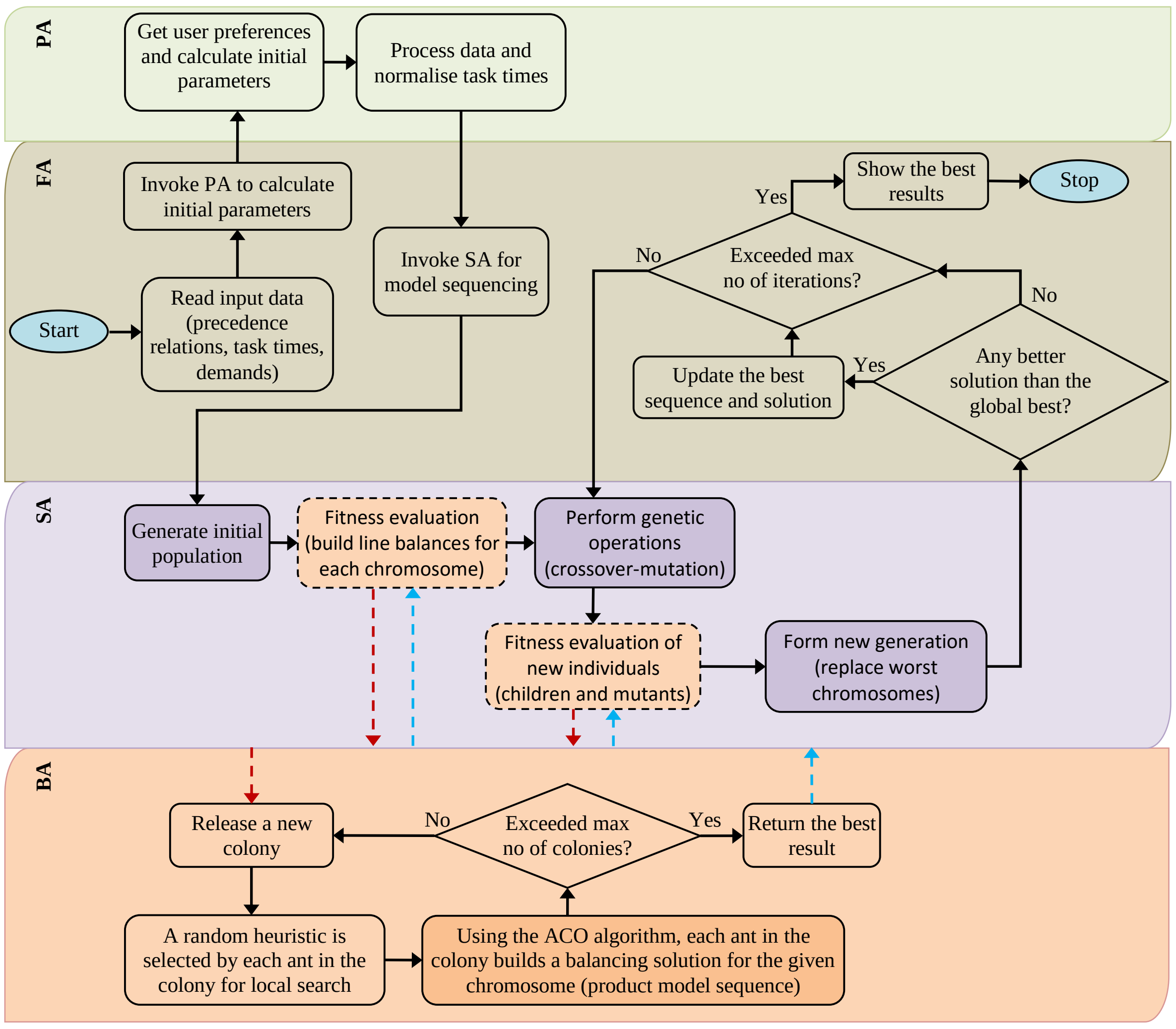

PA
Get user preferences and calculate initial parameters
Process data and normalise task times
FA
Invoke PA to calculate initial parameters
Show the best results
Stop
Yes
No
Exceeded max no of iterations?
No
Invoke SA for model sequencing
Read input data (precedence relations, task times, demands)
Start
Yes
Any better solution than the global best?
Update the best sequence and solution
SA
Generate initial population
Fitness evaluation (build line balances for each chromosome)
Perform genetic operations (crossover-mutation)
Fitness evaluation of new individuals (children and mutants)
Form new generation (replace worst chromosomes)
BA
Release a new colony
No
Exceeded max no of colonies?
Yes
Return the best result
A random heuristic is selected by each ant in the colony for local search
Using the ACO algorithm, each ant in the colony builds a balancing solution for the given chromosome (product model sequence)

Figure 7-1. The outline of the ABACO/S-GA approach

Intermediate parameters (*i.e.* greatest common divisors of product model demands, least common multiple of cycle times) are calculated and data is processed (*e.g.* task times are normalised, common cycle time is calculated, minimum part sets are determined, *etc.*) to make it ready for using by other programme components. The initial population is generated by SA, where each chromosome represents a complete product model sequence. SA requests fitness evaluation of the chromosomes in the population from the BA. BA employs ACO to build balancing solutions for given chromosomes. The fitness value of each balancing solution, which corresponds to the objective function value of each individual, is calculated using Equation (5.3) – Equation (5.5) presented in Chapter 5 and returned to SA. The best balancing solution which gives the best fitness value for each chromosome (or product model sequence) is selected as the best solution for this chromosome. Genetic operators perform crossover and mutation procedures for the selected chromosomes and the fitness values of the newly obtained children (formed after crossover procedure) and mutants (formed after mutation procedure) are evaluated by the BA again. The new generation is shaped by replacing the worst chromosomes with the best ones among the children and mutants (if any). Genetic operations are performed again for the selected individuals in the population and the new generation is formed in accordance with their fitness values (or objective function values). This cycle is repeated until a predetermined number of iterations are exceeded. The best solution is updated if a better solution is found during this cycle and is shown when the algorithm stops. The procedures of the GA (for model sequencing) are explained in the following sub-sections in more detail. Please note that the procedures of ACO, building a balancing solution, and determining available tasks used for line balancing will not be repeated here as they were already presented comprehensively in Chapter 6.

## 7.4. GA for Model Sequencing

The simulation of the employed GA is given in Figure 7-2. Initial population is generated by building up feasible chromosomes randomly, considering the number of chromosomes that the population must have (population size). A chromosome is made up with the ordered product model types according to the demand and minimum part sets calculated. To build a feasible chromosome, product model types that belong to Line I are located to the head of the chromosome while product model types for Line II are allocated to the tail of the chromosome. Each product model appears on the chromosome as the number of times it shows up in the product model sequence ($MS_h$). Thus, chromosome length equals to the sum of the length of the product model sequences produced on both of the lines. A chromosome sample for a given product model sequence of $MS_1 = ACAB$ and $MS_2 = EDDFE$ is exhibited in Figure 7-3.

For the evolution of the individuals, one-point crossover and two-gene mutation operators are applied on randomly selected chromosomes from the population. Figure 7-4 represents the running principle of crossover procedure used in the algorithm. The place where the product models belonging to Line I finishes and the product models of Line II starts is determined as the cutting point of the chromosome and the head and tail parts of the parent chromosomes are matched crosswise to acquire new offspring.

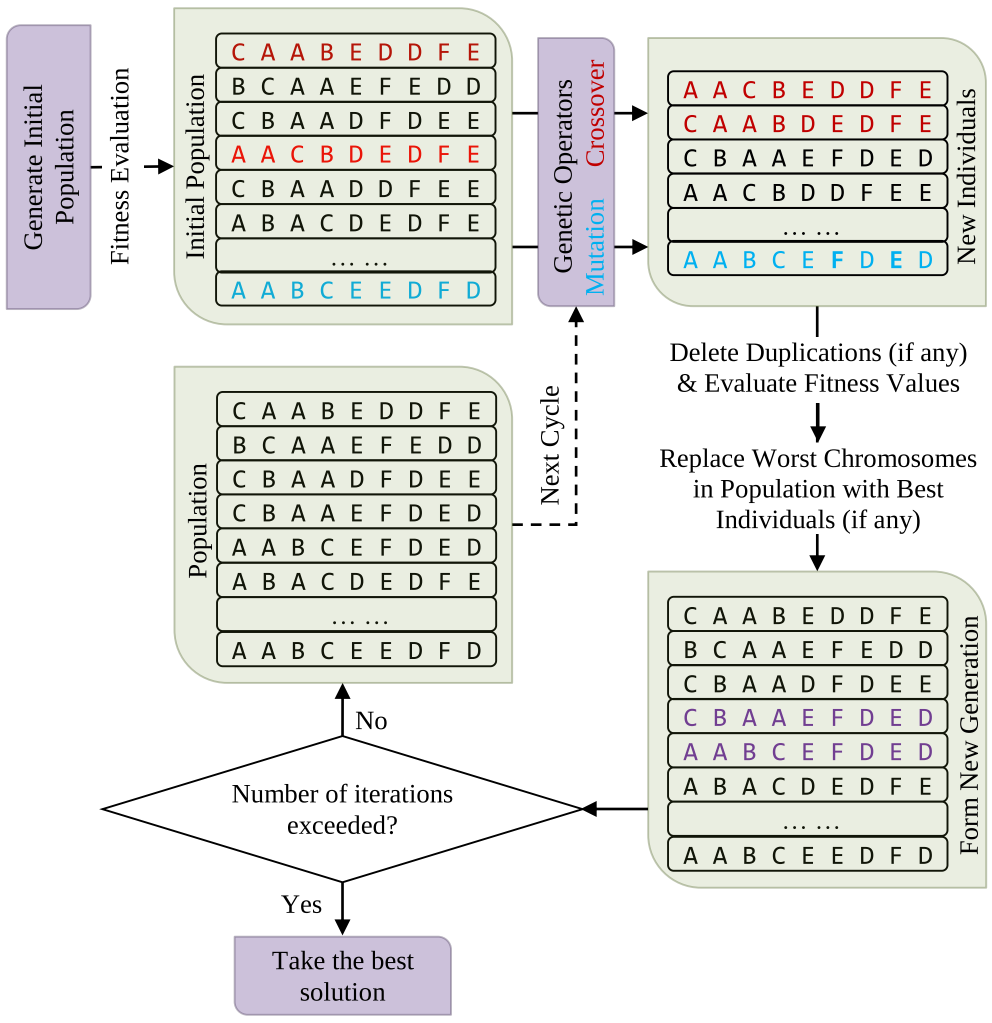


Figure 7-2. Simulation of the employed GA

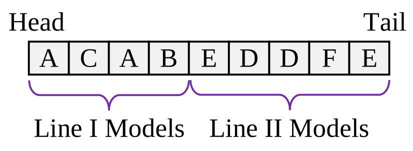


Figure 7-3. Representation of a real-coded chromosome sample

Mutation procedure is conducted by swapping two randomly selected genes within the same zone which is split by the cutting point. In other words, if a gene is selected before the cutting point, it is swapped by another randomly selected gene before the cutting point, or vice versa (see Figure 7-5).

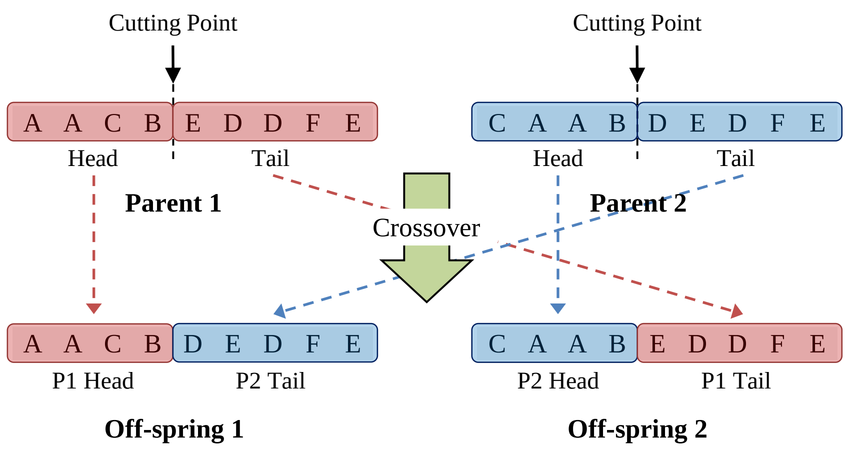


Figure 7-4. Illustration of the single point crossover procedure

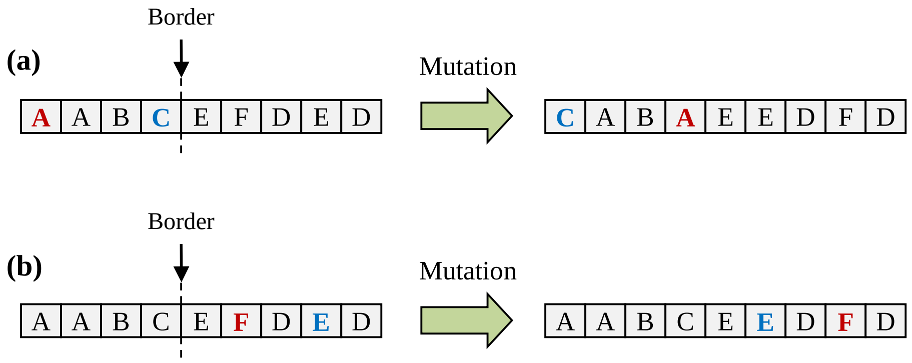


Figure 7-5. Mutation with random two genes exchange; (a) before the border, and (b) after the border

## 7.5. Numerical Example

A numerical example is given in this section to illustrate the model sequencing procedure of the developed approach in detail.

### 7.5.1. Problem data

Problem data is taken from Section 6.3 but a different solution approach is applied on it and a different result is obtained. To remember, two different precedence relationship diagrams, P12 (Kim *et al.*, 2000c) and P16 (Lee *et al.*, 2001), have been taken into account for two parallel two-sided assembly lines (P12 for Line I and P16 for Line II). Each of these diagrams were assumed common among three different product models of a product on each line; *i.e.* product models A, B and C on Line I; and product models D, E and F on Line II. For a fixed planning horizon of 480 time units, demands were assumed $D_{1A} = 8$, $D_{1B} = 8$, $D_{1C} = 16$, $D_{2D} = 8$, $D_{2E} = 8$ and $D_{2F} = 8$. The initial computations will not be repeated here in details as they have already been provided in Section 6.3.

As explained in the previous chapter, cycle times could be computed as $C_1 = 15$ and $C_2 = 20$ time units, for Line I and Line II, respectively. The LCM based approach, proposed by Gökçen *et al.* (2006), is used due to cycle time differences between the lines. $LCM(C_1, C_2) = 60$ is considered as the common cycle time $(C = 60)$ for both lines and line divisors of the lines $(ld_h)$ are obtained as $ld_1 = LCM(C_1, C_2)/C_1 = 60/15 = 4$ and $ld_2 = LCM(C_1, C_2)/C_2 = 60/20 = 3$. Task times of product models on Line I and Line II are normalised by being multiplied with $ld_1$ and $ld_2$, respectively. Table 7-1 provides the problem data with normalised task times that will be used while balancing the lines.

Table 7-1. Problem data with normalised task times

| Task no | Line I (P12) | | | | | Line II (P16) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Processing time | | | Side | Immediate predecessor(s) | Processing time | | | Side | Immediate predecessor(s) |
| | A | B | C | | | D | E | F | | |
| 1 | 16 | 8 | 24 | L | - | 15 | 21 | 12 | E | - |
| 2 | 32 | 40 | 28 | R | - | 0 | 12 | 0 | E | - |
| 3 | 12 | 20 | 12 | E | - | 15 | 30 | 21 | L | 1 |
| 4 | 0 | 8 | 16 | L | 1 | 12 | 24 | 6 | E | 1 |
| 5 | 12 | 4 | 8 | E | 2 | 9 | 12 | 24 | R | 2 |
| 6 | 4 | 24 | 0 | L | 3 | 3 | 6 | 9 | L | 3 |
| 7 | 8 | 0 | 8 | E | 4, 5 | 21 | 3 | 18 | E | 4, 5 |
| 8 | 20 | 24 | 24 | R | 5 | 12 | 12 | 15 | E | 6, 7 |
| 9 | 16 | 16 | 8 | E | 5, 6 | 6 | 6 | 3 | R | 7 |
| 10 | 8 | 20 | 0 | E | 7, 8 | 9 | 9 | 12 | R | 7 |
| 11 | 8 | 36 | 20 | E | 9 | 15 | 21 | 12 | E | 8 |
| 12 | 12 | 8 | 4 | R | 11 | 3 | 18 | 15 | L | 9 |
| 13 | - | - | - | - | - | 12 | 12 | 18 | E | 9, 10 |
| 14 | - | - | - | - | - | 15 | 6 | 9 | E | 11 |
| 15 | - | - | - | - | - | 0 | 12 | 3 | E | 11, 12 |
| 16 | - | - | - | - | - | 15 | 9 | 15 | E | 13 |

Based on the product model demands, minimum part sets are $MPS_1 = (1, 1, 2)$ and $MPS_2 = (1, 1, 1)$ for Line I and Line II, respectively. According to the minimum part sets on the lines, the number of possible product model sequences for Line I and Line II are $TS_1 = 4!/(1! \times 1! \times 2!) = 12$ and $TS_2 = 3!/(1! \times 1! \times 1!) = 6$. This means that $12 \times 6 = 72$ different combinations of product model sequences must be tried if combinatorial sequencing is selected by the user. Finally, the number of different production cycles subject to consideration for each product model sequence is 12.

### 7.5.2. Simulation of the solution procedure and its interpretation

The algorithm is run using the parameters given in Table 7-2. Initial population and evolution of the chromosomes for the first and the last generations of the population are briefly simulated in Table 7-3 and Table 7-4, respectively.

Table 7-2. Parameters used to solve the example problem

| | $\alpha$ | $\beta$ | $\rho$ | $Q$ | Initial pheromone | Colony size | Number of colonies |
|---|---|---|---|---|---|---|---|
| **ACO** | 0.1 | 0.2 | 0.1 | 50 | 10 | 10 | 5 |

| | Population size | Crossover rate | Mutation rate | Number of iterations |
|---|---|---|---|---|
| **GA** | 10 | 0.4 | 0.1 | 10 |

Table 7-3. Initial population

| Chromosome | Obtained best line balance for the chromosome determined | Fitness |
|---|---|---|
| C A B C E F D | [[1, 6, 4], [3, 2], [2, 1], [5, 4, 7], [5, 9, 10], [8, 7, 12], [3, 6, 8], [9, 10, 13, 16], [], [11, 12], [11, 14, 15], []] | 16 |
| C A B C F E D | [[1, 6, 4], [3, 2], [1, 4, 7], [2, 5, 9], [5, 11], [9, 7, 8], [3, 6, 12], [10, 13, 16, 8], [], [10, 12], [11, 15, 14], []] | 16 |
| C C A B D F E | [[3, 1, 6, 4, 7], [2, 5], [2, 3, 6], [1, 4, 5, 7], [], [9, 11, 12], [8, 11, 13], [10, 14, 9], [], [8, 10], [16, 12, 15], []] | 15 |
| C C A B F D E | [[1, 4, 3, 6], [2, 5, 7], [1, 4, 7], [2, 5, 10], [9, 11], [8, 10, 12], [3, 6, 8, 16], [9, 13], [], [], [11, 14, 12, 15], []] | 15 |
| C A C B F D E | [[6, 1, 4], [3, 2], [2, 3, 6], [1, 5, 4, 7], [9, 11], [5, 8, 7, 10, 15], [8, 12, 16], [10, 9, 11, 13, 14], [], [12], [], []] | 15 |
| C B C A D E F | [[1, 4, 3, 6, 7], [2, 5], [1, 3, 6], [2, 5, 4, 7, 10], [11], [9, 8, 10], [8, 12, 14, 12], [9, 11, 15, 13], [], [], [16], []] | 15 |
| C A B C F D E | [[3, 1, 6, 4, 7], [2, 5], [4], [2, 1, 5, 7], [9, 11], [8, 10, 12], [3, 12, 6], [9, 10, 13, 16], [], [], [8, 11, 15, 14], []] | 15 |
| C B A C D E F | [[3, 6, 1, 4, 7], [2, 5, 9], [1, 3, 6, 7], [2, 4, 5], [11], [8, 10, 12], [8, 11, 14, 12], [10, 9, 13, 16], [], [], [15], []] | 15 |
| C A B C E F D | [[3, 1, 6, 4], [2, 5, 7], [1, 3, 6], [2, 4, 5, 7], [9, 10], [8, 11], [8, 12, 15, 16], [9, 10, 11, 13, 14], [], [12], [], []] | 15 |
| C C A B E D F | [[3, 6, 1, 4, 7], [2, 5, 9], [4], [1, 2, 5, 7, 9], [11, 10], [8, 12], [12, 3, 6], [10, 13, 16], [], [], [8, 15], [11, 14]] | 16 |

The importance of line length in computing the fitness value is considered as double of that of the total number of utilised workstations ($\gamma_1 = 2, \gamma_2 = 1$). As could be seen from Table 7-3, the line balancing solutions are given only for the chromosomes in the initial population. However, the fitness values of the chromosomes in the first and the last populations and the ones emerging from the crossover and mutation procedures in the first and the last iterations are given in Table 7-4.

Table 7-4. Evolution of the chromosomes through generations

| New chromosomes after crossover and mutation | Fitness | Generation 1 after replacement of the worst chromosomes in the population | Fitness |
|---|---|---|---|
| C B A C F E D | 15 | C C A B D E F | 12* |
| C A B C D E F | 15 | C B A C F E D | 15 |
| C C A B D E F | 12 | C C A B D F E | 15 |
| C B C A F D E | 16 | C C A B F D E | 15 |
| C A C B E F D | 15 | C A B C F D E | 15 |
| C A B C F D E | 15 | C B C A D E F | 15 |
| C C A B F D E | 16 | C A B C F D E | 15 |
| C A B C D F E | 15 | C B A C D E F | 15 |
| - | - | C A B C E F D | 15 |
| - | - | C A B C D E F | 15 |
| … | … | … | … |
| New chromosomes after crossover and mutation | Fitness | Generation 10 after replacement of the worst chromosomes in the population | Fitness |
| C C A B F E D | 16 | C C A B D E F | 12* |
| C B A C D E F | 15 | C B A C F E D | 15 |
| C A B C F E D | 16 | C C A B D E F | 15 |
| C B A C F E D | 15 | C C A B D E F | 15 |
| C A B C E F D | 15 | C A B C F D E | 15 |
| A C B C F E D | 15 | A B C C D E F | 15 |
| C C A B E D F | 16 | C A B C F E D | 15 |
| - | - | C B A C F E D | 15 |
| - | - | A C B C E F D | 15 |
| - | - | C A B C D E F | 15 |

**Best chromosome: C C A B D E F (product model sequences - Line I: CCAB, Line II: DEF)*
*Best line balancing solution:*
*[[3, 1, 4, 6], [2, 5, 7], [1, 3, 6], [2, 5, 4, 7, 9], [9, 11], [8, 10, 12], [12, 11, 15, 14], [8, 10, 13, 16]]*
*Best fitness: 12 (LL=2, NS=8)*

Each ant starts assigning tasks from a randomly selected line and side (see Figure 7-6). For this example, ant starts by assigning tasks belonging to product models produced on Line I (A, B and C) from left side and selects task 3 to assign from the list of tasks available for this position. Then, tasks 1, 4, and 6 are assigned one by one to this side before ant changes the side.

