# A STUDY OF REAL-TIME ASSEMBLY LINE BALANCING IN THE CONTEXT OF INDUSTRY 4.0

**HUO JIAGE**

**PhD**

**The Hong Kong Polytechnic University**

**2020**

**The Hong Kong Polytechnic University**

**Department of Industrial and Systems Engineering**

# A Study of Real-time Assembly Line Balancing in the Context of Industry 4.0

**HUO Jiage**

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

August 2019

## CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

_____(Signed)

HUO Jiage (Name of student)

# Abstract

Assembly Line Balancing Problem (ALBP) is important to sustain the efficiency of the assembly process. With the development of complex products, the problem size and the operations' complexity in the assembly process are increasing. Consequently, the development of new approaches to suit the complex assembly environment is urgent. Besides, some researchers explored the task assignment plan for ALBP with the assumption that the assembly process is smooth with no disruption. Other researchers considered the impacts of disruptions, but they only explored the task re-assignment solutions for the assembly line re-balancing problem with the assumption that the re-balancing decision has been made already. There is limited literature exploring on-line adjustment solutions for an assembly line in a dynamic environment. This is because real-time monitoring of an assembly process was impossible in the past, and it is difficult to incorporate uncertain factors into the workload balancing process because of the randomness and non-linearity of these factors. However, Industry 4.0 breaks the information barriers between different parts of an assembly line, since smart, connected products, which are enabled by advanced information and communication technology in the context of Industry 4.0, can intelligently interact and communicate with each other and collect, process and produce information. Smart control of an assembly line becomes possible with the large amounts of real-time production data in the era of Industry 4.0, but there is little literature considering this new context.

To solve ALBP efficiently, a hybrid approach which executes the Ant Colony Optimization in combination with Beam Search (ACO-BS) is proposed. The results of 269 benchmark instances show that for 95.54% of the instances, optimal solutions can be found within 360 CPU time seconds. In addition, order strength and time variability are chosen to indicate the complexity of ALBP instances, and the processing times are generated following a unimodal or a bimodal distribution. Then, 27 instances with a total of 400 tasks are generated randomly. The comparison results show that ACO-BS shows advantages in solving the large-scale random instances.

To monitor and control an assembly line in real time, a fuzzy control system composed of two types of fuzzy controllers is developed. Type 1 fuzzy controller is used to determine whether the assembly line should be re-balanced, and type 2 fuzzy controller is used to adjust the production rate of each workstation in time to eliminate blockage and starvation and increase the utilization of machines. Compared with three assembly lines without the proposed fuzzy control system, the assembly line with the fuzzy control system performs better, in terms of blockage ratio, starvation ratio, and buffer level. Furthermore, the above fuzzy control system is developed with the assumption that the processing ability of each operative machine is constant. However, the degradation process of a machine can be divided

into two or more phases, and the switching of the health state will bring significant changes to the processing times of tasks. Therefore, a new fuzzy control system is developed for the context where each machine's health state is generated randomly following a three-state Markov chain. The workload of each workstation is adjusted to match the production ability of the workstation. Compared with the two assembly lines without the fuzzy control system, the assembly line with the proposed fuzzy control system adjusts the assembly plan in time and achieves higher utilization of machines and lower average buffer level, without the expense of production reduction.

In conclusion, the main contributions are concluded in two aspects. To begin with, an efficient algorithm, ACO-BS, is developed to deal with the large-scale ALBP. More importantly, in the context of Industry 4.0, a fuzzy control system is developed to monitor and adjust the assembly process in real time, and the performance of an assembly line with the proposed fuzzy control system is better, in terms of average buffer level and utilization of machines. The research findings shed light on the smart control of the assembly process and provide references for practitioners who are considering the adoption of new technologies involved in Industry 4.0.

# Publications

**Journal papers:**

[1] **Huo, J. G.,** Wang, Z. X., Chan, F. T. S., Lee, C. K. M. & Strandhagen, J. O. (2018), Assembly line balancing based on Beam Ant Colony Optimisation. *Mathematical Problems in Engineering*, 2018, Article ID 2481435, 1-17.

[2] **Huo, J. G.**, Chan, F. T. S., Lee, C. K. M., Strandhagen, J. O. & Niu, B. (2020), Smart control of the assembly process with a fuzzy control system in the context of Industry 4.0. *Advanced Engineering Informatics*, 43, 101031.

[3] **Huo, J. G.**, Zhang J. H. & Chan, F. T. S. (2019), A fuzzy control system for assembly line balancing with a three-state degradation process in the era of Industry 4.0. *International Journal of Production Research*. (under revision)

**Conference papers:**

[1] **Huo, J. G.**, Chan, F. T. S., Chung, S. H., Lee, C. K. M. & Strandhagen, J. O. (2017). Impacts of Industry 4.0 on assembly line balancing, *In International Conference on Advanced Technology Innovation 2017 (ICATI2017)*, 25-28 June, Samui, Thailand.

[2] **Huo, J. G.**, Chan, F. T. S., Lee, C. K. M., Strandhagen, J. O. & Niu, B. (2018). Hybridizing ant colony optimization by beam search for the assembly line balancing problem, *In the 7th International Conference on Mechanics and Industrial Engineering (ICMIE'18)*, 16-18 August, Madrid, Spain.

[3] **Huo, J. G.**, Chan, F. T. S., Lee, C. K. M., & Strandhagen, J. O. (2019) Real-time Fuzzy Control of an Assembly Line, *In International Conference on Advanced Technology Innovation 2019 (ICATI2019)*, 15-18 July, Sapporo, Hokkaido, Japan.

## Acknowledgements

My sincere gratitude goes to my chief supervisor, Professor Felix T.S. Chan, who gives me a lot of support and help. It is his encouragement that drives me to explore more. Meanwhile, I am thankful to my co-supervisors: Dr. Carman K.M. Lee and Professor Jan Ola Strandhagen, who give me many helpful comments and suggestions on my research. Besides, my thanks go to Dr. Nick S.H. Chung, who spends a lot of time discussing with me and gives me many practical suggestions.

I gain knowledge and learn other things, such as how to deal with pressure, how to divide one problem into small problems and how to use sub-objectives to encourage myself. I am grateful for what I have learned, and this study experience will be one of the most precious experiences in my life.

Finally, I must give thanks to my parents and my friends. Without the warmth provided by them, I could not concentrate on my research.

# Table of Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| ACO | Ant Colony Optimization |
| ACO-BS | Ant Colony Optimization hybridized via Beam Search |
| ALBP | Assembly Line Balancing Problem |
| CPS | Cyber-Physical System |
| GALBP | General Assembly Line Balancing Problem |
| IoT | Internet of Things |
| OS | Order Strength |
| RFID | Radio-Frequency Identification |
| RUL | Remaining Useful Life |
| SALBP | Simple Assembly Line Balancing Problem |
| SCP | Smart, Connected Products |
| TI | Time Interval |
| TV | Time Variability |

# Chapter 1. Introduction

In this chapter, the research background and motivation are introduced, and the research scope, objectives and significance are stated. Finally, the structure of the thesis is illustrated with a brief description of each chapter.