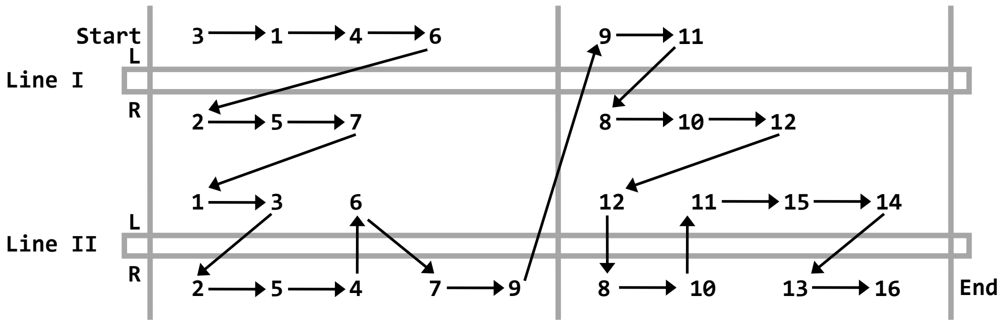
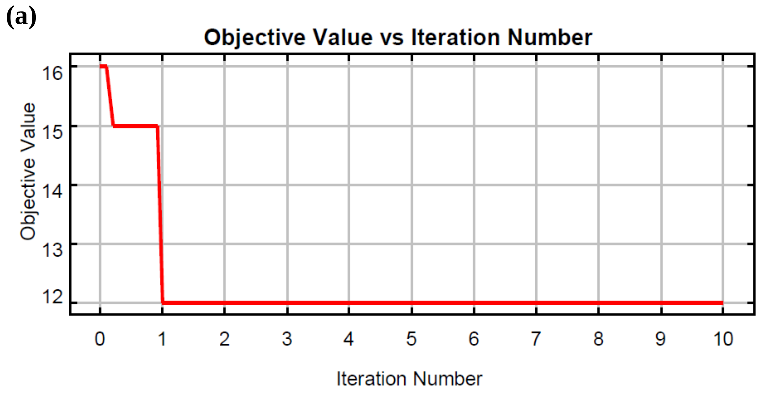
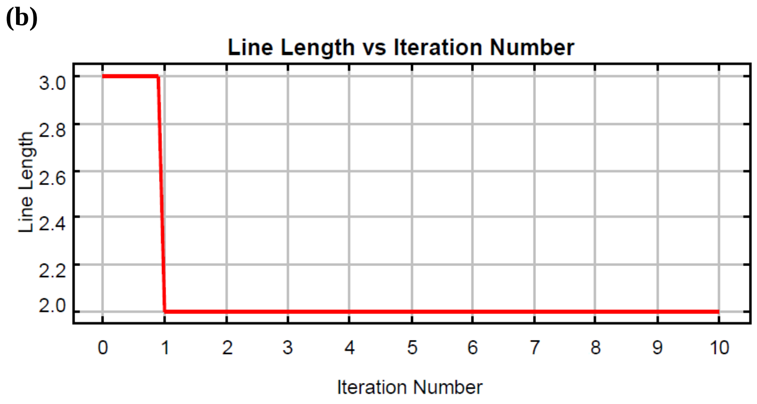


Figure 7-6. The task assignment configuration of the obtained best solution





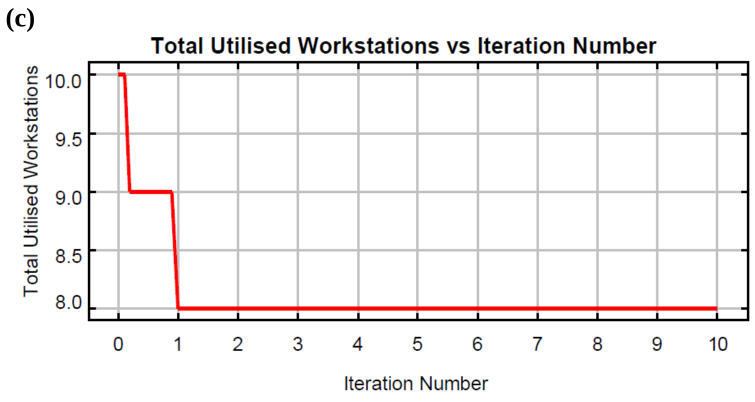


Figure 7-7. Convergence of the algorithm for the example problem by means of (a) objective value (fitness), (b) line length, and (c) total number of utilised workstations

Tasks 2, 5, and 7 are assigned to right side of the Line I and ant moves forward to left side of Line II to assign tasks belonging to produced product models on this line (D, E and F). Available tasks list is updated every time when a new task is assigned and if the capacity is full or there is no available task to assign for the current line, line is changed and tasks are assigned to the new workstations on the new line. This process continues until all tasks are assigned.

The convergence of the algorithm is exhibited in Figure 7-7 by means of objective value, line length and the total number of required workstations. The algorithm is run for 10 iterations but the change in the performance measures while generating the initial population is also recorded and represented between iterations #0 and #1 in the figure. The algorithm calculates fitness values of the chromosomes in the initial population while generating them in iteration #0 and the best fitness value decreases from 16 to 15 at this stage. Then it converges quickly and finds the best solution after crossover and mutation operations in the first iteration. One offspring (*CCABDEF*) is designated as the best individual at this stage and this situation remains the same until the last iteration. Then the algorithm reaches the maximum number of iterations at iteration #10 and is terminated with the best fitness value of 12. These graphs exhibit the effect of model sequencing procedure on the quality of the obtained line balance, once again.

## 7.6. Computational Experiments

To assess the effect of the proposed GA-based model sequencing approach on the quality of the found solution, test bed problems are solved and compared with results obtained in the previous chapter. This section provides the test problem data first, followed by a comparison of the solutions obtained.

### 7.6.1. Test problem data

A total of 24 test cases were generated by combining well-known test problems in the literature for mixed-model parallel two-sided assembly line balancing problem and MPTALB/S problem in Section 5.5 and Section 6.4, respectively. In this section, the same test cases are considered and solved using the developed hybrid ABACO/S-GA algorithm to evaluate its performance and observe the effect of the generated GA-based model sequencing procedure on the performance of the developed approach. Considered precedence relationship diagrams (shown in the ‘Problem’ column), cycle times and product model demands for each line of the test cases are presented in Table 7-5.

Table 7-5. Data for test cases

| **Test case** | **Problem** | | **Cycle time** | | **Demands Line I** | | | **Demands Line II** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *Line I* | *Line II* | *Line I* | *Line II* | A | B | C | D | E | F |
| 1 | P9 | P9 | 4 | 7 | 40 | 20 | 10 | 20 | 10 | 10 |
| 2 | P9 | P9 | 6 | 5 | 20 | 20 | 10 | 15 | 30 | 15 |
| 3 | P9 | P12 | 5 | 8 | 40 | 20 | 20 | 20 | 20 | 10 |
| 4 | P9 | P12 | 7 | 6 | 15 | 15 | 30 | 20 | 10 | 40 |
| 5 | P12 | P12 | 4 | 5 | 20 | 10 | 20 | 10 | 20 | 10 |
| 6 | P12 | P12 | 6 | 5 | 20 | 10 | 20 | 30 | 15 | 15 |
| 7 | P12 | P16 | 9 | 12 | 10 | 20 | 10 | 10 | 10 | 10 |
| 8 | P12 | P16 | 10 | 12 | 20 | 20 | 20 | 10 | 20 | 20 |
| 9 | P16 | P16 | 12 | 15 | 10 | 20 | 20 | 20 | 10 | 10 |
| 10 | P16 | P16 | 16 | 14 | 10 | 40 | 20 | 40 | 20 | 20 |
| 11 | P16 | P24 | 14 | 16 | 40 | 20 | 20 | 40 | 20 | 10 |
| 12 | P16 | P24 | 16 | 18 | 15 | 45 | 30 | 20 | 40 | 20 |
| 13 | P24 | P24 | 15 | 20 | 20 | 10 | 10 | 10 | 10 | 10 |
| 14 | P24 | P24 | 25 | 20 | 10 | 20 | 10 | 10 | 20 | 20 |
| 15 | A65 | A65 | 300 | 480 | 40 | 20 | 20 | 20 | 10 | 20 |
| 16 | A65 | A65 | 420 | 360 | 15 | 15 | 30 | 20 | 40 | 10 |
| 17 | A65 | B148 | 405 | 810 | 10 | 5 | 5 | 4 | 4 | 2 |
| 18 | A65 | B148 | 675 | 540 | 20 | 10 | 10 | 10 | 20 | 20 |
| 19 | B148 | B148 | 255 | 510 | 5 | 10 | 5 | 2 | 4 | 4 |
| 20 | B148 | B148 | 425 | 340 | 20 | 10 | 10 | 20 | 20 | 10 |
| 21 | B148 | A205 | 510 | 1020 | 10 | 5 | 15 | 3 | 6 | 6 |
| 22 | B148 | A205 | 600 | 1200 | 6 | 3 | 3 | 2 | 2 | 2 |
| 23 | A205 | A205 | 1200 | 1200 | 10 | 10 | 10 | 10 | 10 | 10 |
| 24 | A205 | A205 | 1000 | 2000 | 15 | 5 | 10 | 5 | 5 | 5 |

The algorithm is coded in Java$^{TM}$ Standard Edition 7u4 environment and all test cases are solved using a PC with 3.1 GHz Intel Core$^{TM}$ i5-2400 CPU and 4GB of RAM. The parameters of the algorithm are chosen experimentally in accordance with the scale of the test cases for a high-quality solution and are given in Table 7-6.

The table provides the values of parameters alpha ($\alpha$), beta ($\beta$), pheromone evaporation rate ($\rho$), pheromone deposition parameter ($Q$), initial pheromone, colony size, and the number of iterations for the ACO algorithm; and population size, crossover rate, and mutation rate for the GA algorithm. The values of these parameters may differ from one test instance to another to search the exponentially growing search space (with the increasing number of tasks and complexity) more effectively.

Table 7-6. Parameters of the hybrid ABACO/S-GA approach

| | Test cases | $\alpha$ | $\beta$ | $\rho$ | $Q$ | Initial pheromone | Colony size | Number of colonies |
|---|---|---|---|---|---|---|---|---|
| **ACO** | 1-6 | 0.1 | 0.2 | 0.1 | 50 | 10 | 10 | 5 |
| | 7-14 | 0.1 | 0.2 | 0.1 | 50 | 15 | 20 | 10 |
| | 15-20 | 0.1 | 0.2 | 0.1 | 50 | 20 | 30 | 15 |
| | 21-24 | 0.1 | 0.2 | 0.1 | 50 | 25 | 40 | 20 |

| | Test cases | Population size | Crossover rate | Mutation rate |
|---|---|---|---|---|
| **GA** | 1-6 | 8 | 0.2 | 0.1 |
| | 7-14 | 10 | 0.4 | 0.2 |
| | 15-20 | 16 | 0.6 | 0.3 |
| | 21-24 | 20 | 0.6 | 0.35 |

### 7.6.2. Results and comparison

The main objective of the computational tests is to investigate in how many iterations the proposed approach finds the same solution with the ABACO/S results presented in Chapter 6 for the same cases. For this aim, the algorithm is run for designated parameters and available problem data for each test case and terminated when the same fitness value with ABACO/S is found. Obtained results are reported in the ABACO/S-GA column of Table 7-7 where ABACO/S column gives the results from Chapter 6 for the purpose of comparison. The abbreviations *LL*, *NS*, and *OBJ* in Table 7-7 correspond to line length, number of workstations, and objective value, respectively (where the user defined parameters in the objective function are considered as $\gamma_1 = 2$, and $\gamma_2 = 1$). *NI* columns exhibit the number of iterations that ABACO/S and ABACO/S-GA algorithms are run. Based on the coding structure of ABACO/S, the number of sequences that the algorithm tried to find the best solution is considered as the number of iterations and is shown in the table (see column *NI*). For the ABACO/S-GA, *NI* column shows the number of iterations that the GA algorithm is run. In this column, '$IP + X$' represents the best solution is found in the $X^{th}$ iteration after generating initial population. If $X =$ 0, then it means that the best solution is found while generating the random chromosomes for the initial population. This issue (obtaining the best solution while generating the initial population) is observed only for the small-sized test cases.

As could be seen from the table, it is clear that the same objective values are obtained with the same $LL$ and $NS$ values except test case #11. In this test case, the objective value of 34 is obtained with $LL = 7$, and $NS = 20$, different from the solution obtained by ABACO/S. Also, the same objective values are obtained with different product model sequences except test cases #7 and #13, for which the product model sequences are found as *ACBB-FDE* and *ABCA-EFD*, respectively.

Table 7-7. Computational results

| # | **ABACO/S** | | | | | **ABACO/S-GA** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *Best sequence Line I - Line II* | *NI* | *LL* | *NS* | *OBJ* | *Best sequence Line I - Line II* | *NI* |
| 1 | 3 | 10 | 16 | AABAABC-FDED | 15 | 3 | 10 | 16 | BAAACBA-DEFD | IP + 0 |
| 2 | 3 | 9 | 15 | CAABB-EDFE | 15 | 3 | 9 | 15 | AABCB-EDEF | IP + 0 |
| 3 | 3 | 8 | 14 | AABC-DEFDE | 15 | 3 | 8 | 14 | BAAC-EDEDF | IP + 0 |
| 4 | 2 | 8 | 12 | BCAC-DFFFFED | 15 | 2 | 8 | 12 | BACC-FEDFDFF | IP + 1 |
| 5 | 4 | 12 | 20 | CCAAB-EDEF | 15 | 4 | 12 | 20 | BCCAA-EFED | IP + 1 |
| 6 | 3 | 10 | 16 | ABACC-DFDE | 15 | 3 | 10 | 16 | CBCAA-EDFD | IP + 1 |
| 7 | 7 | 17 | 31 | ACBB-FDE | 20 | 7 | 17 | 31 | ACBB-FDE | IP + 2 |
| 8 | 7 | 17 | 31 | BCA-DEFFE | 20 | 7 | 17 | 31 | ABC-EFDFE | IP + 2 |
| 9 | 7 | 23 | 37 | CABCB-EFDD | 20 | 7 | 23 | 37 | BCCBA-EDDF | IP + 2 |
| 10 | 7 | 20 | 34 | CCBBABB-DEFD | 20 | 7 | 20 | 34 | CCBBABB-DEDF | IP + 1 |
| 11 | 6 | 22 | 34 | ABAC-EFDEDDD | 20 | 7 | 20 | 34 | CAAB-EDFDDDE | IP + 3 |
| 12 | 5 | 17 | 27 | BBCCBA-FEDE | 20 | 5 | 17 | 27 | CBCBBA-DEFE | IP + 2 |
| 13 | 5 | 19 | 29 | ABCA-EFD | 20 | 5 | 19 | 29 | ABCA-EFD | IP + 4 |
| 14 | 4 | 14 | 22 | CABB-EFEFD | 20 | 4 | 14 | 22 | BCBA-FEDEF | IP + 2 |
| 15 | 11 | 38 | 60 | BACA-EFFDD | 40 | 11 | 38 | 60 | ABCA-DDEFF | IP + 2 |
| 16 | 10 | 37 | 57 | CBCA-FDEEDEE | 40 | 10 | 37 | 57 | BCAC-DDFEEEE | IP + 2 |
| 17 | 9 | 32 | 50 | AABC-EEDFD | 40 | 9 | 32 | 50 | BACA-EDFDE | IP + 3 |
| 18 | 9 | 31 | 49 | ACBA-EFFED | 40 | 9 | 31 | 49 | BAAC-FFEDE | IP + 3 |
| 19 | 18 | 65 | 101 | ABCB-EEFDF | 40 | 18 | 65 | 101 | ABBC-FEEFD | IP + 4 |
| 20 | 15 | 58 | 88 | BACA-DEDEF | 40 | 15 | 58 | 88 | ABAC-EDEDF | IP + 3 |
| 21 | 17 | 61 | 95 | BCAACC-EFEFD | 50 | 17 | 61 | 95 | ABCACC-FEDEF | IP + 3 |
| 22 | 15 | 53 | 83 | AACB-FDE | 50 | 15 | 53 | 83 | ACAB-FDE | IP + 3 |
| 23 | 16 | 62 | 94 | ACB-FED | 50 | 16 | 62 | 94 | ABC-FDE | IP + 5 |
| 24 | 17 | 59 | 93 | CBCAAA-FED | 50 | 17 | 59 | 93 | CBACAA-EFD | IP + 4 |

According to the computational results, the proposed algorithm finds the same fitness values with the ABACO/S in less number of iterations. For instance, if we consider test case #2, ABACO/S-GA finds the objective value 15 in '$IP + 0$' iteration. In this case, 8 chromosomes are generated for the initial population and the same fitness value is found with the ABACO/S at this stage. For test case #10, the proposed approach found the objective value of 34 in '$IP + 1$' iteration, which means that 10 chromosomes are generated for the initial population and the best solution is discovered in the first iteration of crossover and mutation operations.

One could argue that the total number of evaluated chromosomes depends on the number of offspring created after crossover and mutation operations, but even the number of total chromosome evaluations is most likely less than the number of iterations considered by ABACO/S. Also, the robustness of the algorithm is another advantage gained.

In terms of the performance of the proposed algorithm, it is clear that the algorithm needs higher numbers of iterations for large-sized test cases compared with the small-sized test cases. However, this is naturally caused by the exponentially growing search space with the increasing number of tasks.

In addition to the above tests, ABACO/S-GA algorithm is run again for the same test cases to observe whether the algorithm can find better solutions than the ABACO/S, without a target objective value. The algorithm is run for 15, 20, 30 and 40 iterations for test cases #1–#6, #7–#14, #15–#20, and #21-#24, respectively; and obtained better results are reported in Table 7-8. As could be seen from the table, better solutions than ABACO/S are found for some of the test cases (note that the same solutions have already been reported in Table 7-7 for the remaining problems). The objective function values obtained using the proposed algorithm for three test cases (#16, #19, and #20) are one lower than the solutions obtained by ABACO/S. Consequently, there is no doubt that the solutions found by ABACO/S-GA are better than those obtained by ABACO/S for the same test cases.

Table 7-8. Better solutions obtained by ABACO/S-GA

| # | **ABACO/S** | | | | **ABACO/S-GA** | | | |
|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *Best sequence Line I - Line II* | *LL* | *NS* | *OBJ* | *Best sequence Line I - Line II* |
| 16 | 10 | 37 | 57 | CBCA-FDEEDEE | 10 | 36 | 56 | CABC-EFDDEEE |
| 19 | 18 | 65 | 101 | ABCB-EEFDF | 18 | 64 | 100 | BCBA-EFDFE |
| 20 | 15 | 58 | 88 | BACA-DEDEF | 15 | 57 | 87 | BACA-DEEFD |

## 7.7. Chapter Summary

In this chapter, a GA-based model sequencing procedure was developed and integrated to the ABACO/S approach for solving the problem of simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines effectively. The main aim was to improve the competence of the model sequencing procedure of the entire algorithm for obtaining good quality solutions by requiring less amount of computational effort than the ABACO/S for the same problem.

This chapter contributes to knowledge in terms of its novel methodology which incorporates GA and agent-based ACO algorithm to solve one of the most complex and rarely studied manufacturing problems. Moreover, to the best knowledge of the author, this is the first attempt to hybridise GA and ACO algorithms to balance parallel two-sided assembly lines in the literature.

An example was given to illustrate the running mechanism of the proposed technique and evolution of the chromosomes through generations. To assess the performance of the developed technique, test cases were solved and the iteration numbers that the best solutions found were compared with the results found in the previous chapter. When the algorithm was run without a target objective value for the same test cases, better results were obtained for three test cases out of 24. The results indicated that the solution capacity of ABACO/S has increased with the improvements done in terms of solving the provided test cases.

# 8

# ANALYSIS AND DISCUSSION

**Contents**

## 8.1. Chapter Introduction

This chapter examines the results obtained using the developed algorithms within the scope of this thesis. Some brief background information on the used statistical test technique is provided and the data used for the statistical analysis is summarised in Section 8.2. The tests are conducted to measure the effect of balancing lines together (see Section 8.3) and considering the model sequencing problem together with the line balancing problem (see Section 8.4), on the objective function (which is a weighted summation of line length and the total number of workstations) values of the obtained solutions. The statistical analysis of the performances of proposed ABACO, ABACO/S and ABACO/S-GA algorithms are provided in Section 8.5 and a summary of the chapter is given in Section 8.6. Please note that the histograms, individual value plots and boxplots of differences for the statistical tests performed in the following sub-sections are also reported in Appendices A.2.

## 8.2. Methodology and Test Data

A set of paired sample t-tests (also known as paired two-sample t-test) is conducted in Minitab® 17.1.0 statistics software for the comprehensive statistical analyses of the results obtained in the previous chapters. The reason of selecting this particular test is that the paired sample t-test is accepted as a powerful and sensitive way to detect differences and used to analyse the effect of an application on the same individuals. Usually, the individuals are measured before and after some type of treatment and the results are analysed. Please refer to Stone and Ellis (2006) for more information on this test.

The tests presented in the following sub-sections are one-tailed as we are looking for whether a proposed system or technique, such as balancing the lines together or considering the model sequencing problem as well as the line balancing problem, makes any improvement on the objective function sought. This can also be verified from the hypotheses that will be tested in the following sections and the test data that will be used for the purpose of statistical analysis. Thus, we are only interested in one side of the probability distribution, which is shown in Figure 8-1. In this distribution, the shaded region shows the area represented by the null hypothesis, $H_0$: $\mu = \mu_0$, which actually implies $\mu \leq \mu_0$, since the only unshaded region in the figure shows $\mu > \mu_0$.

One tailed tests are classified as left tailed and right tailed tests in accordance with the region unshaded. A left tailed test has an alternative hypothesis of $H_1$ which uses a less than '$<$' condition resulting in a critical value on left tail while a right tailed test uses a greater than '$>$' operator in the alternative hypothesis resulting in a critical value on right tail.

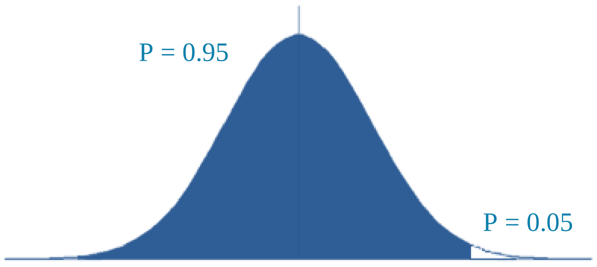


Figure 8-1. Probability distribution (Stone and Ellis, 2006)

An extreme value on only one side of the sampling distribution would cause us to reject the null hypothesis. The decision to reject the null hypothesis ($H_0$) or fail to reject it can be based on the $p$-value and chosen significance level ($a = 0.05$). In accordance with this rule, $H_0$ is rejected if the $p$-value is less than or equal to the considered significance level ($p \leq 0.05$). This can be used regardless of whether the test is left tailed or right tailed (Stone and Ellis, 2006).

The decision can also be based on the confidence interval (or bound) calculated using the same $a$ value. For this aim, test statistic ($t\_stat$) values will also be presented in the results tables of the statistical tests. The $t\_critical$ values will be considered negative if the test is one tailed and positive if the test is right tailed. Thus, for a left tailed test, the null hypothesis is rejected if the test statistic ($t\_stat$) is less than critical value ($t\_stat < t\_critical$); otherwise $H_0$ is retained. For a right tailed test, $H_0$ is rejected if $t\_stat$ is greater than critical value, $t\_stat > t\_critical$ (Minitab, 2015). Table 8-1 gives a summary of decision rules.

Table 8-1. The summary of decision rules used for statistical tests

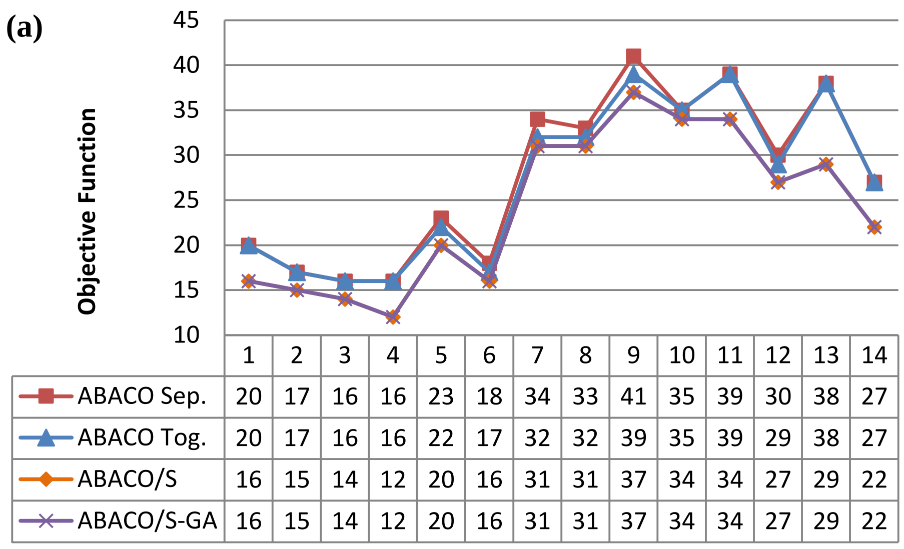
| Test type | The condition for rejecting $H_0$ | |
|---|---|---|
| | Based on $p$ -value | Based on $t\_stat$ value |
| Left tailed | $p \leq 0.05$ | $t\_stat < t\_critical$ |
| Right tailed | $p \leq 0.05$ | $t\_stat > t\_critical$ |

As mentioned earlier, the statistical tests are conducted through the objective function values of the obtained solutions, where the objective function value of a particular solution is composed of the weighted summation of line length and the number of workstations required by this solution. The objective function values of the balancing solutions obtained for test cases under different conditions using the algorithms proposed in the previous chapters are summarised in Table 8-2. Also, Figure 8-2 depicts these values to provide a clear understanding. The ABACO Separate and ABACO Together columns represent the results investigated by ABACO algorithm (in Chapter 5) when the lines are balanced independently and when the lines are balanced together, respectively. Apparently, ABACO/S and ABACO/S-GA columns present the results obtained by ABACO/S and ABACO/S-GA algorithms in Chapter 6 and Chapter 7, respectively.

Table 8-2. Computed objective function values for the instances solved with ABACO Separate, ABACO Together, ABACO/S, and ABACO/S-GA

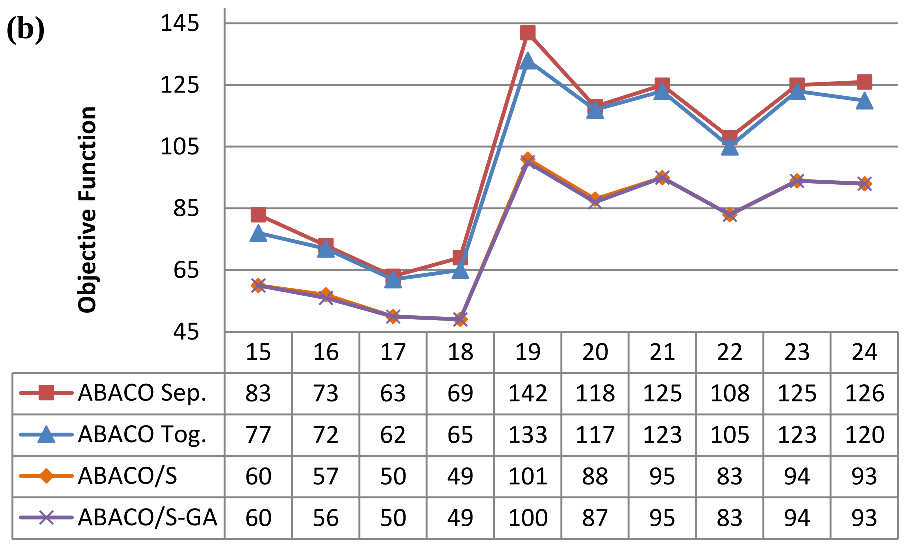
| Test case | ABACO Separate | ABACO Together | ABACO/S | ABACO/S GA |
|---|---|---|---|---|
| 1 | 20 | 20 | 16 | 16 |
| 2 | 17 | 17 | 15 | 15 |
| 3 | 16 | 16 | 14 | 14 |
| 4 | 16 | 16 | 12 | 12 |
| 5 | 23 | 22 | 20 | 20 |
| 6 | 18 | 17 | 16 | 16 |
| 7 | 34 | 32 | 31 | 31 |
| 8 | 33 | 32 | 31 | 31 |
| 9 | 41 | 39 | 37 | 37 |
| 10 | 35 | 35 | 34 | 34 |
| 11 | 39 | 39 | 34 | 34 |
| 12 | 30 | 29 | 27 | 27 |
| 13 | 38 | 38 | 29 | 29 |
| 14 | 27 | 27 | 22 | 22 |
| 15 | 83 | 77 | 60 | 60 |
| 16 | 73 | 72 | 57 | 56 |
| 17 | 63 | 62 | 50 | 50 |
| 18 | 69 | 65 | 49 | 49 |
| 19 | 142 | 133 | 101 | 100 |
| 20 | 118 | 117 | 88 | 87 |
| 21 | 125 | 123 | 95 | 95 |
| 22 | 108 | 105 | 83 | 83 |
| 23 | 125 | 123 | 94 | 94 |
| 24 | 126 | 120 | 93 | 93 |

**(a)**



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ■ ABACO Sep. | 20 | 17 | 16 | 16 | 23 | 18 | 34 | 33 | 41 | 35 | 39 | 30 | 38 | 27 |
| ▲ ABACO Tog. | 20 | 17 | 16 | 16 | 22 | 17 | 32 | 32 | 39 | 35 | 39 | 29 | 38 | 27 |
| ◆ ABACO/S | 16 | 15 | 14 | 12 | 20 | 16 | 31 | 31 | 37 | 34 | 34 | 27 | 29 | 22 |
| × ABACO/S-GA | 16 | 15 | 14 | 12 | 20 | 16 | 31 | 31 | 37 | 34 | 34 | 27 | 29 | 22 |

**(b)**



| | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|
| ■ ABACO Sep. | 83 | 73 | 63 | 69 | 142 | 118 | 125 | 108 | 125 | 126 |
| ▲ ABACO Tog. | 77 | 72 | 62 | 65 | 133 | 117 | 123 | 105 | 123 | 120 |
| ◆ ABACO/S | 60 | 57 | 50 | 49 | 101 | 88 | 95 | 83 | 94 | 93 |
| × ABACO/S-GA | 60 | 56 | 50 | 49 | 100 | 87 | 95 | 83 | 94 | 93 |

Figure 8-2. Comparison of the objective function values; (a) test cases #1–#14, and (b) test cases #15–#24

## 8.3. Balancing Lines Together

A paired sample t-test is conducted to compare the objective function values in separate balancing and together balancing conditions. The test is performed for the ABACO Together and ABACO Separate results where multi-line stations are allowed and multi-line stations are not allowed, respectively. Model sequencing problem is subject to consideration in neither of the cases. The null and alternative hypotheses stated at the $a = 0.05$ significance level (95% confidence interval) for means of objective function values obtained from ABACO Together ($\mu_T$) and ABACO Separate ($\mu_S$) are as follows:

$H_0$: *There is no significant difference between the means of objective function values obtained when the lines are balanced together and separately in favour of the alternative* ($\mu_T \geq \mu_S$).

$\mathrm{H}_1$: *Balancing lines together significantly reduces the obtained objective function values in comparison to balancing lines separately* ($\mu_T < \mu_S$).

As seen from $H_1$, the test is left tailed due to the smaller than condition '<' used. Therefore, the calculated $p$-value should be smaller than or equal to the significance level ($p \leq a$) or the calculated $t\_stat$ should be smaller than the $t\_critical$ one-tail value ($t\_stat < t\_crititical$) to reject the null hypothesis (as given in Table 8-1).

Table 8-3. The results of paired sample t-test to analyse the effect of balancing lines together on the objective function values of the obtained solutions

| | ***ABACO Together*** | ***ABACO Separate*** |
|---|---:|---:|
| Mean ($\mu$) | 57.33 | 59.13 |
| Standard deviation (SD) | 41.00 | 42.59 |
| Standard error mean | 8.37 | 8.69 |
| Variance | 1680.7 | 1814.2 |
| Observations | 24 | 24 |
| Pearson correlation | 0.999 | - |
| Hypothesised mean difference | 0 | - |
| Degrees of freedom | 23 | - |
| $\boldsymbol{t\_stat}$ | **-3.781** | - |
| $\boldsymbol{p(T \leq t)}$ **one-tail** | **0.000** | - |
| $\boldsymbol{t\_critical}$ **one-tail** | **-1.714** | - |
| $p(T \leq t)$ two-tail | 0.000 | - |
| $t\_critical$ two-tail | -2.069 | - |

* The $p$-value is considered zero as it is smaller than 0.0001 ($p < 0.0001$).

The results of the statistical test are reported in Table 8-3. As the calculated probability is much less than the considered significance level ($p = 0.000 \ll a = 0.05$), the null hypothesis of $H_0$ is rejected with a very strong evidence. The calculated $t\_stat$ value also verifies this result, $t\_stat = -3.781 \ll t\_crititical = -1.714$, based on the set of decision rules for paired sample t-tests summarised in Table 8-1. The statistical test results indicate that there is a significant difference in the means of objective function values between balancing the lines together ($\mu_T = 57.33, SD_T = 41.00$) and balancing the lines separately ($\mu_S = 59.13, SD_S = 42.59$) for the solved mixed-model parallel two-sided assembly line instances; $t(23) = -3.781, p = 0.000$. These results suggest that balancing mixed-model parallel two-sided assembly lines together has a significant effect on the means of objective function values of the solutions obtained at $a = 0.05$

significance level. Thus, it is statistically shown that the objective function values of obtained solutions are minimised when the mixed-model parallel two-sided lines are balanced together in comparison to balancing those lines separately.

## 8.4. Considering the Model Sequencing and Line Balancing Problems Together

To compare the objective function values obtained when the model sequencing problem is considered and not considered along with the line balancing problem, a paired sample t-test is carried out using the balancing solution results observed from ABACO/S and ABACO Together algorithms, respectively. To recall, as different from what assumed for the experimental test results used in Section 8.3, multi-line stations are allowed in both cases. However, model sequencing problem is considered only in the ABACO/S solutions. The null and alternative hypotheses stated at the $a = 0.05$ significance level (95% confidence interval) for means of objective function values obtained from ABACO/S ($\mu_{OS}$) and ABACO Together ($\mu_T$) are as follows:

$H_0$: *There is no significant difference between the means of objective function values obtained when the model sequencing problem is considered and not considered together with the line balancing problem, in favour of the alternative* ($\mu_{OS} \geq \mu_T$).

$H_1$: *Considering the model sequencing problem together with the line balancing problem makes a significant reduction in the objective function values of the solutions obtained* ($\mu_{OS} < \mu_T$).

As seen from the hypotheses, the test is left tailed ('<' is used in the alternative hypothesis). Therefore, the same rules considered in Section 8.3 will be applied to decide on whether to reject or accept the null hypothesis. Table 8-4 presents the results of the conducted paired sample t-test in accordance with the hypotheses set. Based on the test results, there is a significant difference between the obtained objective function values when the model sequencing problem is considered ($\mu_{OS} = 46.17, \mu_{OS} = 30.27$) and not considered ($\mu_T = 57.33, SD_T = 41.00$) as well as the line balancing problem; $t(23) = -4.948, p = 0.000$. As $p = 0.000 \ll a = 0.05$ (or $t\_stat = -4.948 \ll t\_crititical = -1.714$), the alternative hypothesis is accepted with a very strong evidence. Specifically, these results emphasise that solving the model sequencing and the line balancing problems simultaneously in MPTALB/S problem helps minimise objective function, which is a weighted summation of line length and the number of workstations. This is one of the most practical findings of this thesis. As shown in Section 8.3, balancing the mixed-model parallel two-sided lines together, where multi-line stations are allowed, also helps minimise the objective function, which is another important outcome of this thesis.

Table 8-4. The results of paired sample t-test to analyse the effect of considering the model sequencing and line balancing problems together on the objective function values of the obtained solutions

| | ***ABACO/S*** | ***ABACO Together*** |
|---|---:|---:|
| Mean ($\mu$) | 46.17 | 57.33 |
| Standard deviation (SD) | 30.27 | 41.00 |
| Standard error mean | 6.18 | 8.37 |
| Variance | 916.3 | 1680.7 |
| Observations | 24 | 24 |
| Pearson correlation | 0.9971 | - |
| Hypothesised mean difference | 0 | - |
| Degrees of freedom | 23 | - |
| $\boldsymbol{t\_stat}$ | **-4.948** | - |
| $\boldsymbol{p(T <= t)}$ **one-tail** | **0.000** | - |
| $\boldsymbol{t\_critical}$ **one-tail** | **-1.714** | - |
| $p(T \leq t)$ two-tail | 0.000 | - |
| $t\_critical$ two-tail | -2.069 | - |

* The $p$-value is considered zero as it is smaller than 0.001 ($p < 0.0001$).

## 8.5. The Performances of the Proposed Algorithms

This section aims to statistically test the performances ABACO and ABACO/S against other heuristics, and the performance of ABACO/S-GA against its primitive version, namely ABACO/S, in terms of the objective function values of the solutions obtained.

### 8.5.1. Analysing the performance of ABACO

To analyse the performance of the developed ABACO algorithm against other heuristics, the best solution found by other heuristics for each test case (in Chapter 5) is taken and listed in *best other heuristic (BOH)* column in Table 8-5. The objective function values (given as *OBJ*) of the solutions obtained by ABACO are compared with the *OBJ* values of BOH. A paired sample t-test is carried out to analyse whether the *OBJ* values obtained by ABACO significantly differ from the best *OBJ* values obtained by other six heuristics. Please note that multi-line stations were allowed to be established in both solution strategies, therefore the cases solved by BOH and ABACO were identical. The null and alternative hypotheses stated at the $a = 0.05$ significance level (95% confidence interval) for means of objective function values obtained from ABACO Together ($\mu_T$) and BOH ($\mu_{BOH}$) are as follows:

$H_0$: *There is no significant difference between the means of objective function values obtained by ABACO and BOH in favour of the alternative* ($\mu_T \geq \mu_{BOH}$).

$H_1$: *The objective function values of solutions obtained by ABACO are significantly lower than those of solutions obtained by BOH* ($\mu_T < \mu_{BOH}$).

Table 8-5. The data used for statistical test to analyse the performance of ABACO

| **Test case** | **BOH** | | | **ABACO Together** | | | **Test case** | **BOH** | | | **ABACO Together** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* |
| 1 | 4 | 12 | *20* | 4 | 12 | *20* | 13 | 7 | 25 | *39* | 7 | 24 | *38* |
| 2 | 3 | 11 | *17* | 3 | 11 | *17* | 14 | 5 | 17 | *27* | 5 | 17 | *27* |
| 3 | 3 | 10 | *16* | 3 | 10 | *16* | 15 | 14 | 49 | *77* | 14 | 49 | *77* |
| 4 | 3 | 10 | *16* | 3 | 10 | *16* | 16 | 13 | 48 | *74* | 13 | 46 | *72* |
| 5 | 4 | 15 | *23* | 4 | 14 | *22* | 17 | 11 | 41 | *63* | 11 | 40 | *62* |
| 6 | 3 | 12 | *18* | 3 | 11 | *17* | 18 | 12 | 41 | *65* | 12 | 41 | *65* |
| 7 | 7 | 18 | *32* | 7 | 18 | *32* | 19 | 25 | 86 | *136* | 24 | 85 | *133* |
| 8 | 7 | 19 | *33* | 7 | 18 | *32* | 20 | 20 | 78 | *118* | 20 | 77 | *117* |
| 9 | 8 | 25 | *41* | 7 | 25 | *39* | 21 | 23 | 80 | *126* | 22 | 79 | *123* |
| 10 | 7 | 21 | *35* | 7 | 21 | *35* | 22 | 20 | 68 | *108* | 19 | 67 | *105* |
| 11 | 7 | 25 | *39* | 7 | 25 | *39* | 23 | 21 | 82 | *124* | 21 | 81 | *123* |
| 12 | 5 | 20 | *30* | 5 | 19 | *29* | 24 | 22 | 77 | *121* | 22 | 76 | *120* |

The results of the statistical test are reported in Table 8-6. The test is left tailed as what we are testing is whether ABACO finds lower objective function values than the best other heuristic. Based on the test results, it is clear that there is a significant difference in the obtained objective function values between the ABACO Together ($\mu_T = 57.33, SD_T = 41.00$) and BOH ($\mu_{BOH} = 58.25, SD_{BOH} = 41.64$) for the solved MPTALB/S problem instances; $t(23) = -4.412, p = 0.0001$. The null hypothesis is rejected with a very strong evidence as ($p = 0.0001 \ll a = 0.05$ or $t\_stat = -4.412 \ll t\_crititical = -1.714$). As there is a significant difference between the means of objective function values obtained from BOH and ABACO, it is statistically proven that the developed ABACO approach outperforms all other six heuristics for solving MPTALB/S problem in terms of sought performance measures.

Table 8-6. The results of paired sample t-test to analyse the performance of ABACO in terms of the objective function values of the obtained solutions

| | ***ABACO Together*** | ***BOH*** |
|---|---:|---:|
| Mean ($\mu$) | 57.33 | 58.25 |
| Standard deviation ($SD$) | 41.00 | 41.64 |
| Standard error mean | 8.37 | 8.50 |
| Variance | 1680.7 | 1734.2 |
| Observations | 24 | 24 |
| Pearson correlation | 0.9998 | - |
| Hypothesised mean difference | 0 | - |
| Degrees of freedom | 23 | - |
| $\boldsymbol{t\_stat}$ | **-4.412** | - |
| $\boldsymbol{p(T \leq t)}$ **one-tail** | **0.0001** | - |
| $\boldsymbol{t\_critical}$ **one-tail** | **-1.714** | - |
| $p(T \leq t)$ two-tail | 0.0002 | - |
| $t\_critical$ two-tail | -2.069 | - |

### 8.5.2. Analysing the performance of ABACO/S

Similar to the previous section, this section analyses the performance of the developed ABACO/S algorithm against other developed heuristics to see whether the obtained results found by ABACO/S significantly differ from the best ones found by other heuristics. For this aim, the best solution found by three heuristics for each test case is determined and reported in the *BOH/S* (which represents *best other heuristic with sequencing*) column in Table 8-7. Multi-line stations were allowed and model sequencing problem was also considered as well as the line balancing problem while obtaining balancing solutions in both cases. Therefore, the statistical test is conducted for the observations done under the same conditions.

A paired sample t-test is conducted to compare the objective function values (given as *OBJ*) of BOH/S solutions with the objective function values of ABACO/S. The null and alternative hypotheses stated at the $a = 0.05$ significance level (95% confidence interval) for means of objective function values obtained from ABACO/S ($\mu_{OS}$) and BOH/S ($\mu_{BOH/S}$) are as follows:

Table 8-7. The data used for statistical test to analyse the performance of ABACO/S

| **Test case** | **BOH/S** | | | **ABACO/S** | | | **Test case** | **BOH/S** | | | **ABACO/S** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* | | *LL* | *NS* | *OBJ* | *LL* | *NS* | *OBJ* |
| 1 | 3 | 10 | *16* | 3 | 10 | *16* | 13 | 6 | 20 | *32* | 5 | 19 | *29* |
| 2 | 3 | 9 | *15* | 3 | 9 | *15* | 14 | 4 | 14 | *22* | 4 | 14 | *22* |
| 3 | 3 | 8 | *14* | 3 | 8 | *14* | 15 | 11 | 39 | *61* | 11 | 38 | *60* |
| 4 | 2 | 8 | *12* | 2 | 8 | *12* | 16 | 10 | 38 | *58* | 10 | 37 | *57* |
| 5 | 4 | 13 | *21* | 4 | 12 | *20* | 17 | 9 | 31 | *49* | 9 | 32 | *50* |
| 6 | 3 | 10 | *16* | 3 | 10 | *16* | 18 | 9 | 31 | *49* | 9 | 31 | *49* |
| 7 | 7 | 17 | *31* | 7 | 17 | *31* | 19 | 21 | 68 | *110* | 18 | 65 | *101* |
| 8 | 7 | 17 | *31* | 7 | 17 | *31* | 20 | 16 | 59 | *91* | 15 | 58 | *88* |
| 9 | 7 | 23 | *37* | 7 | 23 | *37* | 21 | 18 | 61 | *97* | 17 | 61 | *95* |
| 10 | 7 | 20 | *34* | 7 | 20 | *34* | 22 | 15 | 54 | *84* | 15 | 53 | *83* |
| 11 | 6 | 24 | *36* | 6 | 22 | *34* | 23 | 16 | 63 | *95* | 16 | 62 | *94* |
| 12 | 5 | 17 | *27* | 5 | 17 | *27* | 24 | 17 | 60 | *94* | 17 | 59 | *93* |

$H_0$: *There is no significant difference between the means of objective function values obtained by ABACO/S and BOH/S in favour of the alternative* ($\mu_{OS} \geq \mu_{BOH/S}$).

$H_1$: *ABACO/S outperforms BOH/S in terms of reducing objective function values of the solutions obtained through solving MPTALB/S problems* ($\mu_{OS} < \mu_{BOH/S}$).

Table 8-8. The results of paired sample t-test to analyse the performance of ABACO/S in terms of the objective function values of the obtained solutions

| | ***ABACO/S*** | ***BOH/S*** |
|---|---|---|
| Mean ($\mu$) | 46.17 | 47.17 |
| Standard deviation ($SD$) | 30.27 | 31.41 |
| Standard error mean | 6.18 | 6.41 |
| Variance | 916.3 | 986.9 |
| Observations | 24 | 24 |
| Pearson correlation | 0.9986 | - |
| Hypothesised mean difference | 0 | - |
| Degrees of freedom | 23 | - |
| $\boldsymbol{t\_stat}$ | **-2.477** | - |
| $\boldsymbol{p(T \leq t)}$ **one-tail** | **0.011** | - |
| $\boldsymbol{t\_critical}$ **one-tail** | **-1.714** | - |
| $p(T \leq t)$ two-tail | 0.021 | - |
| $t\_critical$ two-tail | -2.069 | - |

As could be seen from the statistical test results reported in Table 8-8, the difference between the means of objective function values found by ABACO/S ($\mu_{\mathrm{OS}} = 46.17, SD_{\mathrm{OS}} = 30.27$) and BOH/S ($\mu_{BOH/S} = 47.17, SD_{BOH/S} = 31.41$) is significant; $t(23) = -2.477, p = 0.011$. Therefore, the null hypothesis is rejected with strong evidence in favour of the alternative ($p = 0.011 < a = 0.05$ or $t\_stat = -2.477 < t\_crititical = -1.714$)). The results indicate that the developed ABACO/S algorithm finds significantly better solutions than the heuristics and truly outperforms them in terms of the obtained objective function values.

### 8.5.3. Analysing the performance of ABACO/S-GA

The performance of ABACO/S-GA is tested against its primitive version, ABACO/S, in terms of objective function values of the solutions obtained by these two algorithms. For a quick recall, multi-line stations were allowed and product models were sequenced while balancing the lines in both ABACO/S-GA and ABACO/S solutions. However, a new GA-based model sequencing approach was integrated to ABACO/S instead of conventional combinatorial sequencing and random sequencing approaches for obtaining better results. To analyse the effect of the newly integrated GA-based model sequencing mechanism on the objective function values of the solutions obtained, a paired sample t-test is conducted using the values presented in ABACO/S-GA and ABACO/S columns presented in Table 8-2 in Section 8.2. The hypotheses stated at the $a = 0.05$ significance level (95% confidence interval) for means of objective function values obtained from ABACO/S-GA ($\mu_{OSGA}$) and ABACO/S ($\mu_{OS}$) are as follows:

> $H_0$: *The integrated GA-based model sequencing mechanism has no significant positive effect on the objective function values of the obtained solutions* ($\mu_{OSGA} \geq \mu_{OS}$).
>
> $H_1$: *For the MPTALB/S problem, ABACO/S-GA finds solutions with significantly lower objective function values than ABACO/S thanks to the newly integrated GA-based model sequencing mechanism into it* ($\mu_{OSGA} < \mu_{OS}$).

As seen from the alternative hypothesis, the test is left tailed and it is sought whether $p \leq a$ (or $t\_stat < t\_crititical$) to reject the null hypothesis. When the statistical test results presented in Table 8-9 are analysed, it is clear that the null hypothesis should be rejected ($p = 0.041 < a = 0.05$ or $t\_stat = -1.813 < t\_crititical = -1.714$). The results of the analysis show that there is a significant difference between integrating

$(\mu_{OSGA} = 46.04, SD_{OSGA} = 30.12)$ and not integrating $(\mu_{OS} = 46.17, SD_{OS} = 30.27)$ the GA-based model sequencing procedure to the ABACO/S; $t(23) = -1.813, p = 0.041$. The results of the statistical test indicate that the integrated GA-based procedure has a significant positive effect on the objective function values of the obtained solutions. Therefore, it is statistically proven that the integrated GA-based procedure helps algorithm find solutions with significantly lower objective function values for the MPTALB/S problem. In other words, ABACO/S finds significantly better solutions for solved test cases of the MPTALB/S problem when a GA-based model sequencing algorithm is integrated into it. In addition to the improvement in the objective function values, for some test cases, the algorithm has found solutions with less computational effort thanks to the integrated mechanism (please refer to Chapter 7).

Table 8-9. The results of paired sample t-test to analyse the performance of ABACO/S-GA in terms of the objective function values of the obtained solutions

| | ***ABACO/S-GA*** | ***ABACO/S*** |
|---|---|---|
| Mean ($\mu$) | 46.04 | 46.17 |
| Standard deviation (SD) | 30.12 | 30.27 |
| Standard error mean | 6.15 | 6.18 |
| Variance | 907.1 | 916.3 |
| Observations | 24 | 24 |
| Pearson correlation | 0.9999 | - |
| Hypothesised mean difference | 0 | - |
| Degrees of freedom | 23 | - |
| $\boldsymbol{t\_stat}$ | **-1.813** | - |
| $\boldsymbol{p(T \leq t)}$ **one-tail** | **0.041** | - |
| $\boldsymbol{t\_critical}$ **one-tail** | **-1.714** | - |
| $p(T \leq t)$ two-tail | 0.083 | - |
| $t\_critical$ two-tail | -2.069 | - |

## 8.6. Chapter Summary

This chapter presented statistical analysis to verify the advantages of the proposed mixed-model parallel two-sided assembly line system and the capabilities of the developed algorithms, *i.e.* the ABACO and the ABACO/S which were described and computationally tested in Chapters 5 and 6, respectively. It was also tested whether integrating a GA-based model sequencing mechanism (as explained in Chapter 7) significantly affects the quality of the obtained solutions.