## 1.1 Research Background

An assembly line, which is essentially a continuous production line, consists of materials and workstations combined by conveyor belts, contacting workers and machines closely and efficiently (Zhong and Ai, 2017). The Assembly Line Balancing Problem (ALBP) is a classic problem (Salveson, 1955), and can be seen as a generalization of the bin packing problem where precedence constraints are added (Wee and Magazine, 1982). It focuses on assigning tasks to workstations, with the aim of satisfying the precedence relationships among the tasks and the workload limitations of workstations, with the aim of optimizing performance measures (Celik et al., 2014). ALBP can be divided into Simple Assembly Line Balancing Problem (SALBP) and General Assembly Line Balancing Problem (GALBP) (Baybars, 1986). There are four types of SALBP: SALBP-I aims to minimise the number of workstations with a given fixed cycle time; SALBP-II minimises the cycle time with a given number of workstations; SALBP-E aims to minimise the cycle time and the number of workstations at the same time by

considering their relation with the total idle time or the inefficiency of the line; SALBP-F determines the feasibility of the problem with given the number of workstations and the cycle time (Becker and Scholl, 2006).

ALBP is a well-known NP-hard problem, and it has been researched for more than sixty years. It was first studied by Salveson (1955) who constructed a mathematical model of ALBP and suggested a solution procedure. For decades, the core problem has been extended to meet robotic, machining and disassembly contexts, but even the simple version is still challenging (Battaïa and Dolgui, 2013). Exact methods and approximate methods have been used to solve ALBP. According to Baybars (1986), if $n$ denotes the total number of tasks, there are $n!$ possible sequences of tasks in SALBP; if there are $r$ precedence constraints, then there are approximately $\frac{n!}{2r}$ distinct, feasible sequences. Consequently, the required computational time for obtaining an optimal solution with an exact method for most of ALBP increases exponentially with the instance size considered (Battaïa and Dolgui, 2013). This limits the performance of exact methodologies especially when the problem size is extremely large. Therefore, exploring efficient heuristic methodologies to cope with large scale ALBP within an acceptable time period is clearly necessary.

The majority of the literature on ALBP addresses the problem in the traditional context where it is manpower intensive. Consequently, in a traditional assembly line, on-line monitoring of buffer levels, production performance of each workstation and processing times of tasks is unrealistic in practice. However, with the development of new technologies, automated assembly lines are attracting increasingly more attention, especially since the introduction of Industry 4.0, which is the fourth industrial revolution, in 2013.

Industry 4.0 aims to increase operational effectiveness and provide new definitions of business models, services and products (Hermann et al., 2016; Rüßmann et al., 2015). It introduces Internet Technology to make factories more intelligent, improves adaptability, resource efficiency and ergonomics, and integrates customers and business partners into the product definition process and value and logistics chains, respectively (Stork, 2015). According to ElMaraghy and ElMaraghy (2016), Internet of Things (IoT), which is a basic premise of Industry 4.0, refers to a networked interconnection of objects aiming to make all things communicable and enables objects to exchange information on their status and condition. Meanwhile, the advanced information and communication technology (e.g. wireless sensor network, and Cyber-Physical Systems (CPS)) enables products to evolve from the usual products to the Smart, Connected Products (SCP) (Zheng et al., 2019; Porter and Heppelmann, 2014). SCP represents the third wave of IT-driven competition,

with IT embedded in the products, and can collect, process and produce information (Zheng et al., 2018a).

IoT is to bring internet to all kinds of devices and build a connectivity with all the devices. All things connected to the internet can be divided into three groups. The first group includes those that can collect and send information. The second group includes those which can receive and act on information. The last group includes those which can take both of the above actions. Therefore, in the context of Industry 4.0, smart manufacturing resources embedded with IoT technologies (i.e. Radio-Frequency Identification (RFID), barcoding) can interact with each other by sending and receiving information intelligently, and smart machines can send their working status to a central cloud-based "manager" in real time (Zheng et al., 2018b). Smart machines of an assembly line can get access to the production information of upstream workstations and the downstream workstations so that they can collaborate with each other by adjusting the production rates to reduce the work-in-process. Thus, smart devices can get more local and global knowledge and the information is more transparent in the context of Industry 4.0. However, for a traditional assembly line, such communication is not possible, thus, there are barriers of between the information of different workstations and buffers. Thus, Industry 4.0 will provide opportunities for the workload balance of an assembly line by breaking the information barriers. Furthermore, when there are re-optimization commands, the

smart machines can receive and act on the commands accordingly. The re-balancing cost will be reduced largely because of the decrease of the cost caused by worker re-training. Meanwhile, it is easier to implement assembly line re-balancing.

## 1.2 Research Motivation

With the development of products, the problem size is increasing and the complexity in the assembly process is greatly increasing to a large extent. Although many explorations have been undertaken by researchers, the development of methods to suit the complex assembly context is urgent, with the increasing complexity of ALBP. Recently, meta-heuristic algorithms such as the Genetic Algorithm (GA), Particle Swarm Optimisation (PSO) and Ant Colony Optimisation (ACO) have been used to deal with ALBP due to these algorithms' good performance on optimization problems (Zhong and Ai, 2017). ACO has a good performance in solving combinatorial optimisation problems. To effectively address the assembly line balancing problem with complicating factors such as parallel workstations, stochastic task durations and mixed-models, McMullen and Tarasewich (2003) proposed an approach based on ant techniques and, in comparison with other heuristics, showed that the proposed method was competitive with other heuristic methods in terms of the performance measures used in the study. Bautista and Pereira (2007) used an ant algorithm incorporating some ideas that had

offered good results with SALBP to solve the time and space constrained ALBP, and got much better results than those by tabu search. Kucukkoc and Zhang (2015) and Zhong and Ai (2017) also explored ALBP with ACO based approaches. Therefore, ACO based methodologies show promising performance in coping with ALBP, and ACO is sufficiently flexible to be combined with other algorithms to achieve better performance.

Most of the existing literature on ALBP assumes an environment that works smoothly without any disruption (Sancı and Azizoğlu, 2017). However, modifications in the input parameters, such as task adding or moving, changes in precedence relationships, increases and decreases in task times, and changes in the cycle time because of changing demand, necessitate a re-balancing (Gamberini et al., 2006). Thus, an assembly line needs to be re-balanced rather than balanced, in practice (Celik et al., 2014), and assembly line balancing or the re-balancing problem should be considered in a dynamic environment with disruptions. There is some literature that deals with the re-balancing problem (Gamberini et al., 2006; Yang et al., 2013; Li, 2017), however, the problem is still underdeveloped (Battaïa and Dolgui, 2013). The existing research assumes that the re-balancing decision has been made already, and only addresses the task re-assignment problem without discussing how disruptions affect the current assembly plan and how to react to

disruptions. Nevertheless, there is a trade-off between re-balancing the assembly line as soon as possible to reduce production losses and keeping the stability of the assembly line to prevent bigger disruptions to the assembly line. Assembly planning and control are essential for managing the expanding product ranges, reducing delivery time, reducing costs and increasing profitability (Huang et al., 2008). How to make such a trade-off is an important problem faced by practitioners.

Besides, many studies consider ALBP from a static perspective: before the line deployment (Gamberini et al., 2006). It is always assumed that the production rate of one station is constant when the station is operative, that is, the processing ability of the station is constant. Consequently, an assembly line is balanced with the assumption that all workstations are always identical. However, the degradation process of a machine can be divided into two or more health phases (Peng et al., 2019), and significantly different processing abilities will be shown at different phases. Even if the processing abilities of workstations are identical at the beginning when all machines are in the same health state, it is impossible that the transitions of health states of independent machines are always synchronous. Thus, asynchronous health states of machines can bring a larger deviation in the operation times of workstations and affect the initial workload balance negatively. At some point, re-balancing will be inevitable (Makssoud et al., 2015). Nevertheless, there is limited literature on when to re-balance an assembly line, especially considering the

disruptions brought by transitions of health states of machines. This is because the trigger point for re-balancing can only be determined when the assembly process is monitored in real time, which is not easy to be realized with the technologies in the past. In addition, it is difficult to fuse the real-time information from different sources to a final decision.