In the analyses performed, the effect of balancing lines together and the effect of considering the model sequencing problem together with the line balancing problem were tested in terms of objective function values of the solutions obtained under different conditions. In accordance with the results of statistical tests (namely paired sample t-tests) reported, it was emphasised that balancing two mixed-model parallel two-sided assembly lines together, where multi-line stations were allowed, helps minimise objective function values of the obtained solutions. The results indicated that solving the model sequencing problem together with the line balancing problem also helps minimise objective function values, with a very strong evidence.

Moreover, the performances of the developed ABACO and ABACO/S were compared against well-known line balancing heuristics. The effect of newly integrated GA-based model sequencing mechanism on the performance of ABACO/S was also tested. It was statistically proven that the developed algorithms outperform other heuristics in terms of the objective function values obtained. Furthermore, it was validated that integrating a GA-based model sequencing mechanism into ABACO/S (and so ABACO/S-GA was obtained) helps the algorithm find better solutions.

The next chapter will provide a more comprehensive conclusion of these statistical test results through a justification with research questions and summarise the contributions made to knowledge. Implications and limitations of the work will be identified followed by some suggestions for future research directions.

# 9

# CONCLUSIONS AND FUTURE WORK

**Contents**

## 9.1. Chapter Introduction

Conclusions of the thesis are drawn in this chapter. Section 9.2 provides answers to research questions and Section 9.3 presents concluding remarks including main contributions of the thesis in relation to the originally set objectives. Section 9.4 emphasises the implications and limitations of the research and discusses future work directions. Finally, the summary of the chapter is given Section 9.5.

## 9.2. Answers to Research Questions

As mentioned in Chapter 1, the main aim of this thesis was to introduce and deal with balancing and sequencing issues on mixed-model parallel two-sided assembly lines and to develop competitive algorithms to efficiently tackle the problems, which are described in this thesis, for such a line system in a dynamic and competitive market environment. The research questions listed in Chapter 1 are answered below in accordance with the research carried out and the experimental test results presented in the previous chapters.

**RQ1:** *If two-sided lines are put together in parallel to produce variants of a large-sized product, would this system be improved in terms of line length and/or the total number of workstations compared to independently balanced two-sided lines?*

This research question addresses utilisation of a new assembly line design by locating two-sided lines in parallel to each other with the aim of producing mixed product models on the lines. The possible advantages of the emerged mixed product model parallel two-sided assembly line configuration are subject to consideration with an emphasis on line length and the total number of workstations. A special importance was given to line length and the total number of workstations, which are the two key performance measures in the assembly line balancing domain. This is because minimising the number of workstations is a broadly sought objective in majority of the line balancing studies regardless of line type and/or solution approach and is also considered as a means of increasing the efficiency of the overall line system. In addition, line length is another mostly used performance indicator of a two-sided assembly line balancing solution, where the shorter the line is the better.

To answer this research question, first of all, a comprehensive survey has been conducted (see Chapter 2 and Chapter 3) to identify whether the proposed line configuration, the mixed-model parallel two-sided assembly line system, has already been studied in the literature. It was observed that the mixed-model parallel two-sided line configuration is an original research domain which has not been studied so far. Afterwards, more than one mixed-model two-sided assembly line, on which final product customisation tasks are performed under a mass customisation concept, has been located in parallel to each other. To identify the benefits provided by the proposed line system in terms of line length and the number of workstations, the balancing and sequencing problem of mixed-model parallel two-sided lines was modelled mathematically and illustrated with examples in Chapter 4. The three variants of a newly developed agent-based ACO algorithm were proposed as possible solution approaches for solving the MPTALB/S problem by considering multi-line stations (see Chapter 5, Chapter 6 and Chapter 7). The results of computational experiments (see Section 5.5) and their statistical analysis (see Section 8.3) made it clear that utilising the proposed mixed-model parallel two-sided assembly line system helps minimise the objective function, which is composed of the weighted summation of line length and the total number of workstations. Therefore, it was observed that putting two mixed-model two-sided assembly lines together in parallel yields a better configuration in terms of line length and the total number of workstations.

**RQ2:** *How to model and solve the problems of mixed-model parallel two-sided assembly line systems effectively? What are newly arising and possibly challenging issues in these problems and how to cope with them?*

This research question looks for answers about modelling and effectively solving problems on mixed-model parallel two-sided lines by taking the special features of this new line system into account and exploring and evaluating all aspects of the newly proposed design. Therefore, the effort of modelling and solving the mixed-model parallel two-sided assembly line problems in various ways by coping with newly arising and possibly challenging issues in doing so builds the answer to this research question. The problem was modelled mathematically using mixed-integer programming model in Chapter 4 and the constraints that must be considered carefully while solving the MPTALB/S problem were explained thoroughly. Changing product model combinations in different production cycles were shown using examples and an agent-based ACO algorithm was proposed for the generalised solution of the problem. The developed agent-based ACO algorithm was then enhanced step-by-step as the need arises during the model development and solving process. Improvements were made to consider multi-line stations and solve the model sequencing problem together with the line balancing problem. A GA-based model sequencing mechanism was also integrated to the developed approach for obtaining more powerful results in reasonable computational times (see Chapter 5, Chapter 6 and Chapter 7).

There is no doubt that the modelled line design combines the advantages of all of its component line structures, *i.e.* parallel lines, two-sided lines and mixed-model lines, as explained in Chapter 4. Producing two or more product models on each of the parallel two-sided lines increases the flexibility and helps companies to respond to changing customised demands rapidly while the characteristic of operating tasks on both sides of the line provides the opportunity of producing large-sized products more efficiently. Additionally, locating two lines in parallel to each other may result in emerging synergy between those lines. However, all of these advantages brings the complexity together.

As an advantage, the lines considered in the proposed model are independent of each other, so there is no reason to obey the same cycle time. That means lines may have different cycle times to produce their own mix of product models, which have their own set of tasks even with different processing times. There is no doubt that having different cycle times provide lines the flexibility of producing in different throughput rates. Nonetheless, this leads to a more complex problem along with the different product model changing slots of lines. Changing combinations of product models especially in multi-line stations in an environment that lines may have different cycle times seems to be one of the most challenging issues to cope with for the structured problem (this situation has been demonstrated for variants of product model mixes in Chapter 4). A quite comprehensive computational effort is required to satisfy the capacity constraints as well as the precedence relationship constraints as workloads of workstations may change from one production cycle to another. At this point, a decision must be made between the *flexibility* and the *efficiency*. To remind, the less the number of workstations and length an assembly line has, the more efficient an assembly line is.

If the former one (flexibility) is decided, lines may be balanced by considering the largest processing time of a task for all product models produced on the same line (as in

Chapter 5). In this case, it could be assumed that there is only one product model (which is composed of the largest processing times of each task) assembled on each line. As a consequence of this assumption, product models are not needed to be sequenced and the line balance obtained would be convenient to assemble product models in any sequence. Therefore, a company which applies such a policy can take action immediately upon any change in product model demands. An agent-based ACO approach (called ABACO) has been developed for the solution of this problem and been presented explicitly with illustrations and a numerical example in Chapter 5. The developed algorithm mimics the collaborative solution building procedure of real ants in nature and also provides ants the opportunity of selecting a random behaviour among ten heuristics, which are commonly used in the line balancing domain.

If the latter one (efficiency) is decided to be maximised, then the total number of different production cycles (caused by cycle time differences and changing product model combinations between the lines) are calculated; and a balancing solution is built based on the processing times of tasks for exclusive product models being assembled at that time in the relevant workstation. In other words, it is determined which product model will be assembled in which workstation and in which production cycle first, then the lines are balanced based on the specific processing times of tasks depending on the product model that will appear at each workstation. This procedure requires a complex model sequencing technique for which reason a model sequencing mechanism has been integrated to the developed algorithm (and so ABACO/S was obtained). All procedures of the algorithm have been demonstrated in Chapter 6 with a numerical example. Line balances have been constructed based on the generated product model sequences in accordance with customer demands, thanks to the two alternative options of the sequencing procedure: *(i) combinatorial sequencing* and *(ii) random sequencing*. All possible combinations of product models are tried one-by-one (based on the calculated minimum part sets of demands, whose calculations were given in Chapter 4) in the combinatorial sequencing. A user defined number of sequences among possible ones are tried in an arbitrary order in the random sequencing procedure.

One could argue that exploring search space randomly could be faster. However, it may not produce solutions as powerful as combinatorial sequencing. On the other hand, combinatorial sequencing requires a substantial amount of time to check every combination. To find a balance between these two procedures, a GA-based model sequencing procedure has been integrated to the algorithm (and so ABACO/S-GA was obtained) in Chapter 7. The GA-based mechanism generates model sequencing combinations (each chromosome represents a possible product model sequence) which would yield better solutions. The chromosomes are evolved in accordance with the performances of line balances obtained by ACO algorithm for these chromosomes and the best product model sequence which gives the best balancing solution is pursued.

A set of paired sample t-tests was accomplished to provide more comprehensive analyses about the results reported and support the arguments stated in this thesis. As mentioned above, during the problem description and algorithm development process, it was observed that the model sequencing issue was one of the most challenging problems which may affect the quality of the line balancing solutions obtained by the proposed algorithms. For that reason, the model sequencing problem was also considered together with the line balancing problem and its effect on the objective function values were examined. To remember, ABACO solved the test cases balancing the mixed-model lines together and so allowing the multi-line stations. However, the model sequencing problem was not considered in ABACO. Therefore, the same test cases were solved by ABACO/S considering the model sequencing problem as well as the line balancing problem and the results obtained from ABACO/S were compared with those obtained from ABACO. Remarkable improvements were achieved in the results reported in Section 6.4. To see whether there are statistically significant differences between *considering* ($\mu_{OS} = 46.17, \mu_{OS} = 30.27$) and *not considering* ($\mu_T = 57.33, SD_T = 41.00$) the model sequencing problem together with the line balancing problem in terms of the objective function values, a paired sample t-test was conducted in Section 8.4. The null hypothesis of *"$H_0$: There is no significant difference between the means of objective function values obtained when the model sequencing problem is considered and not considered together with the line balancing problem, in favour of the alternative ($\mu_{OS} \geq \mu_T$)"* was rejected at the confidence interval of 95%; $t(23) = -4.948, p = 0.000$. The results of the statistical test showed that considering the model sequencing problem together with the line balancing problem in mixed-model parallel two-sided lines helps minimise objective function value. Thus, the overall line system has less idle time and it produces customised product models by requiring less workspace (referred to as line length) and less number of workstations (or operators). This increases the utilisation of the resources and maximises both efficiency and productivity of the line in such a complex but flexible environment.

Another statistical analysis was conducted to confirm that balancing mixed-model parallel two-sided assembly lines together helps minimise line length and the number of workstations. The results used for this aim were obtained through solving the same test problems using ABACO under two different conditions: balancing the mixed-model parallel two-sided lines together and balancing the mixed-model parallel two-sided lines independently (or separately). The together balancing and independent balancing solutions of the same test problems were compared with each other and it was observed that better objective function values were investigated when the lines were balanced together (allowing the utilisation of multi-line stations). To statistically verify this result, a paired sample t-test has been accomplished in Section 8.3 at the significance level of $a = 0.05$. The null hypothesis of *"$H_0$: There is no significant difference between the means of objective function values obtained when the lines are balanced together and separately in favour of the alternative* $(\mu_T \geq \mu_S)$*"* was rejected with a very strong evidence. The statistical test indicated that there is a significant difference in the means of objective function values found when the mixed-model parallel two-sided lines are balanced together $(\mu_T = 57.33, SD_T = 41.00)$ and when the mixed-model parallel two-sided lines are balanced separately $(\mu_S = 59.13, SD_S = 42.59)$; $t(23) = -3.781, p = 0.000$. This emphasises that improved line balances with reduced wastes may be acquired by locating mixed-model two-sided lines in parallel to each other rather than establishing independent lines.

The performances of the developed algorithms were also assessed through test cases. As there is no related study in the literature to compare the results obtained within the scope of this research, test cases were solved using modified versions of well-known heuristic approaches in line balancing domain (as explained in Chapter 5) to comparatively assess the performances of ABACO and ABACO/S in Chapter 5 and Chapter 6, respectively. The same test cases were also solved by ABACO/S-GA and the results were compared with ABACO/S in Chapter 7.

Section 8.5 analysed the performances of the developed ABACO and ABACO/S approaches against other well-known heuristics to see whether the obtained results found by ABACO and ABACO/S significantly differ from the best results found by other heuristics. The performance of ABACO/S-GA has also been compared with the performance of ABACO/S. In Section 8.5.1, the best objective function values among the ones found by six test heuristics in Chapter 5 (reported as BOH) were statistically tested against the objective function values found by ABACO for the same test cases, under the same conditions. The results of the statistical test pointed out that there is a significant difference between the means of objective function values found by ABACO ($\mu_T = 57.33, SD_T = 41.00$) and BOH ($\mu_{BOH} = 58.25, SD_{BOH} = 41.64$) for the solved MPTALB/S problem instances; $t(23) = -4.412, p = 0.0001$. The null hypothesis, which argues that *there is no significant difference between the means of objective function values obtained by ABACO and BOH, in favour of the alternative* ($\mu_T \geq \mu_{BOH}$) has been rejected with a very strong evidence at the $a = 0.05$ significance level. The results verified that ABACO performs significantly better than other six well-known heuristics in terms of the objective functions values of the solutions obtained for the test cases solved under the same conditions.

To confirm the performance of the developed ABACO/S algorithm, another paired sample t-test was conducted in Section 8.5.2. The best objective function values among the ones found by other test heuristics (referred to as BOH/S) in Chapter 6 were also statistically compared to the results found by ABACO/S. The null hypothesis arguing that *there is no significant difference between the means of objective function values obtained by ABACO/S and BOH/S in favour of the alternative* ($\mu_{OS} \geq \mu_{BOH/S}$) has been tested at the same significance level ($a = 0.05$). In accordance with the test results, the difference between the means of objective function values found by ABACO/S ($\mu_{OS} = 46.17, SD_{OS} = 30.27$) and BOH/S ($\mu_{BOH/S} = 47.17, SD_{BOH/S} = 31.41$) is significant; $t(23) = -2.477, p = 0.011$. Therefore, the null hypothesis has been rejected with strong evidence. This suggests that the developed ABACO/S algorithm outperforms other well-known test heuristics and finds better solutions than them in terms of the obtained objective function values.

Finally, the effect of integrating a GA-based model sequencing mechanism to ABACO/S on the quality of obtained solutions in terms of line length and the number of workstations was analysed in Section 8.5.3. The objective function values obtained by ABACO/S-GA ($\mu_{OSGA} = 46.04, SD_{OSGA} = 30.12$) and ABACO/S ($\mu_{OS} = 46.17$, $SD_{OS} = 30.27$) for the same test cases were tested against the null hypothesis of *"$H_0$: The integrated GA-based model sequencing mechanism has no significant positive effect on the objective function values of the obtained solutions* ($\mu_{OSGA} \geq \mu_{OS}$)*"*. As could be seen from the results reported, the integrated GA-based model sequencing mechanism has statistically significant effect on the objective function values of the solutions obtained; $t(23) = -1.813, p = 0.041$. Specifically, the statistical test results suggest that ABACO/S-GA performs better than ABACO/S in obtaining more efficient line balancing solutions with minimised line length and the number of workstations.

## 9.3. Concluding Remarks and Research Contributions

The growing interests from customers in customised products and increasing competitions among peers necessitate companies to configure their manufacturing systems more effectively than ever before. Within this context, companies converted their single-model lines into mixed-model lines to increase the share of resources and improve their utilisation and to avoid the establishment of a new assembly line for each new product model.

Above all, this thesis proposed a new assembly line system configuration, namely mixed-model parallel two-sided assembly line system, for companies that need intelligent solutions to satisfy customised demands on time with existing resources. This is the major contribution of this thesis. The mixed-model parallel two-sided assembly line system was introduced and its main characteristics were illustrated through explanatory examples from the perspective of simultaneous balancing and sequencing. The mathematical programming model of the tackled problem was also built in order to formally describe the problem. The goal of developing mathematical model was to provide an insight about the complexity of the problem and help describe the underlying principles of the proposed approaches but not to solve the model as its high complexity made its resolution harder (even for small and medium-sized problems).

This research made it clear that locating mixed-model two-sided lines in parallel to each other helps minimise line length and the number of workstations. Thanks to the multi-line stations located in between two adjacent lines, idle times were reduced and an improved line balance was maintained. This statement was also supported by the experimental tests and statistical analysis of their results. Another statement which was confirmed through the statistical test results was that one of the key factors of a successfully implemented mixed-model parallel two-sided assembly line system is considering the model sequencing problem as well as the line balancing problem. In the literature, there are many studies, which consider these two tightly interrelated problems individually. However, this study integrated the model sequencing and the line balancing procedures to obtain more efficient solutions for the problem of *simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines*.

The development of novel agent-based ACO algorithms (namely ABACO, ABACO/S and ABACO/S-GA) for solving the MPTALB/S problem is another important contribution of this thesis. In these algorithms, different agents interact with each other to find good quality solutions for the associated problem. As an original contribution to the field, each ant selects a random behaviour from a predefined list of heuristics and builds a solution using this behaviour as a local search rule along with the pheromone value. Different cycle times were allowed for different mixed-model two-sided lines located in parallel to each other and this yielded a more complex problem where different production cycles needed to be considered to build a feasible solution.

First, the ABACO approach was developed to establish a generic balancing solution, which is compatible with any product model sequence launched, for the MPTALB/S problem. The fact beyond this idea was that the algorithm considers the maximum processing time of a task which is common among different product models on the same line. Thus, a more generic as well as flexible balancing solution was obtained regardless of the sequences of the product models on the lines. This algorithm was the first attempt in the literature to solve an assembly line balancing problem with an agent-based ACO approach.

Second, another novel agent-based ACO algorithm, called ABACO/S, was developed to simultaneously solve the model sequencing and line balancing problems on the mixed-model parallel two-sided assembly lines. ABACO/S uses particular task times for each product model and the balancing solution is built ensuring that the capacity constraints are satisfied for every product model being assembled on the line at any time. Thus, tasks are assigned to stations based on the operator’s availability which depends on the changing product model combinations on the lines. This is particularly important in multi-line stations, and feasibility and efficiency of the line configuration are assured for every product model sequence in this way. The algorithm has been designed to provide two types of model sequencing mechanisms, namely combinatorial sequencing and random sequencing. In the former, all combinations of product model sequences on different lines are tried one-by-one and the one which has the best performance measure is determined as the best solution. However, this could require a huge amount of time as each product model combination requires running a new line balancing mechanism. The latter tries a user defined number of product model combinations randomly and comes up with the best result among obtained solutions.

To increase the robustness of the ABACO/S and enhance its model sequencing capability, a GA-based model sequencing mechanism has been integrated to it. Consequently, a new hybrid algorithm – called ABACO/S-GA – has been developed to find a balance between two alternative model sequencing strategies used in the ABACO/S. The newly developed model sequencing procedure evolves chromosomes made up with sequences of product models and seeks the best product model combination that gives the best balancing solution. In accordance with the computational experiments carried out, improvements have been achieved in obtaining the same objective function values with less computational effort. Furthermore, better balancing solutions have been observed for some of the test cases solved.

In accordance with the computational tests performed, it could be clearly seen that the proposed line system and problem handling strategy have advantages over separately balanced lines and traditional solution building practices, respectively. Moreover, the developed algorithms perform well and outperform other tested heuristics. However, a more scientific analysis was required to state whether these improvements are statistically significant. For that reason, paired sample t-tests were conducted at $a = 0.05$ significance level to compare pre-improvement results with post-improvement results. Test results indicated that *(i) balancing lines together* and *(ii) considering the model sequencing problem along with the line balancing problem* help minimise objective function values of the solutions obtained for various MPTALB/S test instances.

Furthermore, to establish the performances of the developed algorithms, paired sample t-tests were accomplished to compare the results of ABACO and ABACO/S against the results obtained from heuristics. Results indicated that both algorithms outperform other test heuristics and find significantly better quality solutions in terms of the objective function values. It was also statistically proven that hybridising ABACO/S algorithm with a GA-based model sequencing mechanism helps the algorithm find better solutions in terms of objective function values.

To briefly summarise the major contributions of the research:

- This research introduced the *mixed-model parallel two-sided assembly line system* and the *simultaneous line balancing and model sequencing problem* on this system. The problem was described and typical illustrations were provided to demonstrate the dynamic situation of product model combinations on the lines when the lines have different cycle times. Moreover, a mathematical model was also developed as a means of the formal description of the introduced problem.

- This is the first attempt in the literature to solve any kind of parallel assembly line balancing problem using an ACO approach. ABACO, ABACO/S and ABACO/S-GA are novel approaches in this domain which incorporate commonly used heuristics into the nature inspired ACO algorithm by giving ants the freedom of selecting a local search rule among ten heuristics. With the help of newly developed GA-based model sequencing algorithm, the ABACO/S-GA algorithm robustly finds the same objective values (even better for some cases) with the ABACO/S by requiring less computational effort. This also is the first attempt in the literature to hybridise an agent-based ACO algorithm with a GA-based algorithm.
- A comprehensive review of the literature was presented on sub-problems of MPTALB/S problem; which are parallel assembly lines, mixed-model assembly lines, two-sided assembly lines and their combinations. The problems and their various solution techniques have also been explained thoroughly.
- The key requirements of implementing a well-balanced mixed-model parallel two-sided assembly line system were identified. Among others, it was statistically proven that balancing the mixed-model parallel two-sided lines together helps significantly minimise objective function, which is composed of line length and number of workstations, thanks to the utilised multi-line stations.
- It was numerically demonstrated and statistically verified that considering model sequencing problem simultaneously with the line balancing problem is vital in terms of maximising system efficiency, and minimising line length and number of workstations.
- New test cases were constituted using precedence relationship diagrams of well-known test problems in the literature. The newly generated task times used for the test cases are presented in Appendices A.1. to be used in future researches on mixed-model parallel-two sided assembly line problems.
- The new research areas contextualised with this research were specified and new topics that could be of interest for future researches were featured.

## 9.4. Research Implications, Limitations and Future Work

Although it is known that efficient mixed-model lines play a crucial role in competitiveness and survivability in the present global market environment, in which the growing demand for customised products increases pressure for manufacturing flexibility, mixed-model parallel two-sided lines have not been studied by academics so far. This thesis addressed balancing and sequencing problems in mixed-model parallel two-sided assembly lines for the first time in the literature and proposed three agent-based solution methods to tackle arising issues which play critical roles in doing so. The managers of existing mixed-model large-sized production lines in a just-in-time production environment may benefit from the line system and solution methods which were put forward and discussed in a wide range of spectrum here. Moreover, others whose companies apply non-mixed assembly on two-sided lines or mixed assembly on non-parallel lines might take advantage of the system described comprehensively and arguments presented explicitly in this thesis.

As is applied in this thesis, the idea of line parallelisation can be applied to a variety of problems in the entire assembly line balancing domain. In two recently published studies, Kucukkoc and Zhang (2015a, 2015b) have already considered two U-shaped lines in parallel to each other with the aim of minimising the total number of workstations needed. With this in mind, the introduced parallel U-shaped assembly line system combined the advantages of both U-shaped line configuration and parallel line configuration and helped to save even more number of workstations in compare with conventional balancing of U-shaped lines.

A new user interface has been implemented to enable users put all developed algorithms (ABACO, ABACO/S, and ABACO/S-GA) into practice using robust, portable, high-performance, and dynamic language structure of Java$^{TM}$. All required task times and precedence relationships data could be imported from spreadsheet files '.xls' (ideally from Microsoft Excel$^{TM}$ 2007 or higher). Then, following parameters and data could be defined by the user:

- algorithmic parameters; such as *alpha*, *beta*, *evaporation rate*, *colony size*, *etc.* for the ACO algorithm (and *population size*, *crossover rate*, *mutation rate*, *etc.* for the GA algorithm if GA-based model sequencing mechanism is preferred)

- problem-related data and parameters; such as *planning horizon*, *file names of task times*, *file names of precedence relationship files*, and *demands for* product models.

Some other features could also be decided and chosen based on the line structure and user preferences: *(i)* whether multi-line stations are allowed, and *(ii)* whether model sequencing problem will be considered as well as the line balancing problem; if yes, which model sequencing mechanism to be used (random, combinatorial or GA-based).

Other ten well-known heuristics, whose aim were to provide results for comparison, are also available and could be used to get very quick but reasonable quality solutions in terms of sought performance measures.

The lack of real data could be highlighted as one of this research's limitations, which may provide insight and inspiration for future studies. The reasonable strict behaviours of companies in sharing data for vital components of their competitiveness shields were one of the most important reasons hindering a real world application. Budget consideration was another affair contributing to this result. Moreover, as the computational tests in this research bases on test problems with randomly generated task processing times, there were huge differences between the processing times of same tasks belonging to different product models which yield poor balancing solutions. This also explains why workloads of some workstations were not as high as desirable in some production cycles for the given numerical example in Chapter 5. A real world application could help overcome this issue.

Particular task times belonging to each product model were used while balancing lines and sequencing product models to ensure that the obtained solution was feasible as well as efficient for each product model sequence being assembled, and capacity and precedence relationship constraints were verified. However, stochastic processing times of tasks, which were ignored in this thesis with the aim of keeping the complexity of the problem at a lower level, could be handled to represent a more realistic production environment. Also, uncertainty in demand is another important factor that could be considered in future researches and feeding lines, on which sub-assemblies are performed, could be integrated to feed the main central line where the final products are assembled.

The line configuration addressed in this research could also be extended for manufacturing of multi-model lines where setup times between product models cannot be ignorable and buffers are exceptionally needed just before or after some workstations. It is apparent that new problems will arise in this case, such as *(i)* determining time slots where setup will take place in a changing product model combination environment and *(ii)* the numbers and places of buffers where semi-finished products will be stored ready to use. Although it is hard to estimate, a trade-off could also be quested between the cost of constructing a new line and its long term income.

Development of new exact and approximate solution procedures could also be of interest for further researches to tackle the MPTALB/S problem and the challenging issues highlighted above. As the performances of meta-heuristics mostly depend on parameters used, a parameter optimisation study can also be carried out as an extension of the work.

## 9.5. Chapter Summary

Conclusions of the research were drawn in this chapter. Final remarks were given first along with the major contributions of this research to the body of knowledge followed by a discussion on research questions and their answers based on the work carried out in this thesis. Research implications were explored from a holistic perspective and some limitations for the use of proposed models were presented with possible future work opportunities.

# Appendices

## Contents

## A.1. Input Data for Test Problems

This section presents the input data used for test problems in Chapters 5, 6, and 7. Please note that the values '0', '1' and '2' given in the *Operation Side* column of *Task Processing Times* tables mean left, right, and either side, respectively.

### P9 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 0 | 2 | 4 | 2 |
| 2 | 1 | 3 | 3 | 1 |
| 3 | 2 | 2 | 2 | 1 |
| 4 | 0 | 3 | 0 | 0 |
| 5 | 1 | 4 | 2 | 3 |
| 6 | 2 | 3 | 2 | 0 |
| 7 | 2 | 0 | 3 | 3 |
| 8 | 0 | 2 | 1 | 1 |
| 9 | 2 | 1 | 2 | 2 |

### P9 - Precedence Relationships

| Task No | Immediate Predecessor(s) |
|---|---|
| 1 | - |
| 2 | - |
| 3 | - |
| 4 | 1 |
| 5 | 2 |
| 6 | 2, 3 |
| 7 | 4, 5 |
| 8 | 5 |
| 9 | 6 |

## P12 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 0 | 3 | 2 | 2 |
| 2 | 1 | 3 | 3 | 2 |
| 3 | 2 | 0 | 2 | 1 |
| 4 | 0 | 2 | 3 | 2 |
| 5 | 2 | 2 | 1 | 2 |
| 6 | 0 | 0 | 1 | 1 |
| 7 | 2 | 2 | 2 | 2 |
| 8 | 1 | 2 | 3 | 3 |
| 9 | 2 | 1 | 2 | 1 |
| 10 | 2 | 3 | 2 | 1 |
| 11 | 2 | 2 | 0 | 1 |
| 12 | 1 | 1 | 1 | 2 |

## P12 - Precedence Relationships

| Task No | Immediate Predecessor(s) |
|---|---|
| 1 | - |
| 2 | - |
| 3 | - |
| 4 | 1 |
| 5 | 2 |
| 6 | 3 |
| 7 | 4, 5 |
| 8 | 5 |
| 9 | 5, 6 |
| 10 | 7, 8 |
| 11 | 9 |
| 12 | 11 |

## P16 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 2 | 6 | 7 | 6 |
| 2 | 2 | 5 | 2 | 0 |
| 3 | 0 | 2 | 5 | 9 |
| 4 | 2 | 9 | 2 | 8 |
| 5 | 1 | 8 | 9 | 5 |
| 6 | 0 | 4 | 8 | 0 |
| 7 | 2 | 7 | 8 | 9 |
| 8 | 2 | 4 | 6 | 3 |
| 9 | 1 | 5 | 0 | 8 |
| 10 | 1 | 4 | 4 | 7 |
| 11 | 2 | 6 | 5 | 7 |
| 12 | 0 | 5 | 6 | 6 |
| 13 | 2 | 6 | 4 | 9 |
| 14 | 2 | 4 | 2 | 7 |
| 15 | 2 | 3 | 6 | 9 |
| 16 | 2 | 4 | 8 | 8 |

## P16 - Precedence Relationships

| Task No | Immediate Predecessor(s) |
|---|---|
| 1 | - |
| 2 | - |
| 3 | 1 |
| 4 | 1 |
| 5 | 2 |
| 6 | 3 |
| 7 | 4, 5 |
| 8 | 6, 7 |
| 9 | 7 |
| 10 | 7 |
| 11 | 8 |
| 12 | 9 |
| 13 | 9, 10 |
| 14 | 11 |
| 15 | 11, 12 |
| 16 | 13 |

## P24 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 0 | 3 | 3 | 0 |
| 2 | 0 | 7 | 0 | 2 |
| 3 | 1 | 7 | 1 | 1 |
| 4 | 1 | 5 | 0 | 0 |
| 5 | 0 | 4 | 6 | 1 |
| 6 | 2 | 3 | 5 | 1 |
| 7 | 1 | 4 | 8 | 5 |
| 8 | 2 | 3 | 0 | 7 |
| 9 | 2 | 6 | 4 | 4 |
| 10 | 2 | 4 | 2 | 9 |
| 11 | 0 | 4 | 8 | 3 |
| 12 | 0 | 3 | 1 | 1 |
| 13 | 2 | 3 | 5 | 3 |
| 14 | 1 | 9 | 4 | 3 |
| 15 | 1 | 5 | 1 | 4 |
| 16 | 0 | 9 | 1 | 2 |
| 17 | 2 | 2 | 7 | 3 |
| 18 | 2 | 7 | 4 | 4 |
| 19 | 2 | 9 | 2 | 1 |
| 20 | 1 | 9 | 1 | 1 |
| 21 | 0 | 8 | 9 | 7 |
| 22 | 2 | 8 | 7 | 9 |
| 23 | 1 | 9 | 9 | 5 |
| 24 | 2 | 9 | 3 | 5 |

## P24 - Precedence Relationships

| Task No | Immediate Predecessor(s) | Task No | Immediate Predecessor(s) |
|---|---|---|---|
| 1 | - | 13 | 9 |
| 2 | - | 14 | 9, 10 |
| 3 | - | 15 | 4 |
| 4 | - | 16 | 11 |
| 5 | 2 | 17 | 12 |
| 6 | 2, 3 | 18 | 13 |
| 7 | 3 | 19 | 13, 14 |
| 8 | 5 | 20 | 15 |
| 9 | 6 | 21 | 16, 17 |
| 10 | 7 | 22 | 18 |
| 11 | 1 | 23 | 19, 20 |
| 12 | 8, 9 | 24 | 20 |

## A65 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 2 | 49 | 57 | 133 |
| 2 | 2 | 49 | 68 | 71 |
| 3 | 2 | 71 | 62 | 135 |
| 4 | 2 | 26 | 145 | 110 |
| 5 | 2 | 42 | 104 | 43 |
| 6 | 2 | 30 | 47 | 101 |
| 7 | 1 | 167 | 83 | 133 |
| 8 | 1 | 91 | 95 | 126 |
| 9 | 0 | 52 | 53 | 114 |
| 10 | 0 | 153 | 265 | 200 |
| 11 | 2 | 68 | 81 | 33 |
| 12 | 2 | 52 | 64 | 29 |
| 13 | 2 | 135 | 28 | 211 |
| 14 | 2 | 54 | 87 | 67 |
| 15 | 2 | 57 | 0 | 18 |
| 16 | 0 | 151 | 86 | 58 |
| 17 | 0 | 39 | 93 | 36 |
| 18 | 1 | 194 | 53 | 27 |
| 19 | 1 | 35 | 69 | 115 |
| 20 | 2 | 119 | 86 | 15 |
| 21 | 2 | 34 | 10 | 89 |
| 22 | 2 | 38 | 86 | 48 |
| 23 | 2 | 104 | 120 | 88 |
| 24 | 2 | 84 | 52 | 89 |
| 25 | 0 | 113 | 95 | 208 |
| 26 | 1 | 72 | 9 | 5 |
| 27 | 1 | 62 | 7 | 104 |
| 28 | 1 | 272 | 47 | 84 |
| 29 | 0 | 89 | 66 | 84 |
| 30 | 0 | 49 | 127 | 63 |
| 31 | 2 | 11 | 133 | 144 |
| 32 | 2 | 45 | 91 | 70 |
| 33 | 2 | 54 | 73 | 135 |
| 34 | 2 | 106 | 128 | 268 |
| 35 | 1 | 132 | 145 | 118 |
| 36 | 2 | 52 | 124 | 17 |
| 37 | 2 | 157 | 149 | 73 |
| 38 | 2 | 109 | 141 | 90 |
| 39 | 0 | 32 | 31 | 145 |
| 40 | 1 | 32 | 144 | 182 |
| 41 | 2 | 52 | 118 | 27 |
| 42 | 2 | 193 | 256 | 103 |

## A65 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 43 | 2 | 34 | 80 | 43 |
| 44 | 1 | 34 | 21 | 29 |
| 45 | 0 | 97 | 119 | 57 |
| 46 | 2 | 37 | 102 | 52 |
| 47 | 0 | 25 | 46 | 144 |
| 48 | 0 | 89 | 118 | 50 |
| 49 | 2 | 27 | 17 | 114 |
| 50 | 2 | 50 | 79 | 5 |
| 51 | 1 | 46 | 56 | 17 |
| 52 | 2 | 46 | 140 | 94 |
| 53 | 0 | 55 | 17 | 50 |
| 54 | 2 | 118 | 86 | 63 |
| 55 | 1 | 47 | 28 | 0 |
| 56 | 2 | 164 | 39 | 65 |
| 57 | 2 | 113 | 149 | 174 |
| 58 | 0 | 69 | 23 | 18 |
| 59 | 1 | 30 | 40 | 115 |
| 60 | 2 | 25 | 48 | 84 |
| 61 | 1 | 106 | 15 | 25 |
| 62 | 2 | 23 | 59 | 1 |
| 63 | 0 | 118 | 122 | 121 |
| 64 | 0 | 155 | 44 | 108 |
| 65 | 2 | 65 | 34 | 44 |

## A65 - Precedence Relationships

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

## B148 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 1 | 2 | 16 | 24 | 69 |
| 2 | 2 | 30 | 13 | 125 |
| 3 | 2 | 7 | 25 | 109 |
| 4 | 2 | 47 | 113 | 69 |
| 5 | 2 | 29 | 16 | 63 |
| 6 | 2 | 8 | 53 | 6 |
| 7 | 2 | 39 | 76 | 123 |
| 8 | 2 | 37 | 30 | 36 |
| 9 | 2 | 32 | 60 | 88 |
| 10 | 2 | 29 | 7 | 5 |
| 11 | 2 | 17 | 99 | 54 |
| 12 | 2 | 11 | 115 | 30 |
| 13 | 2 | 32 | 120 | 35 |
| 14 | 2 | 15 | 117 | 72 |
| 15 | 0 | 53 | 28 | 64 |
| 16 | 1 | 53 | 81 | 116 |
| 17 | 2 | 8 | 43 | 2 |
| 18 | 0 | 24 | 4 | 95 |
| 19 | 1 | 24 | 18 | 58 |
| 20 | 2 | 8 | 78 | 31 |
| 21 | 1 | 7 | 61 | 68 |
| 22 | 0 | 8 | 48 | 48 |
| 23 | 0 | 14 | 115 | 96 |
| 24 | 1 | 13 | 52 | 81 |
| 25 | 1 | 10 | 123 | 124 |
| 26 | 1 | 25 | 2 | 40 |
| 27 | 0 | 11 | 24 | 59 |
| 28 | 0 | 25 | 45 | 43 |
| 29 | 2 | 11 | 47 | 80 |
| 30 | 1 | 29 | 26 | 4 |
| 31 | 2 | 25 | 50 | 33 |
| 32 | 0 | 10 | 34 | 71 |
| 33 | 1 | 14 | 29 | 92 |
| 34 | 0 | 41 | 11 | 49 |
| 35 | 1 | 42 | 117 | 54 |
| 36 | 1 | 47 | 43 | 83 |
| 37 | 1 | 7 | 91 | 34 |
| 38 | 1 | 80 | 84 | 84 |
| 39 | 1 | 7 | 109 | 31 |
| 40 | 1 | 41 | 119 | 67 |
| 41 | 1 | 47 | 21 | 3 |
| 42 | 0 | 16 | 25 | 95 |
| 43 | 0 | 32 | 80 | 37 |

## B148 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 44 | 0 | 66 | 85 | 52 |
| 45 | 0 | 80 | 46 | 21 |
| 46 | 0 | 7 | 33 | 121 |
| 47 | 0 | 41 | 94 | 4 |
| 48 | 2 | 13 | 8 | 123 |
| 49 | 0 | 47 | 121 | 45 |
| 50 | 2 | 33 | 0 | 28 |
| 51 | 0 | 34 | 27 | 30 |
| 52 | 0 | 11 | 51 | 81 |
| 53 | 0 | 118 | 59 | 99 |
| 54 | 0 | 25 | 49 | 41 |
| 55 | 1 | 7 | 3 | 52 |
| 56 | 2 | 28 | 73 | 14 |
| 57 | 0 | 12 | 96 | 43 |
| 58 | 0 | 52 | 49 | 110 |
| 59 | 2 | 14 | 41 | 74 |
| 60 | 2 | 3 | 20 | 78 |
| 61 | 2 | 3 | 86 | 25 |
| 62 | 2 | 8 | 32 | 129 |
| 63 | 2 | 16 | 48 | 46 |
| 64 | 1 | 33 | 37 | 71 |
| 65 | 2 | 8 | 63 | 89 |
| 66 | 2 | 18 | 121 | 57 |
| 67 | 2 | 10 | 31 | 63 |
| 68 | 2 | 14 | 47 | 95 |
| 69 | 1 | 28 | 15 | 60 |
| 70 | 1 | 11 | 45 | 97 |
| 71 | 1 | 18 | 3 | 107 |
| 72 | 1 | 25 | 104 | 55 |
| 73 | 2 | 40 | 30 | 60 |
| 74 | 2 | 40 | 97 | 1 |
| 75 | 2 | 101 | 165 | 129 |
| 76 | 2 | 5 | 33 | 70 |
| 77 | 2 | 28 | 76 | 64 |
| 78 | 2 | 8 | 24 | 126 |
| 79 | 2 | 111 | 43 | 149 |
| 80 | 2 | 7 | 70 | 11 |
| 81 | 2 | 26 | 103 | 24 |
| 82 | 2 | 10 | 45 | 92 |
| 83 | 2 | 21 | 68 | 114 |
| 84 | 2 | 26 | 74 | 87 |
| 85 | 2 | 20 | 8 | 7 |
| 86 | 2 | 21 | 92 | 102 |

## B148 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 87 | 2 | 47 | 6 | 4 |
| 88 | 2 | 23 | 53 | 71 |
| 89 | 2 | 13 | 59 | 87 |
| 90 | 2 | 19 | 32 | 53 |
| 91 | 2 | 115 | 0 | 25 |
| 92 | 2 | 35 | 0 | 66 |
| 93 | 0 | 26 | 40 | 95 |
| 94 | 2 | 46 | 88 | 46 |
| 95 | 2 | 20 | 60 | 72 |
| 96 | 2 | 31 | 4 | 119 |
| 97 | 2 | 19 | 52 | 22 |
| 98 | 2 | 34 | 54 | 28 |
| 99 | 2 | 51 | 29 | 91 |
| 100 | 2 | 39 | 63 | 4 |
| 101 | 2 | 30 | 15 | 81 |
| 102 | 2 | 26 | 30 | 127 |
| 103 | 2 | 13 | 57 | 47 |
| 104 | 2 | 45 | 107 | 52 |
| 105 | 2 | 58 | 129 | 119 |
| 106 | 2 | 28 | 70 | 78 |
| 107 | 2 | 8 | 67 | 80 |
| 108 | 2 | 43 | 68 | 40 |
| 109 | 2 | 40 | 39 | 88 |
| 110 | 2 | 34 | 69 | 111 |
| 111 | 2 | 23 | 103 | 75 |
| 112 | 0 | 162 | 128 | 87 |
| 113 | 0 | 11 | 113 | 28 |
| 114 | 2 | 19 | 19 | 96 |
| 115 | 2 | 14 | 70 | 121 |
| 116 | 2 | 31 | 80 | 36 |
| 117 | 2 | 32 | 55 | 10 |
| 118 | 2 | 26 | 32 | 57 |
| 119 | 2 | 55 | 55 | 69 |
| 120 | 2 | 31 | 100 | 3 |
| 121 | 2 | 32 | 85 | 35 |
| 122 | 2 | 26 | 59 | 71 |
| 123 | 2 | 19 | 48 | 51 |
| 124 | 2 | 14 | 45 | 32 |
| 125 | 2 | 19 | 44 | 119 |
| 126 | 2 | 48 | 51 | 120 |
| 127 | 2 | 55 | 79 | 33 |
| 128 | 0 | 8 | 109 | 43 |
| 129 | 0 | 11 | 123 | 106 |

## B148 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 130 | 0 | 27 | 90 | 1 |
| 131 | 0 | 18 | 46 | 48 |
| 132 | 2 | 36 | 3 | 119 |
| 133 | 0 | 23 | 74 | 109 |
| 134 | 1 | 20 | 80 | 85 |
| 135 | 2 | 46 | 29 | 63 |
| 136 | 2 | 64 | 78 | 91 |
| 137 | 0 | 22 | 126 | 31 |
| 138 | 2 | 15 | 28 | 6 |
| 139 | 2 | 34 | 122 | 48 |
| 140 | 2 | 22 | 54 | 42 |
| 141 | 0 | 151 | 90 | 30 |
| 142 | 1 | 148 | 24 | 106 |
| 143 | 0 | 64 | 32 | 55 |
| 144 | 0 | 170 | 37 | 9 |
| 145 | 1 | 137 | 4 | 89 |
| 146 | 1 | 64 | 24 | 28 |
| 147 | 0 | 78 | 24 | 26 |
| 148 | 1 | 78 | 86 | 111 |

## B148 - Precedence Relationships

| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 1 | 5, 6, 7, 8 | 19 | 20 |
| 2 | 3 | 20 | 21, 22, 23, 24 |
| 3 | 4, 5, 6, 7 | 21 | 25, 26, 27, 28 |
| 4 | 8 | 22 | 25, 26, 27, 28 |
| 5 | 14 | 23 | 25, 26, 27, 28 |
| 6 | 9 | 24 | 25, 26, 27, 28 |
| 7 | 14 | 25 | 29 |
| 8 | 10 | 26 | 29 |
| 9 | 14 | 27 | 29 |
| 10 | 14 | 28 | 29 |
| 11 | 12 | 29 | 31 |
| 12 | 13 | 30 | - |
| 13 | - | 31 | 36 |
| 14 | 15, 16 | 32 | 34 |
| 15 | 17 | 33 | 35 |
| 16 | 17 | 34 | 36 |
| 17 | 18, 19 | 35 | 36 |
| 18 | 20 | 36 | 37 |

**B148 Precedence Relationships (Continued)**

| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 37 | 38, 45 | 81 | 106 |
| 38 | 39 | 82 | 83, 89, 143, 146 |
| 39 | 40 | 83 | - |
| 40 | 41, 48, 55 | 84 | 85 |
| 41 | - | 85 | - |
| 42 | 43 | 86 | - |
| 43 | 44 | 87 | - |
| 44 | - | 88 | 111 |
| 45 | 46 | 89 | 90 |
| 46 | 47 | 90 | 79 |
| 47 | 48, 49, 55 | 91 | 105 |
| 48 | - | 92 | 135 |
| 49 | - | 93 | - |
| 50 | 51 | 94 | - |
| 51 | 53, 69 | 95 | 101 |
| 52 | 53 | 96 | 104 |
| 53 | - | 97 | - |
| 54 | 133 | 98 | 101 |
| 55 | 54, 72, 76, 87, 88 | 99 | 100 |
| 56 | 73 | 100 | 101 |
| 57 | 79 | 101 | 102, 103 |
| 58 | 84, 86 | 102 | 127 |
| 59 | 75, 87 | 103 | 127 |
| 60 | - | 104 | - |
| 61 | 62 | 105 | 119 |
| 62 | 63 | 106 | 107 |
| 63 | 67 | 107 | 108 |
| 64 | 65, 71, 72 | 108 | 109 |
| 65 | 66, 99 | 109 | 110 |
| 66 | 67 | 110 | - |
| 67 | 68 | 111 | 112 |
| 68 | 95, 98 | 112 | 113 |
| 69 | 82 | 113 | 114, 116, 120, 123, 128 |
| 70 | 71 | 114 | 115 |
| 71 | - | 115 | 125 |
| 72 | 134 | 116 | 117 |
| 73 | 84, 86, 87, 88, 96 | 117 | 118 |
| 74 | 75 | 118 | 126 |
| 75 | 88, 97 | 119 | - |
| 76 | 77 | 120 | 121 |
| 77 | 78 | 121 | 122 |
| 78 | 79 | 122 | 126 |
| 79 | 80 | 123 | 124 |
| 80 | 81 | 124 | 125 |

## B148 Precedence Relationships (Continued)

| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 125 | - | 137 | - |
| 126 | - | 138 | 139 |
| 127 | - | 139 | 140 |
| 128 | 129 | 140 | - |
| 129 | 130 | 141 | 142 |
| 130 | 131, 137 | 142 | 143, 146, 147, 148 |
| 131 | - | 143 | - |
| 132 | 135 | 144 | 145 |
| 133 | 135 | 145 | 147, 148 |
| 134 | 135 | 146 | - |
| 135 | 136 | 147 | - |
| 136 | - | 148 | - |

## A205 - Task Processing Times

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| *1* | 2 | *39* | 151 | 204 |
| 2 | 2 | 42 | 104 | 75 |
| 3 | 1 | 261 | 52 | 126 |
| 4 | 0 | 261 | 447 | 394 |
| 5 | 2 | 157 | 75 | 10 |
| 6 | 2 | 90 | 7 | 139 |
| 7 | 1 | 54 | 167 | 145 |
| 8 | 1 | 67 | 296 | 168 |
| 9 | 1 | 30 | 77 | 48 |
| 10 | 1 | 106 | 124 | 200 |
| 11 | 1 | 32 | 34 | 89 |
| 12 | 1 | 62 | 79 | 19 |
| 13 | 0 | 54 | 176 | 188 |
| 14 | 0 | 67 | 8 | 192 |
| 15 | 0 | 30 | 116 | 172 |
| 16 | 0 | 106 | 69 | 297 |
| 17 | 0 | 32 | 225 | 15 |
| 18 | 0 | 62 | 300 | 37 |
| 19 | 2 | 56 | 156 | 234 |
| 20 | 2 | 67 | 35 | 224 |
| 21 | 2 | 86 | 71 | 4 |
| 22 | 2 | 37 | 12 | 59 |
| 23 | 2 | 41 | 64 | 16 |
| 24 | 2 | 72 | 105 | 3 |
| 25 | 1 | 86 | 61 | 40 |
| 26 | 0 | 16 | 54 | 355 |
| 27 | 1 | 51 | 167 | 107 |
| 28 | 1 | 66 | 29 | 116 |
| 29 | 1 | 41 | 285 | 181 |
| 30 | 1 | 72 | 30 | 78 |
| 31 | 1 | 51 | 36 | 16 |
| 32 | 1 | 16 | 49 | 5 |
| 33 | 1 | 15 | 13 | 65 |
| 34 | 0 | 15 | 84 | 63 |
| 35 | 2 | 85 | 34 | 241 |
| 36 | 2 | 59 | 384 | 116 |
| 37 | 0 | 23 | 26 | 110 |
| 38 | 0 | 13 | 115 | 179 |
| 39 | 0 | 19 | 114 | 36 |
| 40 | 2 | 108 | 203 | 7 |
| 41 | 2 | 214 | 166 | 64 |
| 42 | 2 | 80 | 46 | 180 |
| 43 | 0 | 37 | 40 | 96 |