However, with the development of technologies, Industry 4.0 will break the information barriers between the different parts of an assembly line by enabling the communications of all the smart devices included in an assembly line. Also, the re-balancing cost will be reduced largely and the assembly line re-balancing can be implemented more easily. Different parts of an assembly line can communicate with each other and can get access to real-time information easily. Smart assembly lines become possible with real-time information obtained by advanced information and network technologies in the era of Industry 4.0. Industry 4.0 will bring new attributes and opportunities to an assembly line, and will create a novel context with real-time information on the assembly line. Nevertheless, there is sparse literature on on-line planning and control of an assembly line, taking the novel context into account. Furthermore, how to efficiently use real-time data to make advanced decisions in a smart factory is an urgent problem to be solved (Feng et al., 2018).

Accurate analysis of an automotive assembly line will be difficult because of the randomness and nonlinearity caused by unpredictable machine failures, asynchronousness among various sections in the assembly line, the coupling of sections through finite buffers, and coupling between the production and material handling system (Chang et al., 2013). Additionally, it is challenging to design a control system for a nonlinear system with unexpected events (Liu et al., 2017), and a mathematical model of such a control system is difficult to obtain. However, fuzzy controllers can provide a systematic and efficient framework for incorporating data obtained by sensors and human judgments, and it is always possible to design a fuzzy controller that is suitable for the nonlinear system under control by carefully choosing the parameters (Wang, 1993). Although there are some explorations on ALBP with fuzzy theory (Zacharia and Nearchou, 2012; Cheshmehgaz et al., 2012; Simona, 2015), the fuzzy theory is always used to deal with uncertain processing times, multiple goals, or improve the method to solve ALBP. Besides, there are some explorations of production control with fuzzy controllers, but only production rates are monitored and controlled by fuzzy controllers, without the discussion of assembly line re-balancing (e.g. Hui et al., 2002; Tamani et al., 2011). Different from the traditional production system, assembly lines in Industry 4.0 will be more re-configurable and be re-balanced more frequently, thus, the assembly process to be controlled should be treated from a dynamic perspective.

## 1.3 Research Scope and Objectives

This research focuses on the exploration of real-time assembly line balancing in the environment of Industry 4.0. There are many advantages brought by Industry 4.0 to the assembly process, however, the exploration in this study focuses on the more transparent information brought by Industry 4.0, and the intelligent interactions between workstations are considered. A decision support system is developed to adjust the assembly line in time with the real-time information of the assembly process. The main objectives of this study are stated as follows:

(1) To develop an efficient algorithm to solve the large-scale SALBP so that satisfactory results can be obtained within an acceptable computation time.

(2) To develop a fuzzy control system, by which real-time information of the assembly process can be processed and transferred into assembly plan adjustments, including assembly line re-balancing and adjustments of production rates, to control the assembly line in real time and improve the collaboration between workstations.

(3) To realize real-time workload balance of the assembly line with the real-time information of the degradation process of machines, so that the workloads of

different workstations can be balanced with the consideration of the changing processing abilities of machines.

## 1.4 Research Significance

The significance of this research is illustrated as follows:

(1) This study explores the real-time balance control of an assembly line in a novel context created by Industry 4.0, with the higher information transparency level considered. The research findings will shed light on the smart control of the assembly process and contribute to the smart manufacturing theory.

(2) The research findings also provide references for practitioners who are considering the adoption of new technologies involved in Industry 4.0 and those who are exploring on-line methods to deal with uncertainties, eliminate starvation and blockage, and maintain a low buffer level of the assembly line without the expense of production reduction.

## 1.5 Organization of the Thesis

The research background, motivation, scope, objectives and significance are stated in this chapter. The rest of the thesis is organized as follows.

Chapter 2 presents works related to Industry 4.0, ALBP, the application of ACO on ALBP, assembly line re-balancing and the fuzzy logic system. Finally, the research gaps are concluded accordingly.

Chapter 3 introduces an algorithm based on beam ant colony optimisation to solve SALBP, and the effectiveness of the proposed algorithm is examined with the benchmark instances and large-scale instances which are generated randomly.

Chapter 4 presents a fuzzy control system used to monitor and control the assembly process in real time. There are two types of fuzzy controllers. Type 1 fuzzy controller is used to determine when to re-balance an assembly line, while type 2 fuzzy controller is used to support the decision on how to adjust the production rate of each workstation. The performance of the assembly line with the fuzzy control system is compared with the performance of three different assembly lines without the fuzzy control system.

Chapter 5 shows a further exploration of real-time fuzzy control system, with the degradation process of machines considered. The degradation process is divided into the normal stage, minor defective stage and severe defective stage, and characteristics of each state are defined. The effect of the proposed fuzzy control system on the assembly process is demonstrated by the comparisons of the performance of the assembly line with the fuzzy control system and the performance of the assembly lines without the fuzzy control system.

Chapter 6 presents the main contributions and conclusions drawn from the research findings. Limitations of this research and future work are also stated.

# Chapter 2. Literature Review

Works related to the new opportunities brought by Industry 4.0, ALBP, the application of ACO on ALBP, characteristics of assembly line re-balancing, and the application of fuzzy theory on solving ALBP and the application of the fuzzy logic system on decision-making are presented in this chapter. Besides, the research gaps are also discussed.

## 2.1 Industry 4.0

Industry 4.0, as an industrial revolution, will reshape the ways things are made. Optimized cells will be integrated, automated and optimized to improve the efficiency and change the relationships among suppliers, producers and customers, and redefine the relationship between humans and machines (Rüßmann et al., 2015). Apart from providing great opportunities to reshape the future, Industry 4.0 aims to increase the operational effectiveness and new definitions of business models, services and products (Hermann et al., 2016; Rüßmann et al., 2015). With the latest advanced technologies, smart factory in the context of Industry 4.0 is becoming a new manufacturing pattern (Lee et al., 2017a).

According to Jazdi (2014), we are experiencing Industry 4.0 in terms of CPS. With

cyber technology, automated systems and equipment, internal logistics systems and operating supplies are connected, which enables direct access to the higher-level processes and services, optimal resource utilization and smart control (Jazdi, 2014).

The CPS connected to the Internet is often referred to as the IoT, which is an information network that consists of physical objects which allows interaction and cooperation to reach common goals (Atzori et al., 2010). IoT is the basic premise for the implementation of Industry 4.0 (Wan et al., 2016). It provides a version for the future Internet where physical things (such as RFID tags, sensors, actuators and mobile phones) are connected (Bandyopadhyay and Sen, 2011). What is more, its basic idea is the pervasive presence of large amounts and kinds of things or objects, allowing it to gain ground in the scenario of modern wireless telecommunications (Atzori et al., 2010). It gives access to information about the physical world and promotes innovative services to increase efficiency and productivity (Bandyopadhyay and Sen, 2011).

With the real-time data obtained by IoT, big data analysis can be used to make logistic decisions (Zhong et al., 2017), and smart production (Lee et al., 2017b), smart logistics (Lee et al., 2018a) and smart cities (Keung et al., 2018) are possible. Meanwhile, advanced technologies enable better implementation of automated guided vehicles (Lee et al., 2018b; Lu et al., 2018), which supports smart warehouse.

Additionally, big data analysis can be used to improve the speed and accuracy in maintenance decision making (Lee et al., 2016).

Wan et al. (2016) pointed out some possibilities in a smart factory: For the traditional networks, the communication protocols of all related devices should be updated, if a new mechanism of cooperation is necessary. Large amounts of time and financial investment will be needed. However, with IoT, all the related data can be transmitted to the cloud and neighboring nodes for management and optimization, during the process of production. Consequently, monitoring and controlling the production process become possible, and high quality and efficiency ensue.