## A205 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 44 | 0 | 84 | 28 | 106 |
| 45 | 0 | 18 | 72 | 29 |
| 46 | 0 | 12 | 51 | 7 |
| 47 | 0 | 29 | 86 | 252 |
| 48 | 0 | 37 | 184 | 179 |
| 49 | 0 | 13 | 154 | 219 |
| 50 | 0 | 70 | 91 | 111 |
| 51 | 0 | 217 | 304 | 61 |
| 52 | 0 | 72 | 167 | 107 |
| 53 | 0 | 85 | 185 | 53 |
| 54 | 1 | 43 | 49 | 34 |
| 55 | 1 | 97 | 116 | 59 |
| 56 | 1 | 37 | 206 | 113 |
| 57 | 1 | 13 | 36 | 143 |
| 58 | 1 | 35 | 73 | 299 |
| 59 | 1 | 217 | 102 | 219 |
| 60 | 1 | 72 | 199 | 62 |
| 61 | 1 | 85 | 373 | 154 |
| 62 | 2 | 25 | 137 | 44 |
| 63 | 2 | 37 | 210 | 89 |
| 64 | 2 | 37 | 38 | 139 |
| 65 | 2 | 103 | 41 | 253 |
| 66 | 2 | 140 | 424 | 425 |
| 67 | 2 | 49 | 59 | 152 |
| 68 | 2 | 35 | 143 | 139 |
| 69 | 2 | 51 | 51 | 163 |
| 70 | 2 | 88 | 221 | 123 |
| 71 | 2 | 53 | 341 | 288 |
| 72 | 2 | 144 | 301 | 102 |
| 73 | 2 | 337 | 83 | 394 |
| 74 | 2 | 107 | 184 | 156 |
| 75 | 2 | 371 | 100 | 444 |
| 76 | 2 | 97 | 276 | 11 |
| 77 | 2 | 166 | 104 | 6 |
| 78 | 0 | 92 | 87 | 50 |
| 79 | 1 | 92 | 160 | 95 |
| 80 | 2 | 106 | 210 | 238 |
| 81 | 2 | 49 | 257 | 252 |
| 82 | 2 | 92 | 219 | 285 |
| 83 | 2 | 371 | 256 | 49 |
| 84 | 2 | 87 | 57 | 248 |
| 85 | 2 | 162 | 183 | 435 |
| 86 | 2 | 96 | 208 | 154 |

## A205 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 87 | 2 | 79 | 267 | 22 |
| 88 | 2 | 96 | 175 | 134 |
| 89 | 2 | 42 | 262 | 309 |
| 90 | 1 | 88 | 128 | 404 |
| 91 | 1 | 90 | 46 | 39 |
| 92 | 1 | 97 | 196 | 172 |
| 93 | 1 | 270 | 199 | 29 |
| 94 | 2 | 452 | 363 | 91 |
| 95 | 1 | 48 | 319 | 297 |
| 96 | 2 | 338 | 423 | 188 |
| 97 | 2 | 34 | 140 | 196 |
| 98 | 2 | 65 | 39 | 178 |
| 99 | 2 | 50 | 120 | 136 |
| 100 | 2 | 112 | 25 | 122 |
| 101 | 2 | 48 | 197 | 137 |
| 102 | 2 | 117 | 287 | 42 |
| 103 | 2 | 50 | 170 | 179 |
| 104 | 1 | 68 | 123 | 370 |
| 105 | 0 | 232 | 84 | 6 |
| 106 | 0 | 122 | 396 | 410 |
| 107 | 2 | 151 | 222 | 120 |
| 108 | 0 | 31 | 76 | 105 |
| 109 | 2 | 97 | 16 | 42 |
| 110 | 1 | 308 | 426 | 429 |
| 111 | 0 | 116 | 196 | 151 |
| 112 | 1 | 312 | 291 | 22 |
| 113 | 2 | 34 | 136 | 195 |
| 114 | 0 | 128 | 15 | 83 |
| 115 | 2 | 54 | 89 | 3 |
| 116 | 1 | 175 | 180 | 76 |
| 117 | 2 | 55 | 111 | 221 |
| 118 | 2 | 306 | 368 | 15 |
| 119 | 2 | 59 | 198 | 122 |
| 120 | 2 | 59 | 204 | 46 |
| 121 | 2 | 66 | 130 | 234 |
| 122 | 2 | 66 | 118 | 99 |
| 123 | 2 | 23 | 95 | 127 |
| 124 | 2 | 244 | 250 | 50 |
| 125 | 2 | 54 | 83 | 292 |
| 126 | 1 | 294 | 10 | 266 |
| 127 | 2 | 84 | 136 | 110 |
| 128 | 2 | 61 | 113 | 127 |
| 129 | 2 | 57 | 237 | 177 |

## A205 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 130 | 1 | 38 | 89 | 221 |
| *131* | 2 | *57* | 2 | 4 |
| *132* | 1 | *129* | 122 | 211 |
| *133* | 1 | *276* | 115 | 238 |
| 134 | 1 | 445 | 383 | 309 |
| 135 | 0 | 68 | 81 | 244 |
| 136 | 0 | 53 | 89 | 78 |
| 137 | 2 | 49 | 138 | 128 |
| 138 | 2 | 92 | 71 | 127 |
| 139 | 2 | 236 | 339 | 103 |
| 140 | 0 | 116 | 216 | 259 |
| 141 | 0 | 265 | 314 | 133 |
| 142 | 0 | 149 | 270 | 357 |
| 143 | 0 | 74 | 118 | 176 |
| 144 | 2 | 332 | 306 | 384 |
| 145 | 2 | 324 | 264 | 22 |
| 146 | 0 | 104 | 253 | 119 |
| 147 | 0 | 51 | 65 | 238 |
| 148 | 1 | 58 | 131 | 98 |
| 149 | 1 | 67 | 49 | 148 |
| 150 | 1 | 49 | 39 | 84 |
| 151 | 2 | 107 | 364 | 272 |
| 152 | 0 | 38 | 17 | 31 |
| 153 | 0 | 27 | 98 | 33 |
| 154 | 2 | 68 | 41 | 78 |
| 155 | 2 | 207 | 386 | 168 |
| 156 | 2 | 202 | 227 | 177 |
| 157 | 2 | 83 | 113 | 189 |
| 158 | 1 | 35 | 84 | 35 |
| 159 | 1 | 58 | 61 | 40 |
| 160 | 2 | 42 | 108 | 70 |
| 161 | 1 | 68 | 165 | 100 |
| 162 | 1 | 68 | 139 | 74 |
| 163 | 1 | 68 | 44 | 42 |
| 164 | 1 | 103 | 71 | 90 |
| 165 | 1 | 103 | 35 | 108 |
| 166 | 1 | 103 | 94 | 46 |
| 167 | 1 | 103 | 87 | 58 |
| 168 | 1 | 103 | 134 | 44 |
| 169 | 0 | 68 | 19 | 37 |
| 170 | 0 | 103 | 138 | 23 |
| 171 | 0 | 68 | 28 | 5 |
| 172 | 0 | 103 | 342 | 431 |

## A205 Task Processing Times (Continued)

| Task No | Operation Side | Processing Time | | |
|---|---|---|---|---|
| | | Product Model A | Product Model B | Product Model C |
| 173 | 0 | 103 | 93 | 258 |
| 174 | 0 | 68 | 7 | 42 |
| 175 | 0 | 103 | 230 | 201 |
| 176 | 0 | 103 | 185 | 139 |
| 177 | 2 | 10 | 69 | 64 |
| 178 | 2 | 187 | 96 | 145 |
| 179 | 0 | 134 | 82 | 163 |
| 180 | 0 | 89 | 156 | 147 |
| 181 | 0 | 58 | 113 | 148 |
| 182 | 0 | 49 | 120 | 15 |
| 183 | 0 | 134 | 254 | 203 |
| 184 | 0 | 53 | 129 | 21 |
| 185 | 2 | 334 | 50 | 380 |
| 186 | 1 | 24 | 84 | 56 |
| 187 | 1 | 76 | 85 | 56 |
| 188 | 0 | 76 | 57 | 36 |
| 189 | 2 | 192 | 201 | 98 |
| 190 | 2 | 98 | 85 | 67 |
| 191 | 1 | 258 | 187 | 241 |
| 192 | 2 | 165 | 251 | 189 |
| 193 | 1 | 38 | 3 | 62 |
| 194 | 2 | 115 | 262 | 89 |
| 195 | 0 | 83 | 68 | 39 |
| 196 | 1 | 56 | 21 | 17 |
| 197 | 1 | 29 | 62 | 39 |
| 198 | 1 | 303 | 433 | 225 |
| 199 | 1 | 18 | 0 | 43 |
| 200 | 1 | 29 | 56 | 37 |
| 201 | 0 | 154 | 126 | 248 |
| 202 | 0 | 90 | 74 | 18 |
| 203 | 0 | 93 | 95 | 38 |
| 204 | 2 | 94 | 118 | 36 |
| 205 | 2 | 165 | 203 | 186 |

## A205 - Precedence Relationships

| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 1 | 36 | 45 | 46, 48, 51, 53 |
| 2 | 3, 4 | 46 | 47 |
| 3 | 5 | 47 | 92 |
| 4 | 5 | 48 | 49 |
| 5 | 7, 13 | 49 | 50 |
| 6 | 36 | 50 | 92 |
| 7 | 8 | 51 | 52 |
| 8 | 9 | 52 | 92 |
| 9 | 10 | 53 | 92 |
| 10 | 11 | 54 | 55 |
| 11 | 12 | 55 | 56, 59, 61 |
| 12 | 36 | 56 | 57 |
| 13 | 14 | 57 | 58 |
| 14 | 15 | 58 | 92 |
| 15 | 16 | 59 | 60 |
| 16 | 17 | 60 | 92 |
| 17 | 18 | 61 | 92 |
| 18 | 36 | 62 | 63 |
| 19 | 36 | 63 | 64 |
| 20 | 22 | 64 | 65, 68 |
| 21 | 22 | 65 | 66 |
| 22 | 23 | 66 | 67 |
| 23 | 24, 34 | 67 | 80 |
| 24 | 26, 27, 28 | 68 | 80 |
| 25 | 28 | 69 | 70 |
| 26 | 35 | 70 | 71 |
| 27 | 35 | 71 | 73 |
| 28 | 29 | 72 | 73 |
| 29 | 30, 33 | 73 | 74 |
| 30 | 31, 32 | 74 | 76 |
| 31 | 35 | 75 | 92 |
| 32 | 35 | 76 | 77, 78, 79 |
| 33 | 35 | 77 | 80, 82 |
| 34 | 35 | 78 | 80 |
| 35 | 36 | 79 | 80 |
| 36 | 37, 40, 41, 42, 62, 69, 72, 75, 83, 110, 111, 112 | 80 | 81 |
| 37 | 38 | 81 | 84 |
| 38 | 39 | 82 | 92 |
| 39 | 45 | 83 | 92 |
| 40 | 43, 54 | 84 | 85 |
| 41 | 92 | 85 | 86, 88, 90 |
| 42 | 43, 54 | 86 | 87 |
| 43 | 44 | 87 | 92 |
| 44 | 45 | 88 | 89 |

## A205 Precedence Relationships (Continued)

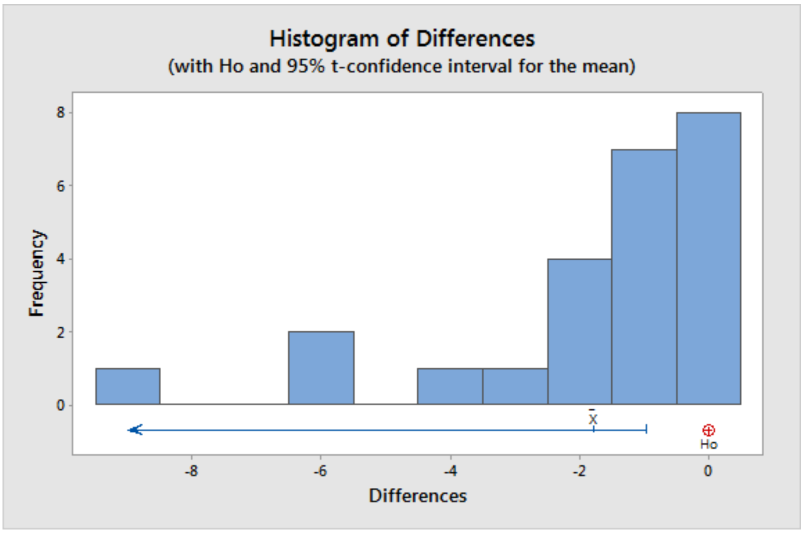
| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 89 | 92 | 130 | 136 |
| 90 | 91 | 131 | 132 |
| 91 | 92 | 132 | 133 |
| 92 | 93, 94, 95, 96, 97, 98, 99 | 133 | 189 |
| 93 | 135 | 134 | 189 |
| 94 | 135 | 135 | 136, 137, 138, 139, 140, 141, 142, 144, 145, 147, 148, 149, 150, 151, 152, 153, 158 |
| 95 | 113 | 136 | 189 |
| 96 | 113 | 137 | 160 |
| 97 | 100 | 138 | 160 |
| 98 | 100 | 139 | 160 |
| 99 | 100 | 140 | 143 |
| 100 | 101, 103, 105, 109, 130, 131, 134 | 141 | 143 |
| 101 | 102 | 142 | 143 |
| 102 | 113 | 143 | 160 |
| 103 | 104 | 144 | 160 |
| 104 | 113 | 145 | 146 |
| 105 | 106, 107 | 146 | 160 |
| 106 | 108 | 147 | 160 |
| 107 | 108 | 148 | 160 |
| 108 | 113 | 149 | 160 |
| 109 | 113 | 150 | 160 |
| 110 | 113 | 151 | 160 |
| 111 | 113 | 152 | 160 |
| 112 | 113 | 153 | 154 |
| 113 | 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 161, 162, 163, 169, 171, 174, 203, 204, 205 | 154 | 155 |
| 114 | 160 | 155 | 156 |
| 115 | 160 | 156 | 157 |
| 116 | 160 | 157 | 189 |
| 117 | 160 | 158 | 159 |
| 118 | 126 | 159 | 189 |
| 119 | 126 | 160 | 164, 170, 178, 179, 184 |
| 120 | 126 | 161 | 167 |
| 121 | 126 | 162 | 165 |
| 122 | 126 | 163 | 164 |
| 123 | 126 | 164 | 165 |
| 124 | 125 | 165 | 166 |
| 125 | 126 | 166 | 167 |
| 126 | 127, 128, 129 | 167 | 168 |
| 127 | 135 | 168 | 177 |
| 128 | 135 | 169 | 170 |
| 129 | 135 | 170 | 172 |

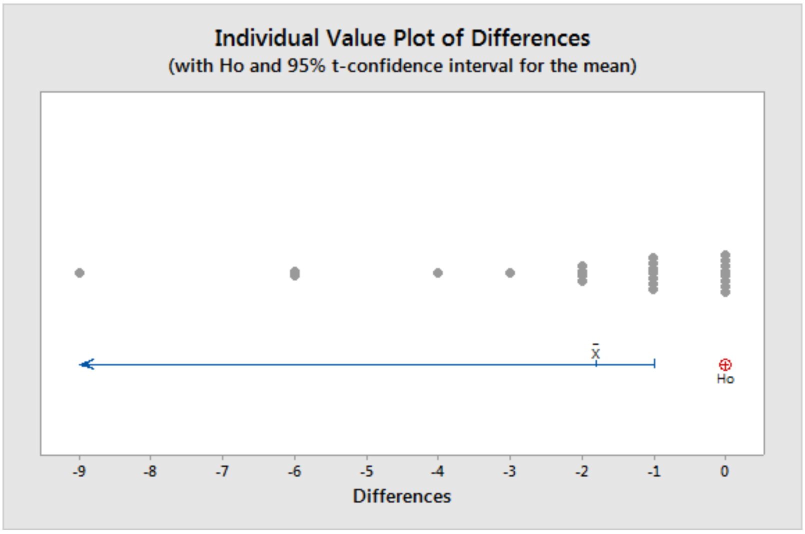
## A205 Precedence Relationships (Continued)

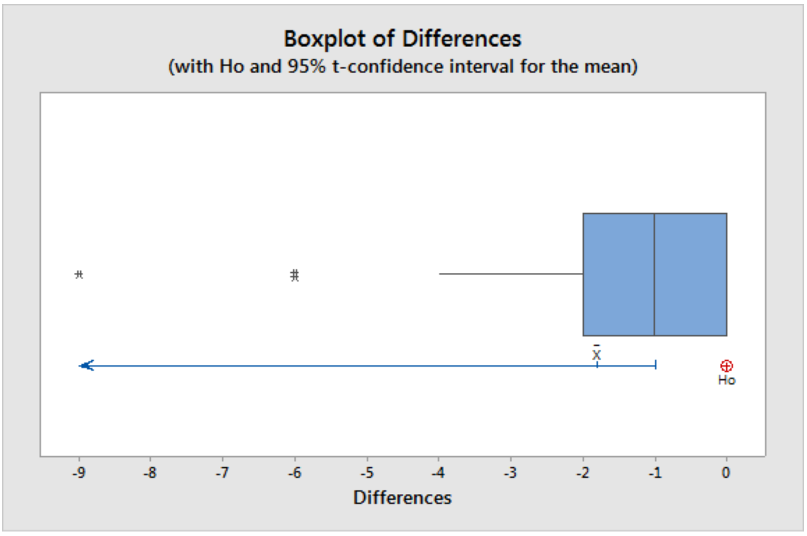
| Task No | Immediate Successor(s) | Task No | Immediate Successor(s) |
|---|---|---|---|
| 171 | 172 | 189 | 190, 191, 193 |
| 172 | 173 | 190 | - |
| 173 | 175 | 191 | 192 |
| 174 | 175 | 192 | - |
| 175 | 176 | 193 | - |
| 176 | 177 | 194 | 197 |
| 177 | 185, 186, 187, 188, 194, 195 | 195 | 196 |
| 178 | 180 | 196 | 197 |
| 179 | 180 | 197 | 198, 199, 201 |
| 180 | 181, 183 | 198 | - |
| 181 | 182 | 199 | 200 |
| 182 | - | 200 | - |
| 183 | - | 201 | 202 |
| 184 | - | 202 | - |
| 185 | 189 | 203 | - |
| 186 | 189 | 204 | - |
| 187 | 189 | 205 | - |
| 188 | 189 | - | - |

## A.2. Histograms, Individual Value Plots and Boxplots of Differences for Statistical Tests

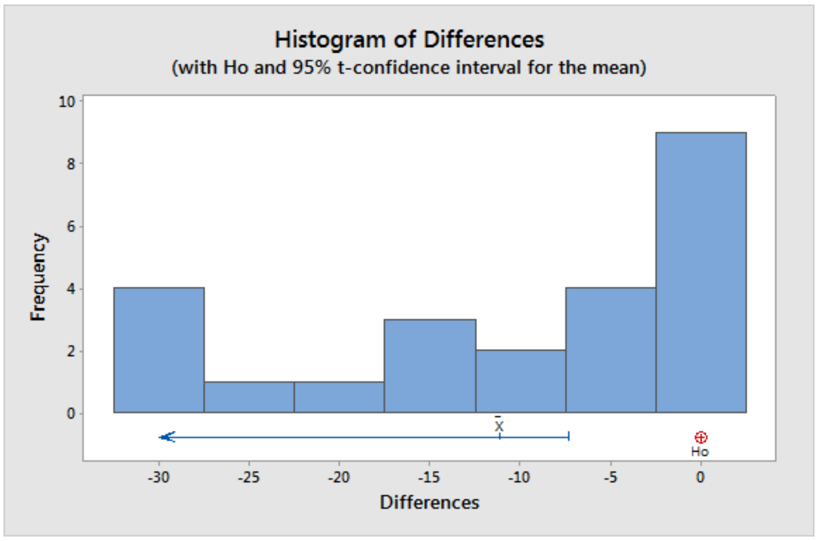
### A.2.1. Balancing lines together

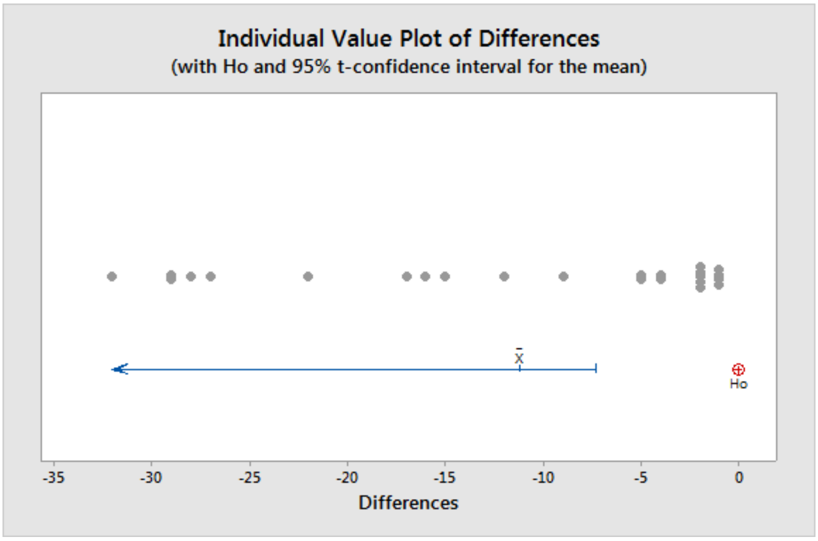


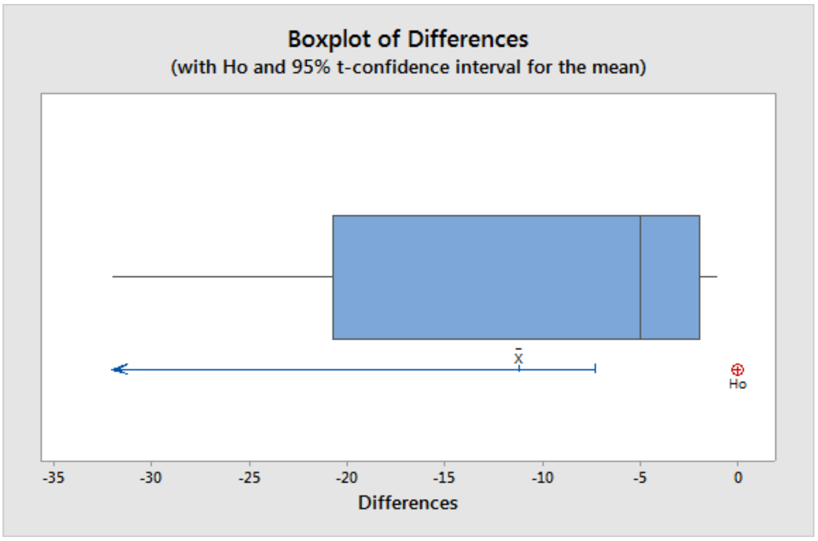




### A.2.2. Considering the model sequencing and line balancing problems together

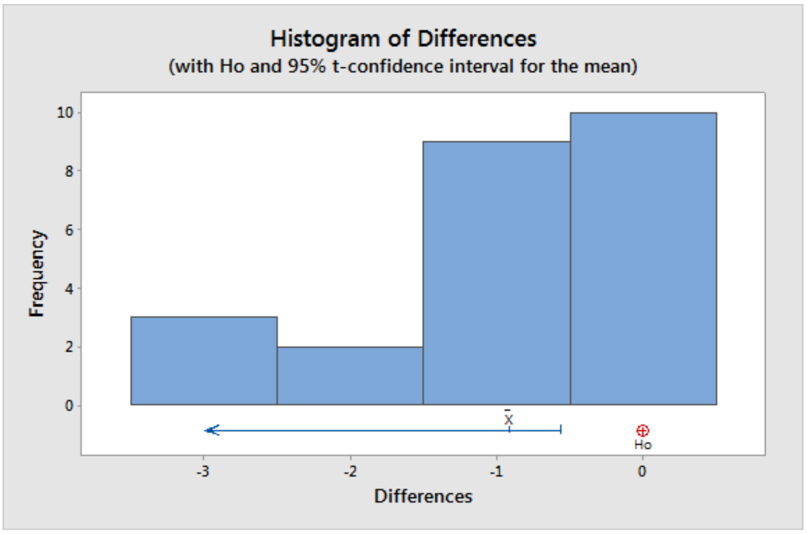


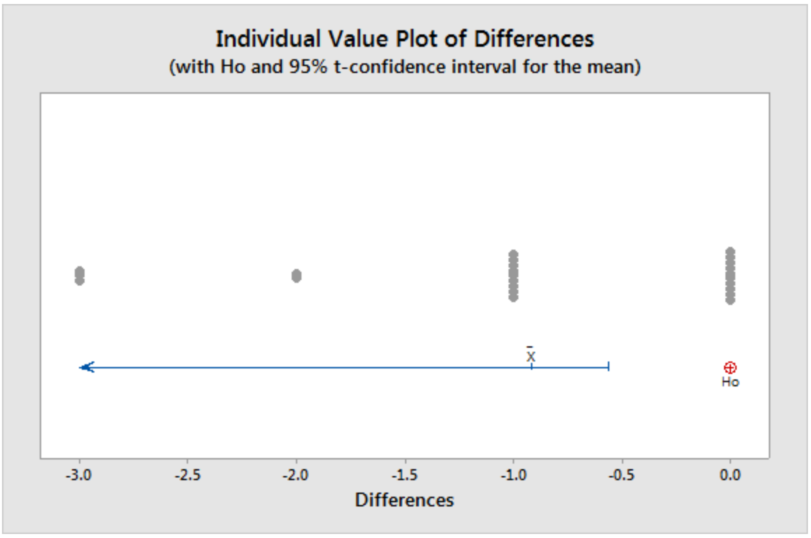


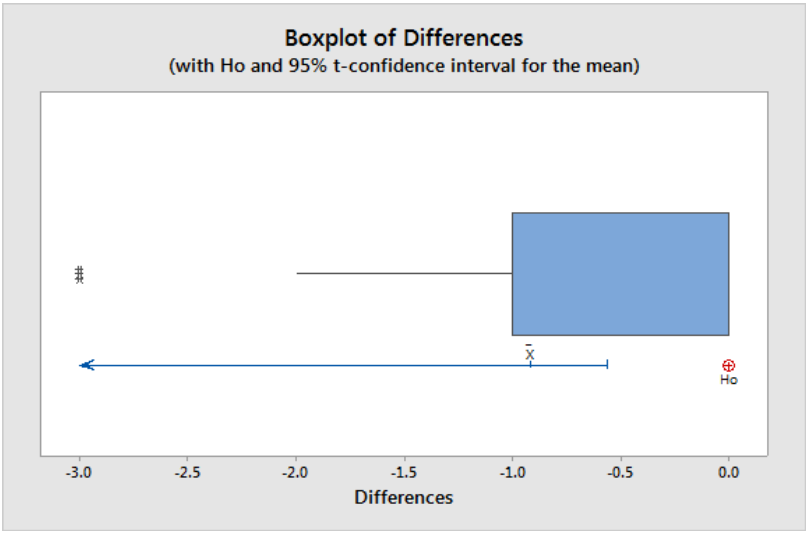


## A.2.3. The performances of the proposed algorithms

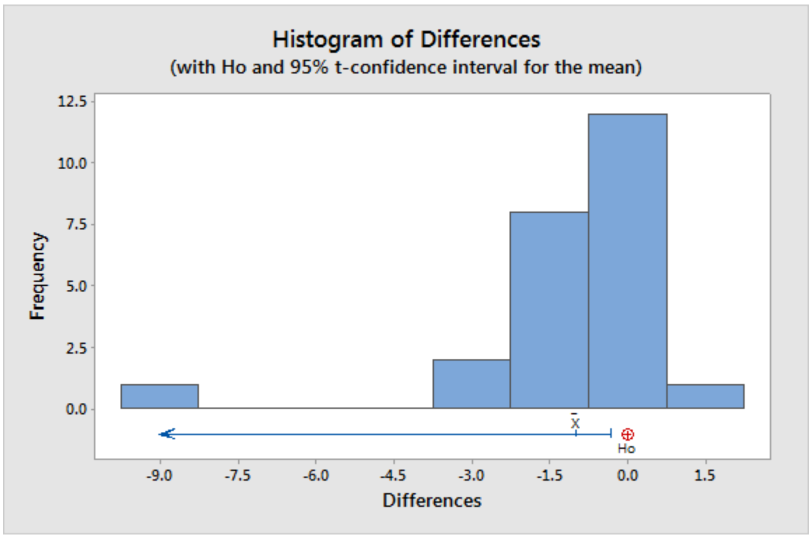
### A.2.3.1. ABACO versus BOH

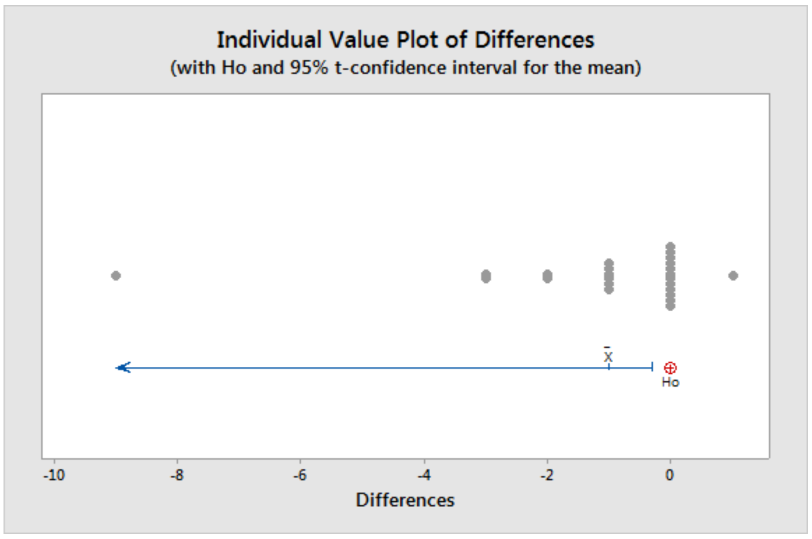


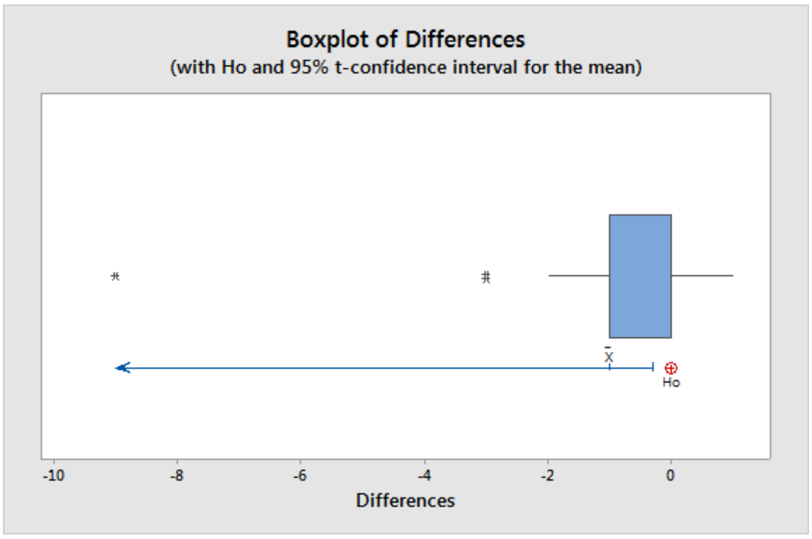




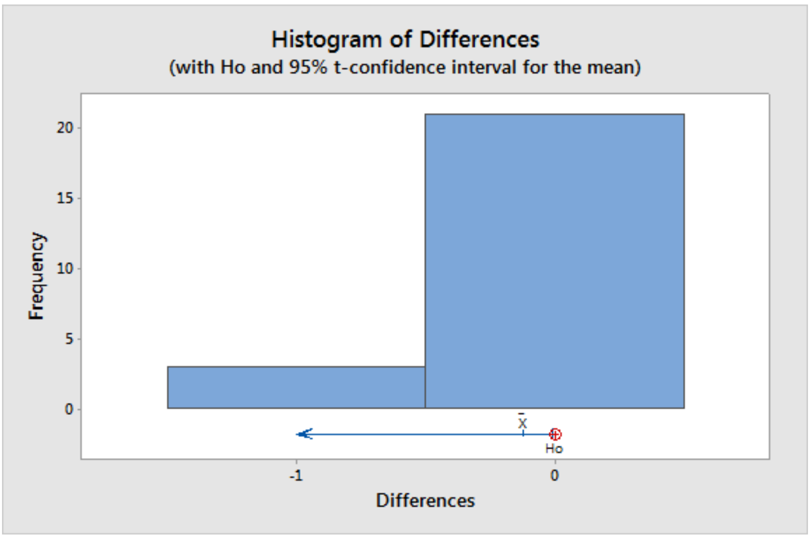
## A.2.3.2. ABACO/S versus BOH/S

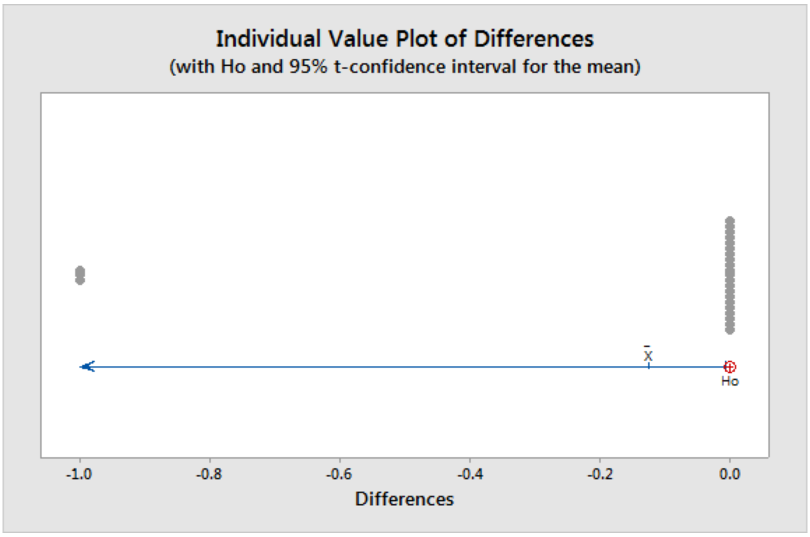


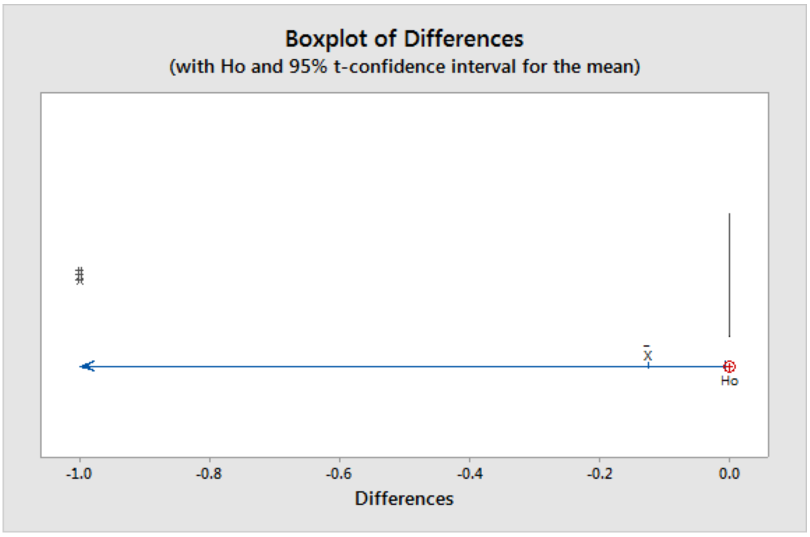




### A.2.3.3. ABACO/S-GA versus ABACO/S







# Bibliography

# Bibliography

AASE, G. R., OLSON, J. R. & SCHNIEDERJANS, M. J. 2004. U-shaped assembly line layouts and their impact on labor productivity: An experimental study. *European Journal of Operational Research,* 156**,** 698-711.

AGRAWAL, S. & TIWARI, M. K. 2008. A collaborative ant colony algorithm to stochastic mixed-model U-shaped disassembly line balancing and sequencing problem. *International Journal of Production Research,* 46**,** 1405-1429.

AKGUNDUZ, O. S. & TUNALI, S. 2010. An adaptive genetic algorithm approach for the mixed-model assembly line sequencing problem. *International Journal of Production Research,* 48**,** 5157-5179.

AKPINAR, S. & BAYHAN, G. M. 2011. A hybrid genetic algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints. *Engineering Applications of Artificial Intelligence,* 24**,** 449-457.

AKPINAR, S., BAYHAN, G. M. & BAYKASOGLU, A. 2013. Hybridizing ant colony optimization via genetic algorithm for mixed-model assembly line balancing problem with sequence dependent setup times between tasks. *Applied Soft Computing,* 13**,** 574-589.

AMINI, M., WAKOLBINGER, T., RACER, M. & NEJAD, M. G. 2012. Alternative supply chain production-sales policies for new product diffusion: An agent-based modeling and simulation approach. *European Journal of Operational Research,* 216**,** 301-311.

ANOSIKE, A. I. & ZHANG, D. Z. 2009. An agent-based approach for integrating manufacturing operations. *International Journal of Production Economics,* 121**,** 333-352.

ANUSSORNNITISARN, P., NOF, S. Y. & ETZION, O. 2005. Decentralized control of cooperative and autonomous agents for solving the distributed resource allocation problem. *International Journal of Production Economics,* 98**,** 114-128.

ARCUS, A. 1966. COMSOAL, A Computer Method of Sequencing Operations for Assembly Lines. *The International Journal of Production Research,* 4**,** 259-277.

ARCUS, A. L. 1963. *An analysis of a computer method of sequencing assembly line operations.* Ph.D. Dissertation, University of California, Berkeley, USA.

ASKIN, R. G. & ZHOU, M. 1997. A parallel station heuristic for the mixed-model production line balancing problem. *International Journal of Production Research,* 35**,** 3095-3106.

BARD, J. F. 1989. Assembly Line Balancing with Parallel Workstations and Dead Time. *International Journal of Production Research,* 27**,** 1005-1018.

BARD, J. F., DAREL, E. & SHTUB, A. 1992. An Analytic Framework for Sequencing Mixed Model Assembly Lines. *International Journal of Production Research,* 30**,** 35-48.

BARTHOLDI, J. J. 1993. Balancing 2-Sided Assembly Lines - a Case-Study. *International Journal of Production Research,* 31**,** 2447-2461.

BATTAÏA, O. & DOLGUI, A. 2013. A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics,* 142**,** 259-277.

BATTINI, D., FACCIO, M., FERRARI, E., PERSONA, A. & SGARBOSSA, F. 2007. Design configuration for a mixed-model assembly system in case of low product demand. *International Journal of Advanced Manufacturing Technology,* 34**,** 188-200.

BATTINI, D., FACCIO, M., PERSONA, A. & SGARBOSSA, F. 2009. Balancing-sequencing procedure for a mixed model assembly system in case of finite buffer capacity. *International Journal of Advanced Manufacturing Technology,* 44**,** 345-359.

BAUTISTA, J. & PEREIRA, J. 2007. Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research,* 177**,** 2016-2032.

BAUTISTA, J. & PEREIRA, J. 2011. Procedures for the Time and Space constrained Assembly Line Balancing Problem. *European Journal of Operational Research,* 212**,** 473-481.

BAYBARS, I. 1986a. An Efficient Heuristic Method for the Simple Assembly Line Balancing Problem. *International Journal of Production Research,* 24**,** 149-166.

BAYBARS, I. 1986b. A Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem. *Management Science,* 32**,** 909-932.

BAYKASOGLU, A. 2006. Multi-rule multi-objective simulated annealing algorithm for straight and U type assembly line balancing problems. *Journal of Intelligent Manufacturing,* 17**,** 217-232.

BAYKASOGLU, A. & DERELI, T. 2008. Two-sided assembly line balancing using an ant-colony-based heuristic. *International Journal of Advanced Manufacturing Technology,* 36**,** 582-588.

BAYKASOGLU, A. & DERELI, T. 2009. Simple and U-Type Assembly Line Balancing by Using an Ant Colony Based Algorithm. *Mathematical & Computational Applications,* 14**,** 1-12.

BAYKASOGLU, A., OZBAKIR, L., GORKEMLI, L. & GORKEMLI, B. 2009. Balancing Parallel Assembly Lines via Ant Colony Optimization. *CIE: 2009 International Conference on Computers and Industrial Engineering, Vols 1-3***,** 506-511.

BEARZOTTI, L. A., SALOMONE, E. & CHIOTTI, O. J. 2012. An autonomous multi-agent approach to supply chain event management. *International Journal of Production Economics,* 135**,** 468-478.

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

BECKER, C. & SCHOLL, A. 2009. Balancing assembly lines with variable parallel workplaces: Problem definition and effective solution procedure. *European Journal of Operational Research,* 199**,** 359-374.

BENZER, R., GOKCEN, H., CETINYOKUS, T. & CERCIOGLU, H. 2007. A network model for parallel line balancing Problem. *Mathematical Problems in Engineering,* doi:10.1155/2007/10106.

BERGER, I., BOURJOLLY, J. M. & LAPORTE, G. 1992. Branch-and-Bound Algorithms for the Multiproduct Assembly Line Balancing Problem. *European Journal of Operational Research,* 58**,** 215-222.

BLUM, C. 2005. Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews,* 2**,** 353-373.

BLUM, C., BAUTISTA, J. & PEREIRA, J. 2006. Beam-ACO applied to assembly line balancing. *Ant Colony Optimization and Swarm Intelligence, Proceedings,* 4150**,** 96-107.

BLUM, C., BAUTISTA, J. & PEREIRA, J. 2008. An extended Beam-ACO approach to the time and space constrained simple assembly line balancing problem. *Evolutionary Computation in Combinatorial Optimization, Proceedings,* 4972**,** 85-96.

BOCTOR, F. F. 1995. A Multiple-Rule Heuristic for Assembly-Line Balancing. *Journal of the Operational Research Society,* 46**,** 62-69.

BOWMAN, E. H. 1960. Assembly-Line Balancing by Linear Programming. *Operations Research,* 8**,** 385-389.

BOYSEN, N., FLIEDNER, M. & SCHOLL, A. 2007. A classification of assembly line balancing problems. *European Journal of Operational Research,* 183**,** 674-693.

BOYSEN, N., FLIEDNER, M. & SCHOLL, A. 2008. Assembly line balancing: Which model to use when? *International Journal of Production Economics,* 111**,** 509-528.

BOYSEN, N., FLIEDNER, M. & SCHOLL, A. 2009. Sequencing mixed-model assembly lines: Survey, classification and model critique. *European Journal of Operational Research,* 192**,** 349-373.

BRYTON, B. 1954. *Balancing of a Continuous Production Line.* M.S. Thesis, Northwestern University, Evanston, IL.

CERCIOGLU, H., OZCAN, U., GOKCEN, H. & TOKLU, B. 2009. A Simulated Annealing Approach for Parallel Assembly Line Balancing Problem. *Journal of the Faculty of Engineering and Architecture of Gazi University,* 24**,** 331-341.

CHEHADE, H., YALAOUI, F., AMODEO, L. & DE GUGLIELMO, P. 2008. Ant colony optimization for assembly lines design problem. *Computational Intelligence in Decision and Control,* 1**,** 1135-1140.

CHEN, S. M. & CHIEN, C. Y. 2011. Parallelized genetic ant colony systems for solving the traveling salesman problem. *Expert Systems with Applications,* 38**,** 3873-3883.

CHICA, M., CORDON, O., DAMAS, S. & BAUTISTA, J. 2009. Integration of an EMO-based Preference Elicitation Scheme into a Multi-objective ACO Algorithm for Time and Space Assembly Line Balancing. *Mcdm: 2009 Ieee Symposium on Computational Intelligence in Multi-Criteria Decision-Making***,** 157-162.

CHICA, M., CORDON, O., DAMAS, S. & BAUTISTA, J. 2010. Multiobjective constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: ACO and random greedy search. *Information Sciences,* 180**,** 3465-3487.

CHICA, M., CORDON, O., DAMAS, S. & BAUTISTA, J. 2011. Including different kinds of preferences in a multi-objective ant algorithm for time and space assembly line balancing on different Nissan scenarios. *Expert Systems with Applications,* 38**,** 709-720.

CHOI, G. 2009. A goal programming mixed-model line balancing for processing time and physical workload. *Computers & Industrial Engineering,* 57**,** 395-400.

CHUTIMA, P. & CHIMKLAI, P. 2012. Multi-objective two-sided mixed-model assembly line balancing using particle swarm optimisation with negative knowledge. *Computers & Industrial Engineering,* 62**,** 39-55.

CORDON, O., HERRERA, F. & STÜTZLE, T. 2002. A Review on the Ant Colony Optimization Metaheuristic: Basis, Models and New Trends *Mathware & Soft Computing,* 9**,** 141-175.

COSTA, A., CAPPADONNA, F. A. & FICHERA, S. 2014. Joint optimization of a flow-shop group scheduling with sequence dependent set-up times and skilled workforce assignment. *International Journal of Production Research,* doi: 10.1080/00207543.2014.883469.

DAGANZO, C. F. & BLUMENFELD, D. E. 1994. Assembly system design principles and tradeoffs. *International Journal of Production Research,* 32**,** 669-681.

DAREL, E. M. & COTHER, R. F. 1975. Assembly line sequencing for model mix. *International Journal of Production Research,* 13**,** 463-477.

DENG, G. F. & LIN, W. T. 2011. Ant colony optimization-based algorithm for airline crew scheduling problem. *Expert Systems with Applications,* 38**,** 5787-5793.

DEPUY, G. W. & WHITEHOUSE, G. E. 2000. Applying the COMSOAL computer heuristic to the constrained resource allocation problem. *Computers & Industrial Engineering,* 38**,** 413-422.

DOLGUI, A., EREMEEV, A., KOLOKOLOV, A. & SIGAEV, V. 2002. A Genetic Algorithm for the Allocation of Buffer Storage Capacities in a Production Line with Unreliable Machines. *Journal of Mathematical Modelling and Algorithms,* 1**,** 89-104.

DONG, J. H., XIAO, T. Y., FAN, S. H. & QIANG, L. 2002. Mixed-model assembly line sequencing with hybrid genetic algorithm and simulation. *System Simulation and Scientific Computing (Shanghai), Vols I and II***,** 541-545.

DORIGO, M. & BLUM, C. 2005. Ant colony optimization theory: A survey. *Theoretical Computer Science,* 344**,** 243-278.

DORIGO, M., DI CARO, G. & GAMBARDELLA, L. M. 1999. Ant Algorithms for Discrete Optimization. *Artificial Life,* 5**,** 137–172.

DORIGO, M., MANIEZZO, V. & COLORNI, A. 1996. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics,* 26**,** 29-41.

DORIGO, M. & STUTZLE, T. 2004. *Ant Colony Optimization,* Bradford Books, MIT Press, Cambridge, MA.

EMDE, S., BOYSEN, N. & SCHOLL, A. 2010. Balancing mixed-model assembly lines: a computational evaluation of objectives to smoothen workload. *International Journal of Production Research,* 48**,** 3173-3191.

EREL, E. & SARIN, S. C. 1998. A survey of the assembly line balancing procedures. *Production Planning & Control,* 9**,** 414-434.

ERYURUK, S. H., BASKAK, M. & KALAOGLU, F. 2011. Assembly Line Balancing by Using Statistical Method in Clothing Production. *Tekstil Ve Konfeksiyon,* 21**,** 65-71.

ERYURUK, S. H., KALAOGLU, F. & BASKAK, M. 2008. Assembly line balancing in a clothing company. *Fibres & Textiles in Eastern Europe,* 16**,** 93-98.

FATTAHI, P., ROSHANI, A. & ROSHANI, A. 2011. A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. *International Journal of Advanced Manufacturing Technology,* 53**,** 363-378.

FORD, H. 2009. *My Life and Work - An Autobiography of Henry Ford,* New York, Classic House Books.

GHOSH, S. & GAGNON, R. J. 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *International Journal of Production Research,* 27**,** 637-670.

GOH, W. T. & ZHANG, Z. 2003. An intelligent and adaptive modelling and configuration approach to manufacturing systems control. *Journal of Materials Processing Technology,* 139**,** 103-109.

GÖKÇEN, H., AGPAK, K. & BENZER, R. 2006. Balancing of parallel assembly lines. *International Journal of Production Economics,* 103**,** 600-609.

GÖKÇEN, H., AĞPAK, K., GENCER, C. & KIZILKAYA, E. 2005. A shortest route formulation of simple U-type assembly line balancing problem. *Applied Mathematical Modelling,* 29**,** 373-380.

GUNASEKARAN, A. & CECILLE, P. 1998. Implementation of productivity improvement strategies in a small company. *Technovation,* 18**,** 311-320.

GUTJAHR, A. L. & NEMHAUSER, G. L. 1964. An Algorithm for the Line Balancing Problem. *Management Science,* 11**,** 308-315.

HAMZADAYI, A. & YILDIZ, G. 2012. A genetic algorithm based approach for simultaneously balancing and sequencing of mixed-model U-lines with parallel workstations and zoning constraints. *Computers & Industrial Engineering,* 62**,** 206-215.