## 2.2 ALBP

A detailed analysis of ALBP in different industrial contexts can be found in a survey presented by Battaïa and Dolgui (2013), who analyzed about 300 studies on balancing flow lines within many different industrial contexts to classify and compare the means for input data modeling, constraints and objective functions used. Despite the enormous academic effort in ALBP, there is still a lack of communication between researchers and practitioners, and a considerable gap between requirements of real configuration problems and the status of research. Boysen et al. (2007) provided a classification scheme for ALBP to ease

communication between researchers and practitioners. Scholl and Becker (2006) focused on SALBP and provided a comprehensive survey of SALBP on recent contributions to this field, while Becker and Scholl (2006) explored the developments in GALBP. There are many advances and developments of ALBP. Boysen et al. (2008) structured the vast field of ALBP based on settings which would reflect real-world problems, and suggestions on how to single out the balancing procedures for practitioners were provided.

ALBP is a well-known NP-hard decision problem about assigning a finite set of tasks to workstations optimally with the restriction of given precedence relationships (Boysen et al., 2007). Salveson (1955) was the first researcher who constructed a mathematical model of the ALBP and suggested a solution procedure. For several decades, the core problem originally introduced for manual assembly has been extended to suit robotic, machining and disassembly contexts, but even the simpler version of this problem, that is, the simple assembly line balancing problem, is still a challenging topic for researchers (Battaïa and Dolgui, 2013).

During the last few decades, the mathematical formulation of ALBP was enriched and intensively studied from various points of view. The simpler version of this problem, SALBP, was first defined by Baybars (1986). Other kinds of assembly lines have also been studied. For example, Yang et al. (2013) proposed a

multi-objective genetic algorithm to address the re-balancing problem for a mixed-model assembly line with seasonal demands; Zha and Yu (2014) formalized U-line re-balancing problem with respect to minimization the moving cost of machines and labor cost, and used a new hybrid algorithm of ant colony optimization and filtered beam search to solve the problem; Celik et al. (2014) defined the U-line re-balancing problem with stochastic task times and proposed a solution procedure based on ant colony optimization. The objective of the proposed algorithm was to minimize the total cost of re-balancing which is the sum of task transposition costs, workstation opening/closing costs and operating costs of workstations.

Both exact methods and approximate methods have been used to solve the ALBP. The required computational time for obtaining an optimal solution with an exact method for most line balancing problems increases exponentially with the size of the instances considered so that even SALBP is NP-hard (Battaïa and Dolgui, 2013). Consequently, approximate methods are needed when coping with large scale cases. Besides, simulation can be a useful tool to evaluate the dynamic uncertainties of assembly lines. In the work on robust assembly line balancing conducted by HazıR and Dolgui (2013), ALBP under uncertainty was considered and two robust optimization models were developed, with a decomposition-based exact algorithm developed and combined with enhancement strategies to solve optimally large-scale

instances. Ege et al. (2009) proposed two branch and bound algorithms to study the NP-hard problem of assembly line balancing with station paralleling to find an assignment of tasks to various stages to minimize the sum of station opening and tooling/equipment costs.

Scheduling of the execution of tasks assigned to every workstation following the balancing of the assembly line has been scarcely reported in the literature, and researchers always assume that tasks could be executed in an arbitrary precedence-feasible sequence within each station without changing the operation time of each station. However, there are researchers that hold the view that setups between tasks exist and optimal or near-optimal tasks schedules should be provided inside each workstation. For example, Andres et al. (2008) added sequence-dependent setup time considerations to the classical SALBP, with assumptions that task processing times, setup times matrix and precedence relationships were known deterministically, and processing and setup times were independent of the workstation where the tasks are processed. Scholl et al. (2013) believed that the task sequence influenced the station time and sequence-dependent setups, such as walking distances and tool changes, had to be considered. They modified the setup assembly line balancing and scheduling problem by modeling setups more realistically, and developed a new, more compact mathematical model formulation.

## 2.3 ACO and ALBP

Swarm intelligence algorithms are based on the collective behavior in decentralized, self-organized systems, and consist of agents interacting with each other and the environment. There is no centralized control structure. This kind of algorithms can be scalable since the number of agents can be easily added or removed. Besides, each agent is simple to design, and reliance on individual agents is small. Although each agent is not sophisticated, complex tasks can be solved in cooperation. As to ant colony algorithm, its main novel idea is the synergistic use of cooperation among many relatively simple agents which communicate by distributed memory implemented as pheromone deposited on edges of a graph (Dorigo and Gambardella, 1997). The colony coordinates the activities without direct communication between individual ants, as an isolated ant basically moves at random (Simaria and Vilarinho, 2009). Each ant can build a solution step by step, and information left by other ants are used during the solution generation process.

The ant colony algorithm has been applied to solve ALBP, and the traditional ant colony algorithms have been adapted to deal with the complex models of ALBP. Furthermore, researchers have also validated the effectiveness of the ant colony heuristic in solving ALBP. Bautista and Pereira (2002) solved the ALBP

(minimizing the number of workstations with a given a fixed cycle time) efficiently with an ACO metaheuristic method, which was adapted to solve a real case problem that was found in a bicycle assembly line. Baykasoğlu and Dereli (2009) integrated the computer method of sequencing operations for assembly lines, ranked the positional weight heuristic and the ant colony heuristic to deal with the simple and U-shaped ALBPs. Additionally, Fattahi et al. (2011) developed a heuristic approach based on the ant colony optimisation approach to solve the medium- and large-size scales of this problem, since the problem is NP-hard. The experimental results validate the effectiveness and efficiency of the proposed algorithm.

SALBP which belongs to a class of intensively studied combinatorial optimisation problems known to be NP-hard, has attracted the attention of researchers and practitioners of operations research for almost half a century (Scholl and Becker, 2006). With the development of ALBP, the core problem has been extended from a manual assembly background to robotic, machining and disassembly contexts, thus there are various industrial environments and line configurations (Battaïa and Dolgui, 2013). To effectively address the assembly line balancing problem with complicating factors such as parallel workstations, stochastic task durations and mixed-models, McMullen and Tarasewich (2003) proposed an approach based on ant techniques, and comparison with other heuristics showed that the proposed method was competitive with other heuristic methods. Simaria and Vilarinho (2009)

presented a method to solve the two-sided mixed-model ALBP with an ant colony optimisation algorithm, where two ants were used simultaneously to build a balancing solution which verified the precedence, zoning, capacity, side and synchronism constraints. The superior performance of the approach was demonstrated by the results of a computational experiment. AkpıNar et al. (2013) presented a hybrid algorithm combining an ant colony algorithm with a genetic algorithm for type I mixed model ALBP with features such as parallel workstations, zoning constraints and sequence-dependent setup times between tasks. To carry out assembly sequence planning and assembly line balancing simultaneously, Lu and Yang (2016) proposed an ant colony algorithm based on the searching mechanism and the pheromone updating mechanism.

There are many situations in which multiple objectives are considered and these objectives are sometimes conflicting. Therefore, methods to solve multi-objective ALBP are valuable to guide practitioners. McMullen and Tarasewich (2006) simultaneously addressed the objectives of crew size, system utilization, the probability of jobs being completed within a certain time frame and system design costs, and the superiority of the modified ant colony optimisation technique was shown in comparative results. Özcan and Toklu (2009) proposed a pre-emptive goal programming model for precise goals and a fuzzy goal programming model for imprecise goals for two-sided assembly line balancing. They proposed the first

multiple-criteria decision-making approach for two-sided ALBP with multiple objectives (i.e. minimize the number of mated-stations, cycle time and the number of tasks assigned per station), and the flexibility and the efficiency of the proposed goal programming models were well illustrated. Chica et al. (2010) presented two new multi-objective proposals based on ant colony optimization and random greedy search algorithms to solve the time and space assembly line balancing problem, and promising results were obtained. To study the influence of incorporating user preferences based on Nissan automotive domain knowledge to guide the multi-objective search process, Chica et al. (2011) proposed a multi-objective ant colony optimization algorithm to solve the time and space assembly line balancing problem. They obtained the most useful solutions for the decision-makers in six different Nissan scenarios around the world, using the real data of the Nissan Pathfinder engine. Yagmahan (2011) dealt with the mixed-model assembly line balancing problem, using a multi-objective ant colony optimization algorithm, and the results of a few test problems showed that the proposed algorithm was more efficient and effective than the other methods compared in this study. Rada-Vilela et al. (2013) adapted eight different multi-objective ant colony optimization algorithms and compared their performance on ten well-known problem instances, and the algorithms were ranked according to three multi-objective indicators and the differences between the top-4 were further reviewed using statistical significance tests. Zha and Yu (2014) presented a new hybrid algorithm of ant colony

optimisation and filtered beam search to solve the U-line rebalancing problem with two objectives. In the process of constructing a path, each ant explored several nodes for one step and chose the best one by global and local evaluation at a given probability. The proposed algorithm was shown to be good at solving the U-line rebalancing problem. Kucukkoc and Zhang (2015) introduced a type-E parallel two-sided ALBP, and proposed a new ant colony optimisation method with optimised parameters for solving the problem and found promising ways to simultaneously minimise two conflicting objectives: cycle time and number of workstations.

Some researchers used one colony of ants to update the pheromone values and guide the searching process, while other researchers used multiple colonies of ants in the searching process to make the searching process more efficient. For example, Kucukkoc and Zhang (2016) proposed a mixed-model parallel two-sided assembly line system that could be utilized to produce large-sized items in an inter-mixed sequence, and developed a flexible agent-based ant colony optimization algorithm to solve the problem, with dynamically changing workloads of workstations (based on specific product models during the production process) explored. Multiple ants can be applied to the searching process in multiple objective problems. Agrawal and Tiwari (2008) utilized collaborative ACO that bilateral colonies of ants independently identified two sequences and information obtained by their

collaboration was utilized to guide the future path in solving a balancing problem in mixed-model disassembly. Ozbakir et al. (2011) studied parallel assembly lines with a novel multiple-colony ant algorithm, and the effective algorithm was examined with benchmark instances and compared with other algorithms.

## 2.4 Assembly Line Re-balancing

Assembly Line Balancing (ALB) is important for overall efficiency. Most of the ALB literature assumes an environment that works smoothly without any disruptions, however, manufacturing environments are often prone to disruptions (Sancı and Azizoğlu, 2017). A wide variety of modifications in the input parameters, such as task adding or moving, changes in precedence relationships, increases and decreases in task times, and changes in the cycle time because of the changing demand, necessitate a re-balancing (Gamberini et al., 2006). Robust line balancing solutions can retain the initial task assignment to some extent without modifications of the line, however, at some point, re-balancing of the line becomes inevitable (Makssoud et al., 2015). Not surprisingly, Celik et al. (2014) claimed that an assembly line needs to be re-balanced rather than balanced.

The re-balancing problems are quite different from the balancing problems since the existing configuration must be considered, thus, the methods and solutions

developed for line balancing problems cannot be directly used for re-balancing problems (Makssoud et al., 2015). There are some researchers who have explored the re-balancing problems. Based on "Technique for Order Preference by Similarity to Ideal Solution", which is an integration of a multi-attribute decision-making procedure, Gamberini et al. (2006) dealt with the assembly line re-balancing problem by considering minimizing the unit labor and expected unit incompletion costs and tasks re-assignment. Yang et al. (2013) proposed a multi-objective genetic algorithm to address the re-balancing problem for a mixed-model assembly line with seasonal demands. Celik et al. (2014) defined a U-line re-balancing problem with stochastic task times, and proposed a method based on ant colony optimization. Motivated by task improvements during the production process along an assembly line, Li (2017) used an algorithm named ENCORE to solve the problem in the context of automatic assembly line systems. Sancı and Azizoğlu (2017) considered the re-balancing problem in which tasks at least on the disrupted workstations should be reassigned to other workstations.

For the re-balancing problem, a quick resolution is more important than an optimal solution, and the aim is to react quickly to reduce negative impacts of any disturbing events and define a new solution that is close to the initial line balancing (Antoine et al., 2016). When dealing with the re-balancing problem, Celik et al. (2014) proposed an algorithm to minimize the total cost of re-balancing which is the sum of task

transposition costs, workstation opening/closing costs and operating costs of workstations for a particular planning horizon. Sancı and Azizoğlu (2017) made a trade-off between the efficiency of the new balance and the stability, indicated by the differences between the initial and the new task assignments.

The assembly line re-balancing problem is still underdeveloped (Battaïa and Dolgui, 2013). Because of the uncertainties, exact parameters of the production process are difficult to obtain before the process begins (Hu et al., 2016). Thus, on-line solutions are needed to deal with uncertainties effectively.

Industry 4.0 will encompass numerous technologies and associated paradigms, and a few of these emerging paradigms include IoT, cloud-based manufacturing, and social product development (Thames and Schaefer, 2016). Successful applications of IoT have been demonstrated in the retail business, logistics, military, environment surveillance, and healthcare, and in those applications, real-time data can be collected by numerous sensors and the data can be shared by the network to support decision-making (Wang et al., 2014). Thus, differing from the traditional production system, assembly lines in the era of Industry 4.0 will be more re-configurable, and the information transparency level will be improved significantly. Assembly line re-balancing problem needs to be explored under this novel context, with the real-time information of the assembly process considered.

## 2.5 Fuzzy Logic System

Researchers have used fuzzy logic when dealing with ALBP. Some use fuzzy logic to define the processing time of one task. For example, Zacharia and Nearchou (2012) presented a fuzzy extension of the type 2 ALBP with fuzzy job processing times, and the processing times were formulated by triangular fuzzy membership functions. Some researchers use fuzzy theory to deal with multiple goals and heuristic algorithms improvements in ALBP. For example, fuzzy goal programming was used, and an appropriate genetic algorithm was developed by Cheshmehgaz et al. (2012), to consider three criteria during the balancing: cycle time, overall workload and assembly worker postures. To solve a multi-objective ALBP, Simona (2015) utilized a fuzzy controller for tuning inertia weight in particle swarm optimization.

There are few studies exploring the application of fuzzy controllers in workload balancing control of assembly lines, and real-time production rate adjustment in each workstation to decrease inventory and improve the overall production rate when there are uncertainties. To make balance control of the sewing operations on assembly lines, Hui et al. (2002) used a fuzzy system to determine the number of operators to be moved in and out of a sewing section. This is an important

exploration in that a fuzzy system is utilized to deal with the balance control of a manufacturing process. However, they restricted the balance control problem to apparel manufacturing and assumed that the machine downtime due to failure was insignificant. Thus, they did not consider possible disruptions in the manufacturing process, and did not discuss when disruptions can lead to assembly plan modifications.