HE, N., ZHANG, D. Z. & LI, Q. 2014. Agent-based hierarchical production planning and scheduling in make-to-order manufacturing system. *International Journal of Production Economics*, 149**,** 117-130, doi: 10.1016/j.ijpe.2013.08.022.

HELGESON, W. B. 1954. How to Balance an Assembly Line. *Management Report, No.7.* Carr Press.

HELGESON, W. B. & BIRNIE, D. P. 1961. Assembly Line Balancing Using the Ranked Positional Weight Technique. *The Journal of Industrial Engineering,* 12**,** 394-398.

HOFFMANN, T. R. 1992. Eureka - a Hybrid System for Assembly Line Balancing. *Management Science,* 38**,** 39-47.

HOLLAND, J. H. 1975. *Adaptation in Natural and Artificial Systems*, Cambridge, MIT Press.

HU, X. F., WU, E. F., BAO, J. S. & JIN, Y. 2010. A branch-and-bound algorithm to minimize the line length of a two-sided assembly line. *European Journal of Operational Research,* 206**,** 703-707.

HU, X. F., WU, E. F. & JIN, Y. 2008. A station-oriented enumerative algorithm for two-sided assembly line balancing. *European Journal of Operational Research,* 186**,** 435-440.

HWANG, R. & KATAYAMA, H. 2009. A multi-decision genetic approach for workload balancing of mixed-model U-shaped assembly line systems. *International Journal of Production Research,* 47**,** 3797-3822.

HWANG, R. & KATAYAMA, H. 2010. Integrated procedure of balancing and sequencing for mixed-model assembly lines: a multi-objective evolutionary approach. *International Journal of Production Research,* 48**,** 6417-6441.

ILIE, S. & BADICA, C. 2013. Multi-agent approach to distributed ant colony optimization. *Science of Computer Programming,* 78**,** 762-774.

JAYASWAL, S. & AGARWAL, P. 2014. Balancing U-shaped assembly lines with resource dependent task times: A Simulated Annealing approach. *Journal of Manufacturing Systems,* 33**,** 522-534.

JOHNSON, R. V. 1983. A Branch and Bound Algorithm for Assembly Line Balancing Problems with Formulation Irregularities. *Management Science,* 29**,** 1309-1324.

KARA, Y., GOKCEN, H. & ATASAGUN, Y. 2010. Balancing parallel assembly lines with precise and fuzzy goals. *International Journal of Production Research,* 48**,** 1685-1703.

KARA, Y., OZCAN, U. & PEKER, A. 2007a. An approach for balancing and sequencing mixed-model JIT U-lines. *International Journal of Advanced Manufacturing Technology,* 32**,** 1218-1231.

KARA, Y., OZCAN, U. & PEKER, A. 2007b. Balancing and sequencing mixed-model just-in-time U-lines with multiple objectives. *Applied Mathematics and Computation,* 184**,** 566-588.

KARA, Y., OZGUVEN, C., YALCIN, N. & ATASAGUN, Y. 2011. Balancing straight and U-shaped assembly lines with resource dependent task times. *International Journal of Production Research,* 49**,** 6387-6405.

KARA, Y. & TEKIN, M. 2009. A mixed integer linear programming formulation for optimal balancing of mixed-model U-lines. *International Journal of ProductionResearch,* 47**,** 4201-4233.

KARABATI, S. & SAYIN, S. 2003. Assembly line balancing in a mixed-model sequencing environment with synchronous transfers. *European Journal of Operational Research,* 149**,** 417-429.

KARP, R. M. 1972. Reducibility Among Combinatorial Problems. *In:* MILLER, R. E. & THATCHER, J. W. (eds.) *Complexity of Computer Computations.* New, York: Plenum Press.

KHAW, C. L. E. & PONNAMBALAM, S. G. 2009. Multi-Rule Multi-Objective Ant Colony Optimization for Straight and U-Type Assembly Line Balancing Problem. *2009 Ieee International Conference on Automation Science and Engineering***,** 177-182.

KIM, Y. K., KIM, J. Y. & KIM, Y. 2000a. A coevolutionary algorithm for balancing and sequencing in mixed model assembly lines. *Applied Intelligence,* 13**,** 247-258.

KIM, Y. K., KIM, J. Y. & KIM, Y. 2006. An endosymbiotic evolutionary algorithm for the integration of balancing and sequencing in mixed-model U-lines. *European Journal of Operational Research,* 168**,** 838-852.

KIM, Y. K., KIM, S. J. & KIM, J. Y. 2000b. Balancing and sequencing mixed-model U-lines with a co-evolutionary algorithm. *Production Planning & Control,* 11**,** 754-764.

KIM, Y. K., KIM, Y. H. & KIM, Y. J. 2000c. Two-sided assembly line balancing: a genetic algorithm approach. *Production Planning & Control,* 11**,** 44-53.

KIM, Y. K., SONG, W. S. & KIM, J. H. 2009. A mathematical model and a genetic algorithm for two-sided assembly line balancing. *Computers & Operations Research,* 36**,** 853-865.

KLEIN, R. & SCHOLL, A. 1996. Maximizing the production rate in simple assembly line balancing - A branch and bound procedure. *European Journal of Operational Research,* 91**,** 367-385.

KONG, S. H., NOH, S. D., HAN, Y. G., KIM, G. & LEE, K. I. 2006. An agent-based collaborative assembly process planning system. *International Journal of Advanced Manufacturing Technology,* 28**,** 176-183.

KUCUKKOC, I., KARAOGLAN, A. D. & YAMAN, R. 2013. Using response surface design to determine the optimal parameters of genetic algorithm and a case study. *International Journal of Production Research,* 51**,** 5039-5054, doi: http://dx.doi.org/10.1080/00207543.2013.784411.

KUCUKKOC, I. & ZHANG, D. Z. 2015a. Balancing of parallel U-shaped assembly lines. *Computers and Operations Research,* 64**,** 233–244, doi: http://dx.doi.org/10.1016/j.cor.2015.05.014.

KUCUKKOC, I. & ZHANG, D. Z. 2015b. Coping with Model Variations on Parallel U-shaped Assembly Line Configurations. *IFAC-PapersOnLine,* 48**,** 2030-2035.

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

LEE, Z. J., SU, S. F., CHUANG, C. C. & LIU, K. H. 2008. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Applied Soft Computing,* 8**,** 55-78.

LEUNG, C. W., WONG, T. N., MAK, K. L. & FUNG, R. Y. K. 2010. Integrated process planning and scheduling by an agent-based ant colony optimization. *Computers & Industrial Engineering,* 59**,** 166-180.

LI, N., WANG, S. & LI, Y. 2011a. A Hybrid Approach of GA and ACO for VRP. *Journal of Computational Information Systems,* 7**,** 4939-4946.

LI, Y. C., ZHAO, L. N. & ZHOU, S. J. 2011b. Review of Genetic Algorithm. *Advanced Materials Research,* 179-180**,** 365-367.

LIAO, L. M., HUANG, C. J. & HUANG, J. H. 2012. Applying Multi-agent Approach to Mixed-model Assembly Line Balancing. *Proceedings of the IEEE ICMIT.*

LIAO, L. M., HUANG, C. J. & LIN, C. S. Multi-agent System for Balancing Mixed-model Assembly Lines with Bi-objective. International Conference on Manufacturing Automation, 2010. 191-195.

LIM, M. K. & ZHANG, D. Z. 2004. An integrated agent-based approach for responsive control of manufacturing resources. *Computers & Industrial Engineering,* 46**,** 221-232.

LIM, M. K. & ZHANG, Z. 2003. A multi-agent based manufacturing control strategy for responsive manufacturing. *Journal of Materials Processing Technology,* 139**,** 379-384.

LIM, M. K. & ZHANG, Z. 2012. A multi-agent system using iterative bidding mechanism to enhance manufacturing agility. *Expert Systems with Applications,* 39**,** 8259-8273.

LIU, S. B., NG, K. M. & ONG, H. L. 2008. Branch-and-bound algorithms for simple assembly line balancing problem. *International Journal of Advanced Manufacturing Technology,* 36**,** 169-177.

LUSA, A. 2008. A survey of the literature on the multiple or parallel assembly line balancing problem. *European Journal of Industrial Engineering,* 2**,** 50-72.

MALAKOOTI, B. B. 1994. Assembly-Line Balancing with Buffers by Multiple Criteria Optimization. *International Journal of Production Research,* 32**,** 2159-2178.

MANAVIZADEH, N., HOSSEINI, N. S., RABBANI, M. & JOLAI, F. 2013a. A Simulated Annealing algorithm for a mixed model assembly U-line balancing type-I problem considering human efficiency and Just-In-Time approach. *Computers & Industrial Engineering,* 64**,** 669-685.

MANAVIZADEH, N., RABBANI, M., MOSHTAGHI, D. & JOLAI, F. 2012. Mixed-model assembly line balancing in the make-to-order and stochastic environment using multi-objective evolutionary algorithms. *Expert Systems with Applications,* 39**,** 12026-12031.

MANAVIZADEH, N., TAVAKOLI, L., RABBANI, M. & JOLAI, F. 2013b. A multi-objective mixed-model assembly line sequencing problem in order to minimize total costs in a Make-To-Order environment, considering order priority. *Journal of Manufacturing Systems,* 32**,** 124-137.

MAVROVOUNIOTIS, M. & YANG, S. X. 2013. Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. *Applied Soft Computing,* 13**,** 4023-4037.

MCMULLEN, P. R. & FRAZIER, G. V. 1998. Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations. *International Journal of Production Research,* 36**,** 2717-2741.

MCMULLEN, P. R. & TARASEWICH, P. 2003. Using ant techniques to solve the assembly line balancing problem. *IIE Transactions,* 35**,** 605-617.

MCMULLEN, P. R. & TARASEWICH, P. 2006. Multi-objective assembly line balancing via a modified ant colony optimization technique. *International Journal of Production Research,* 44**,** 27-42.

MENDES, A. R., RAMOS, A. L., SIMARIA, A. S. & VILARINHO, P. M. 2005. Combining heuristic procedures and simulation models for balancing a PC camera assembly line. *Computers & Industrial Engineering,* 49**,** 413-431.

MERENGO, C., NAVA, F. & POZZETTI, A. 1999. Balancing and sequencing manual mixed-model assembly lines. *International Journal of Production Research,* 37**,** 2835-2860.

MES, M., VAN DER HEIJDEN, M. & VAN HARTEN, A. 2007. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. *European Journal of Operational Research,* 181**,** 59-75.

MILTENBURG, G. J. & WIJNGAARD, J. 1994. The U-Line Line Balancing Problem. *Management Science,* 40**,** 1378-1388.

MILTENBURG, J. 1998. Balancing U-lines in a multiple U-line facility. *European Journal of Operational Research,* 109**,** 1-23.

MINITAB. 2015. *Using a confidence interval to decide whether to reject the null hypothesis* [Online]. Available: http://support.minitab.com/en-us/minitab/17/topic-library/basic-statistics-and-graphs/hypothesis-tests/basics/using-a-confidence-interval/ [Accessed 25th May 2015.

MONDEN, Y. 1983. *Toyota Production System,* Norcross, GA, Industrial Engineering and Management Press.

MOSADEGH, H., ZANDIEH, M. & GHOMI, S. M. T. F. 2012. Simultaneous solving of balancing and sequencing problems with station-dependent assembly times for mixed-model assembly lines. *Applied Soft Computing,* 12**,** 1359-1370.

NOURMOHAMMADI, A. & ZANDIEH, M. 2011. Assembly line balancing by a new multi-objective differential evolution algorithm based on TOPSIS. *International Journal of Production Research,* 49**,** 2833-2855.

OZBAKIR, L., BAYKASOGLU, A., GORKEMLI, B. & GORKEMLI, L. 2011. Multiple-colony ant algorithm for parallel assembly line balancing problem. *Applied Soft Computing,* 11**,** 3186-3198.

OZBAKIR, L. & TAPKAN, P. 2010. Balancing fuzzy multi-objective two-sided assembly lines via Bees Algorithm. *Journal of Intelligent & Fuzzy Systems,* 21**,** 317-329.

OZBAKIR, L. & TAPKAN, P. 2011. Bee colony intelligence in zone constrained two-sided assembly line balancing problem. *Expert Systems with Applications,* 38**,** 11947-11957.

OZCAN, U. 2010. Balancing stochastic two-sided assembly lines: A chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm. *European Journal of Operational Research,* 205**,** 81-97.

OZCAN, U., CERCIOGLU, H., GOKCEN, H. & TOKLU, B. 2009. A Tabu Search Algorithm for the Parallel Assembly Line Balancing Problem. *Gazi University Journal of Science,* 22**,** 313-323.

OZCAN, U., CERCIOGLU, H., GOKCEN, H. & TOKLU, B. 2010a. Balancing and sequencing of parallel mixed-model assembly lines. *International Journal of Production Research,* 48**,** 5089-5113.

OZCAN, U., GOKCEN, H. & TOKLU, B. 2010b. Balancing parallel two-sided assembly lines. *International Journal of Production Research,* 48**,** 4767-4784.

OZCAN, U., KELLEGOZ, T. & TOKLU, B. 2011. A genetic algorithm for the stochastic mixed-model U-line balancing and sequencing problem. *International Journal of Production Research,* 49**,** 1605-1626.

OZCAN, U. & TOKLU, B. 2009a. Balancing of mixed-model two-sided assembly lines. *Computers & Industrial Engineering,* 57**,** 217-227.

OZCAN, U. & TOKLU, B. 2009b. Multiple-criteria decision-making in two-sided assembly line balancing: A goal programming and a fuzzy goal programming models. *Computers & Operations Research,* 36**,** 1955-1965.

OZCAN, U. & TOKLU, B. 2009c. A tabu search algorithm for two-sided assembly line balancing. *International Journal of Advanced Manufacturing Technology,* 43**,** 822-829.

OZCAN, U. & TOKLU, B. 2010. Balancing two-sided assembly lines with sequence-dependent setup times. *International Journal of Production Research,* 48**,** 5363-5383.

OZTURK, C., TUNALI, S., HNICH, B. & ORNEK, A. M. 2010. Simultaneous Balancing and Scheduling of Flexible Mixed Model Assembly Lines with Sequence-Dependent Setup Times. *Electronic Notes in Discrete Mathematics,* 36**,** 65-72.

PASTOR, R., ANDRES, C., DURAN, A. & PEREZ, M. 2002. 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**,** 1317-1323.

PEETERS, M. & DEGRAEVE, Z. 2006. A linear programming based lower bound for the simple assembly line balancing problem. *European Journal of Operational Research,* 168**,** 716-731.

PELLICHERO, F. 1999. *Computer-aided choice of assembly methods and selection of equipment in production line design.* PhD Thesis, Universite libre de Bruxelles.

PINTO, P., DANNENBRING, D. G. & KHUMAWALA, B. M. 1975. A branch and bound algorithm for assembly line balancing with paralleling. *International Journal of Production Research,* 13**,** 183-196.

PINTO, P. A., DANNENBRING, D. G. & KHUMAWALA, B. M. 1981. Branch and Bound and Heuristic Procedures for Assembly Line Balancing with Paralleling of Stations. *International Journal of Production Research,* 19**,** 565-576.

PONNAMBALAM, S. G., ARAVINDAN, P. & NAIDU, G. M. 2000. A multi-objective genetic algorithm for solving assembly line balancing problem. *International Journal of Advanced Manufacturing Technology,* 16**,** 341-352.

PRACA, I. C. & RAMOS, C. Multi-agent simulation for balancing of assembly lines. International Symposium on Assembly and Task Planning, 1999 Porto, Portugal. IEEE, 459-464.

PURNOMO, H. D., WEE, H. M. & RAU, H. 2013. Two-sided assembly lines balancing with assignment restrictions. *Mathematical and Computer Modelling,* 57**,** 189-199.

RABBANI, M., MOGHADDAM, M. & MANAVIZADEH, N. 2012. Balancing of mixed-model two-sided assembly lines with multiple U-shaped layout. *International Journal of Advanced Manufacturing Technology,* 59**,** 1191-1210.

RASHID, M. F. F., HUTABARAT, W. & TIWARI, A. 2012. A review on assembly sequence planning and assembly line balancing optimisation using soft computing approaches. *International Journal of Advanced Manufacturing Technology,* 59**,** 335-349.

REKIEK, B. & DELCHAMBRE, A. 2006. *Assembly line design: The balancing of mixed-model hybrid assembly lines with genetic algorithms,* London, Springer.

REKIEK, B., DOLGUI, A., DELCHAMBRE, A. & BRATCU, A. 2002. State of art of optimization methods for assembly line design. *Annual Reviews in Control,* 26**,** 163-174.

RUBINOVITZ, J. & LEVITIN, G. 1995. Genetic algorithm for assembly line balancing. *International Journal of Production Economics,* 41**,** 343-354.

SABUNCUOGLU, I., EREL, E. & ALP, A. 2009. Ant colony optimization for the single model U-type assembly line balancing problem. *International Journal of Production Economics,* 120**,** 287-300.

SALTZMAN, M. J. & BAYBARS, I. 1987. A 2-Process Implicit Enumeration Algorithm for the Simple Assembly Line Balancing Problem. *European Journal of Operational Research,* 32**,** 118-129.

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

SARKER, B. R. & SHANTHIKUMAR, J. G. 1983. A Generalized-Approach for Serial or Parallel Line Balancing. *International Journal of Production Research,* 21**,** 109-133.

SCHOLL, A. & BECKER, C. 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168**,** 666-693.

SCHOLL, A. & BOYSEN, N. 2009. Designing parallel assembly lines with split workplaces: Model and optimization procedure. *International Journal of Production Economics*, 119**,** 90-100.

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

SIMARIA, A. S. 2006. *Assembly Line Balancing - New Perspectives and Procedures.* PhD, Universidade de Aveiro.

SIMARIA, A. S. & VILARINHO, P. M. 2001. The simple assembly line balancing problem with parallel workstations - A simulated annealing approach. *International Journal of Industrial Engineering-Theory Applications and Practice*, 8**,** 230-240.

SIMARIA, A. S. & VILARINHO, P. M. 2009. 2-ANTBAL: An ant colony optimisation algorithm for balancing two-sided assembly lines. *Computers & Industrial Engineering*, 56**,** 489-506.

SPARLING, D. & MILTENBURG, J. 1998. The mixed-model U-line balancing problem. *International Journal of Production Research*, 36**,** 485-501.

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

SRINIVAS, M. & PATNAIK, L. M. 1994. Genetic Algorithms - a Survey. *Computer*, 27**,** 17-26.

STONE, D. & ELLIS, J. 2006. *One and Two-Tailed Tests* [Online]. Available: http://www.chem.utoronto.ca/coursenotes/analsci/stats/12tailed.html [Accessed 21st May 2015.

SU, P. & LU, Y. 2007. Combining genetic algorithm and simulation for the mixed-model assembly line balancing problem. *ICNC 2007: Third International Conference on Natural Computation, Vol 4, Proceedings***,** 314-318.

SUER, G. A. 1998. Designing parallel assembly lines. *Computers & Industrial Engineering*, 35**,** 467-470.

SULAIMAN, M. N. I., CHOO, Y. H. & CHONG, K. E. 2011. Ant Colony Optimization with Look Forward Ant in Solving Assembly Line Balancing Problem. *2011 3rd Conference on Data Mining and Optimization (Dmo)***,** 115-121.

TAHA, R. B., EL-KHARBOTLY, A. K., SADEK, Y. M. & AFIA, N. H. 2011. A Genetic Algorithm for solving two-sided assembly line balancing problems. *Ain Shams Engineering Journal*, 2**,** 227-240.

TALBOT, F. B. & PATTERSON, J. H. 1984. An Integer Programming Algorithm with Network Cuts for Solving the Assembly Line Balancing Problem. *Management Science*, 30**,** 85-99.

TANENBAUM, M. & HOLSTEIN, W. K. 2012. *Mass Production* [Online]. Britannica Academic Edition, Science and Technology. Available: www.britannica.com/EBchecked/topic/368270/mass-production [Accessed June 28 2012].

TASAN, S. O. & TUNALI, S. 2008. A review of the current applications of genetic algorithms in assembly line balancing. *Journal of Intelligent Manufacturing*, 19**,** 49-69.

THEPPHAKORN, T., PONGCHAROEN, P. & HICKS, C. 2013. An ant colony based timetabling tool. *International Journal of Production Economics*, doi: 10.1016/j.ijpe.2013.04.026i.

TIACCI, L. 2012. Event and object oriented simulation to fast evaluate operational objectives of mixed model assembly lines problems. *Simulation Modelling Practice and Theory,* 24**,** 35-48.

TING, C. J. & CHEN, C. H. 2013. A multiple ant colony optimization algorithm for the capacitated location routing problem. *International Journal of Production Economics,* 141**,** 34-44.

TOKSARI, M. D., ISLEYEN, S. K., GUNER, E. & BAYKOC, O. F. 2008. Simple and U-type assembly line balancing problems with a learning effect. *Applied Mathematical Modelling,* 32**,** 2954-2961.

TONGE, F. M. 1960. Summary of a Heuristic Line Balancing Procedure. *Management Science,* 7**,** 21-42.

UGARTE, B. S., PELLERIN, R. & ARTIBA, A. 2011. An improved genetic algorithm approach for on-line optimisation problems. *Production Planning & Control,* 22**,** 742-753.

VENKATA NARASIMHA, K., KIVELEVITCH, E., SHARMA, B. & KUMAR, M. 2013. An ant colony optimization technique for solving min–max Multi-Depot Vehicle Routing Problem. *Swarm and Evolutionary Computation,* 13**,** 63-73.

VILARINHO, P. M. & SIMARIA, A. S. 2002. A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research,* 40**,** 1405-1420.

VILARINHO, P. M. & SIMARIA, A. S. 2006. ANTBAL: an ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research,* 44**,** 291-303.

WEE, T. S. & MAGAZINE, M. J. 1982. Assembly line balancing as generalized bin packing. *Operations Research Letters,* 1**,** 56-58.

WHITE, W. W. 1961. Letter to the Editor—Comments on a Paper by Bowman. *Operations Research,* 9**,** 274-276.

WU, E. F., JIN, Y., BAO, J. S. & HU, X. F. 2008. A branch-and-bound algorithm for two-sided assembly line balancing. *International Journal of Advanced Manufacturing Technology,* 39**,** 1009-1015.

XU, W. & XIAO, T. 2011. Strategic Robust Mixed Model Assembly Line Balancing Based on Scenario Planning. *Tsinghua Science and Technology,* 16**,** 308-314.

YAGMAHAN, B. 2011. Mixed-model assembly line balancing using a multi-objective ant colony optimization approach. *Expert Systems with Applications,* 38**,** 12453-12461.

YAMAN, R. 2008. An assembly line design and construction for a small manufacturing company. *Assembly Automation,* 28**,** 163-172.

YEGUL, M. F., AGPAK, K. & YAVUZ, M. 2010. A New Algorithm for U-Shaped Two-Sided Assembly Line Balancing. *Transactions of the Canadian Society for Mechanical Engineering,* 34**,** 225-241.

YOKOYAMA, K., MORIKAWA, K. & TAKAHASHI, K. 2006. A multi-agent system for assembly line balancing. *ICIM 2006: Proceedings of the Eighth International Conference on Industrial Management***,** 33-39.

YOKOYAMA, K., MORIKAWA, K. & TAKAHASHI, K. A Multi-Agent System for Mixed-Model Assembly Line Balancing. 9th Asia Pasific Industrial Engineering & Management Systems Conference, December 3rd – 5th 2008 INDONESIA. 2597-2605.

YOKOYAMA, K., MORIKAWA, K. & TAKAHASHI, K. A modified multi-agent system for simple assembly line balancing. 8th International Conference of Modeling and Simulation - MOSIM’10, May 10-12 2010 Hammamet - Tunisia.

YU, B., YANG, Z. Z. & YAO, B. Z. 2009. An improved ant colony optimization for vehicle routing problem. *European Journal of Operational Research*, 196**,** 171-176.

ZHANG, W. Q. & GEN, M. 2011. An efficient multiobjective genetic algorithm for mixed-model assembly line balancing problem considering demand ratio-based cycle time. *Journal of Intelligent Manufacturing,* 22**,** 367-378.

ZHANG, X. M., HAN, X. G. & KONG, X. Z. 2008. An Immune Particle Swarm Optimization Algorithm for Mixed Assembly Line Balancing. *Proceedings of the 38th International Conference on Computers and Industrial Engineering, Vols 1-3***,** 1875-1880.

ZHANG, Z. Q., CHENG, W. M., TANG, L. S. & ZHONG, B. 2007. Ant algorithm with summation rules for assembly line balancing problem. *Proceedings of 14th International Conference on Management Science & Engineering,* Vols 1-3**,** 369-374.

ZHANG, Z. W. & SHARIFI, H. 2007. Towards theory building in agile manufacturing strategy - A taxonomical approach. *IEEE Transactions on Engineering Management,* 54**,** 351-370.