However, fuzzy controllers are always applied in decision making. Tsourveloudis et al. (2000) developed a line, assembly, and disassembly controller to adjust the processing rate of each production stage so that the workflow is balanced, and the extreme events of machine starving or blocking are reduced, and simulation results showed that the proposed approach outranks other control policies in keeping the work-in-progress inventory low. Nakandala et al. (2013) proposed a fuzzy-based decision support model for determining the chance of meeting on-time delivery in a complex supply chain environment. Fuzzy logic principles and a unitary structure-based supply chain model were integrated, and uncertainties associated with key inputs of on-time delivery performance for effective decision-making process were addressed, to minimize of business losses that result from penalties and customer dissatisfaction and the consequently reduced market share. Al-Ebbini et al. (2016) presented a fuzzy lung allocation system to determine which potential recipients would receive a lung for transplantation in order to deal with the

vagueness and fuzziness of the decision making of the medical experts, and the proposed decision process provided a more effective, time-efficient, and systematic decision support tool.

## 2.6 Research Gaps

The research gaps are stated as follows:

(1) There are many explorations related to ACO and ALBP, however, the exploration of approaches that can solve ALBP for complex products within an acceptable time is critical in real industrial applications, since ALBP is an NP-hard problem and the ALBP of complex products brings new challenges. More advances in methods to solve large scale ALBP with high complexity are necessary to suit the dynamic and changing industrial environment.

(2) In the era of Industry 4.0, there will be vast changes in the assembly process. Different parts of an assembly line can communicate with each other, and with more easily accessible real-time information, it is expected to realize better collaboration between different parts. Smart assembly systems are needed to achieve more autonomy in communication between entities in the system and more adaptable control of assembly flow and better performance (ElMaraghy

and ElMaraghy, 2016). However, there are few studies dealing with workload balance control with the benefits brought by Industry 4.0 considered.

(3) The re-assignment solutions are searched for with the assumption that the re-balancing decision has been made, however, there is sparse literature on when to re-balance the assembly line, although this problem is faced by practitioners. Actually, there are many factors resulting in re-balancing, but monitoring all these factors and combining them together to make a decision are difficult.

(4) Although there are some explorations on ALBP with fuzzy theory, there are few publications considering the problem in the environment of Industry 4.0. There are researchers who utilized the fuzzy control system to balance the workloads of workstations by adjusting the production rates, but they did not discuss when to re-balance the assembly line.

Therefore, there is little literature that addresses ALBP, considering the novel context brought by Industry 4.0 where real-time information is accessible to workstations, and commands can be easily sent to machines to adjust their production rates to react to disruptions and achieve a better collaboration of workstations. Differing from the traditional production system, assembly lines in the

era of Industry 4.0 will be more re-configurable and be re-balanced more frequently, thus, the assembly process to be controlled should be treated from a dynamic perspective. To fill the above research gaps, an efficient algorithm to solve the large scale ALBP is developed, and a fuzzy control system is developed to deal with the disruptions to an assembly line and to adjust the assembly line to achieve better performance.

## 2.7 Summary

This chapter presents works related to Industry 4.0, ALBP and ACO, and highlights the difference between assembly line balancing and re-balancing. The utilization of fuzzy theory is also stated. The research gaps, which are important to show the motivation of this research, are clearly presented after reviewing the related works.

# Chapter 3. ALBP Based on Beam Ant Colony Optimisation

In this chapter, the description and the mathematical model of ALBP are presented, and an algorithm based on beam ant colony optimisation is proposed to solve ALBP. The effectiveness of the algorithm is examined on the benchmark instances and the large-scale instances which are generated randomly.

## 3.1 Assembly Line Balancing Problem

### 3.1.1 Problem Description

ALBP focuses on assigning elementary tasks, which are necessary to assemble or disassemble a product, to a set of workstations or modules that compose the line, consistently and efficiently (Bautista et al., 2016). For SALBP, the cumulative constraints associated with the available work time at workstations and the precedence constraints established by the order in which the tasks must be executed need to be considered (Bautista et al., 2016). Nevertheless, GALBP problems contain additional considerations, such as the restricted assignment of tasks (Scholl et al., 2010), or the assignment in a block of certain tasks (Battaïa and Dolgui, 2012). Large scale SALBP-I is considered in this study.

Figure 3-1 shows the structure of an assembly line. There are $M$ workstations,

which are associated with groups of workers and/or robots (Bautista et al., 2016). They are arranged one behind another and are connected by a transport system, which determines the speed of the movement of the work-in-progress. Consequently, the assigned workload to each workstation should be completed within a constant time, which is the cycle time and the workload limit of each workstation.

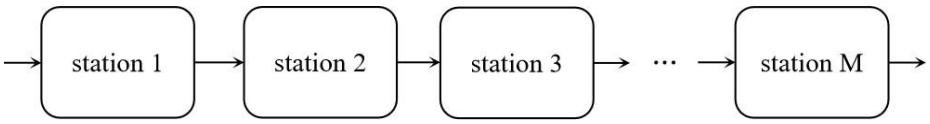


**Figure 3-1:** Structure of an assembly line

For example, Figure 3-2 is a precedence graph of the assembly process of a single product. The graph shows the precedence relationships between tasks and the corresponding processing times of tasks. $i$ in a circle denotes task $i$, and $t_i$ outside the circle denotes the processing time of task $i$.

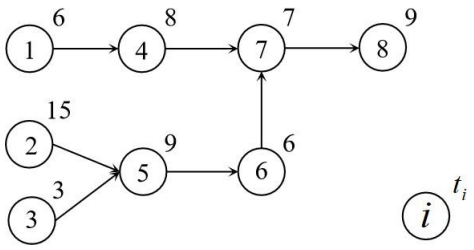


**Figure 3-2:** A precedence graph

For each workstation, a task is available to be assigned to the workstation, when all

its predecessors are assigned to workstations already and its processing time is not larger than the left time in the workstation. If the cycle time is set to be 15, based on the precedence graph in Figure 3-2, tasks can be assigned to five workstations with the following task groups: {1, 4}, {2}, {3, 5}, {6, 7}, {8}. Thus, based on the given precedence graph and cycle time, SALBP-I is to explore the optimal solution with the least number of workstations.

According to Baybars (1986), five main assumptions in SALBP are stated as follows:

(1) A task cannot be split among two or more stations, and all tasks must be processed.

(2) Tasks cannot be processed in arbitrary sequences due to technological precedence requirements.

(3) All stations under consideration are equipped and manned to process any task, and any task can be processed at any station.

(4) The task times are independent of the station at which they are performed and of the preceding or following tasks.

(5) The assembly system is assumed to be designed for a unique model of a single product.

### 3.1.2 Mathematical Model

Notations:

$n$ : total number of tasks;

$UB$ : upper bound of the total number of workstations;

$t_i$ : processing time of task $i$ ;

$LB = \left\lceil \sum_{i=1}^{n} t_i \Big/ C \right\rceil$ : lower bound of the total number of workstations;

$C$ : cycle time;

$P$ : set of pairs of tasks $(i,k)$ such that $i$ immediately precedes $k$ ;

$P_i(S_i)$ : set of tasks that precede (succeed) task $i$ ;

$E_i = \left\lceil \left( t_i + \sum_{k \in P_i} t_k \right) \Big/ C \right\rceil$ : lower bound on the number of the workstation to which task $i$ can be assigned;

$L_i = UB + 1 - \left\lceil \left( t_i + \sum_{k \in S_i} t_k \right) \Big/ C \right\rceil$ : upper bound on the number of the workstation to which task $i$ can be assigned;

Decision variables:

$x_{ij} \in \{0,1\}$ : the decision variable equals to 1, if and only if task $i$ is assigned to workstation $j$ ; otherwise, the variable equals to 0;

$y_j \in \{0,1\}$ : the decision variable equals to 1, if and only if any task is assigned to workstation $j$ ; otherwise, the variable equals to 0.

The mathematical model of SALBP-I is presented as follows:

$$\min z = \sum_{j=LB+1}^{UB} j \cdot y_j \tag{3-1}$$

$$\sum_{j=E_i}^{L_i} x_{ij} = 1 \quad i = 1,2,...,n \tag{3-2}$$

$$\sum_{i=1}^{n} t_i \cdot x_{ij} \le C \quad j = 1,2,...,LB \tag{3-3}$$

$$\sum_{i=1}^{n} t_i \cdot x_{ij} \le C \cdot y_j \quad j = LB+1,...,UB \tag{3-4}$$

$$\sum_{j=E_i}^{L_i} j \cdot x_{ij} \le \sum_{j=E_k}^{L_k} j \cdot x_{kj} \quad \forall (i,k) \in P \tag{3-5}$$

The objective function (3-1) minimises the total number of workstations; constraint (3-2) suggests that every task is assigned to one and only one workstation; workload constraints (3-3) and (3-4) imply that the total processing time of each workstation does not exceed the cycle time; constraint (3-5) ensures that all the precedence relations are satisfied.

### 3.1.3 Reversibility of ALBP

One ALBP instance can transfer to its reverse version after all the precedence relationships are reversed. If $S_{rev} = \{S_1,\ldots,S_m\}$ is a solution for the reverse problem, then a solution for the original problem can be obtained by inverting the workstation orders of $S_{rev}$. Thus, a solution to the original problem can be $S = \{S_m,\ldots,S_1\}$.

Following Bautista and Pereira (2007), we solve the original problem and the reverse problem respectively, and then choose a better solution from the solutions obtained.

There are two criteria for selecting the best solution: (1) number of workstations; (2) idle time in the last workstation. The second criterion is added since there are always large plateaus when only the first criterion is used, and more idle time in the last workstation means better resource utilization of the previous workstations. When there are several solutions with the same number of workstations, preference goes to the solution with more idle time in the last workstation, and then the solution will be chosen randomly if there are still ties after the above two steps.

## 3.2 The algorithm of ACO-BS

For the classical ant colony algorithm, many ants search for solutions separately in one iteration. According to Dorigo et al. (1996), there are three kinds of classical ant colony algorithm: ant system, ant-density and ant-quantity, and for the latter two models, each ant lays pheromone at each step; while for ant system, ants lay pheromone after the end of the tour. Thus, pheromone values are updated by global information in the ant system model, while local information is used to update the pheromone values in the other two models. Not surprisingly, the results of the ant

system model are better since global information rather than local information is used to guide the solution searching process.

However, there is one significant disadvantage of classical ACO that needs to be recognized: although the convergence of the algorithm is guaranteed, the time to convergence is uncertain. Due to the complexity of ALBP, the solution space is quite large. Thus, strategies should be developed to speed up the searching process. The solution construction mechanism of ACO map the search space onto a search tree, and this is similar to that of beam search, which is an adaptation of the branch and bound method in which only certain nodes are evaluated and only promising nodes are kept for further branching and the remaining nodes are pruned permanently (Sabuncuoglu and Bayiz, 1999). With ACO, the probabilistic method is used to extend the partial solutions by roulette-wheel selection with the chosen probability of each extension considered. However, with beam search, the partial solutions are chosen in a deterministic way by choosing the extension which has the largest probability to be chosen. This motivates us to combine these two approaches and propose ACO-BS. Both deterministic way and the probabilistic way can be utilized to extend the partial solutions.

Compared with the standard ACO, there are several improvements made by ACO-BS. To begin with, the priority rule introduced in section 3.2.1 is used to solve

SALBP and the reverse problem, and the optimal solution is used to initialize the best-so-far solution for ACO-BS. In this way, the searching speed is guaranteed since the algorithm will not waste time on solutions worse than the best-so-far solution. Also, for each workstation, a task is chosen in a deterministic method or a probabilistic method, with equal possibility, so that the solution extension is conducted with two different strategies. In addition, each partial solution extends for constant times, but for each level of extension, only some promising partial solutions are selected to extend at the next stage. In this way, quality of solutions can be guaranteed with acceptable computation cost. Therefore, the good performance of ACO-BS is expected.

### 3.2.1 Priority Rule

At first, the priority values of tasks are computed as follows:

$$\eta_j = \frac{t_j}{C} + \frac{|S_j|}{\max_{1 \le i \le n} |S_i|}, \quad j = 1, \ldots, n, \tag{3-6}$$

Where $|S_j|$ is the number of successors of task $j$. Starting from the first workstation, put all the tasks in set $N$ and set the idle time to be $C$. Also, tasks whose predecessors have been assigned are put into the set $N_{nopre}$. Figure 3-3 shows the flow chart for the priority rule. Task assignment is implemented by the following steps:

(1) determine the available tasks. Put all tasks in $N_{nopre}$ with processing times equal to the idle time into the available task set, $N_{av}$, since saturating the time resource of a workstation is preferable in order to improve the utilization of resources. If $N_{av}$ is empty, tasks in $N_{nopre}$ with processing times no longer than the remaining time are put into $N_{av}$.

(2) if $N_{av}$ is not empty, choose a task with the highest priority value from $N_{av}$ (if there is more than one task with the highest priority value, choose one from them randomly), and go to step (3); If the set is empty, go to step (4).

(3) delete the chosen task from $N$. Set $N_{nopre}$ and $N_{av}$ to be $\phi$, and decrease the idle time by the processing time of the chosen task. Go to step (1).

(4) close the current workstation. If $N$ is not empty, open a new one and set the cycle time to be $C$, then go to step (1); if $N$ is empty, end the procedure.

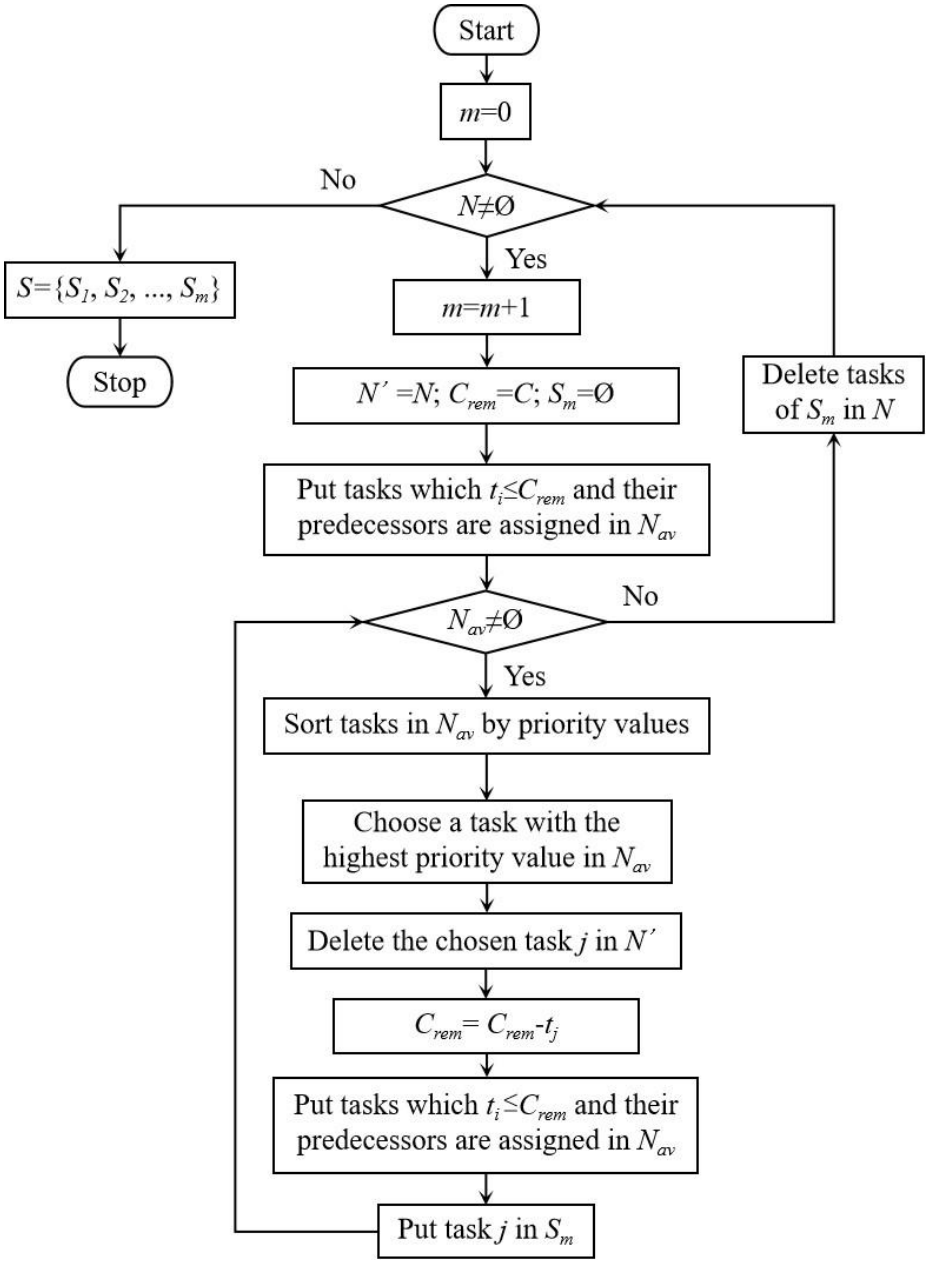


**Figure 3-3:** Flow chart for priority rule

### 3.2.2 ACO-BS

There are four steps in ACO-BS. The first step is used to initialize the parameters, and steps 2 to 4 are repeated within a certain time. The flow chart for ACO-BS is

shown in Figure 3-4.

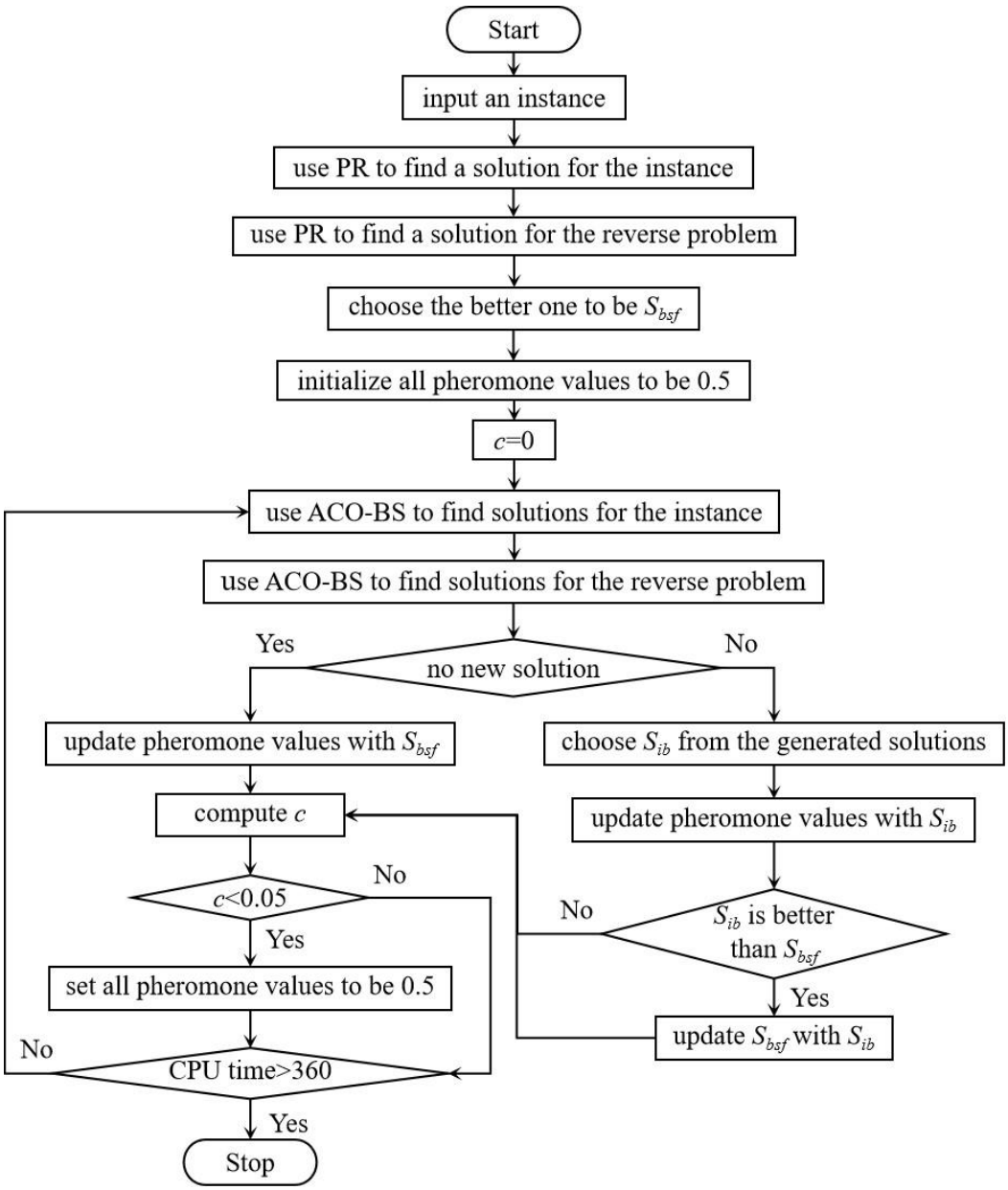


**Figure 3-4:** Flow chart for ACO-BS

**Step 1: Initialization**

Generate two optimal solutions by using the priority rule described in section 3.2.1 for the original problem and its reverse version. Then the two criteria introduced in section 3.1.3 are used to choose a better one to initialize the best-so-far solution.

Additionally, the pheromone value is used to guide the searching process for good solutions. Let $\tau_{ij}$ be the pheromone value between task $j$ and workstation $i$, and all the pheromone values are initialized to be 0.5.

**Step 2: Generate solutions from the original problem and the reverse one respectively**

Differing from the computation of priority values used in priority rule, the priority values used in ACO-BS are processed using equation (3-7) (Blum, 2008):

$$\eta'_j = \frac{\eta_j - \eta_{\min} + 1}{\eta_{\max}}, \quad j = 1,\ldots,n \tag{3-7}$$

where $\eta_{\min} = \min_{1\le j\le n} \eta_j$ and $\eta_{\max} = \max_{1\le j\le n} \eta_j$.

Let $p_j$ denote the probability that task $j$ is chosen by workstation $k$. A task is chosen by maximizing $p_j$ or by the roulette-wheel method, with the same probability. $p_j$ is calculated by the summation rule as follows (Merkle and Middendorf, 2000):

$$p_j = \frac{\left(\sum_{i=1}^{k} \tau_{ij}\right) \cdot \eta'_j}{\sum_{q \in N_{av}} \left(\sum_{i=1}^{k} \tau_{iq}\right) \cdot \eta'_q}. \quad (3\text{-}8)$$

Figure 3-5 shows the flow chart for the solution generation process of ACO-BS. To illustrate the procedure of the algorithm, we illustrate the situation for the first workstation at first, and then the next steps are given. At first, task assignment for the first workstation is explored for $n_{\text{ext}}$ times, and the procedure is the similar as that by priority rule, but task selection rule here contains pheromone values and the priority values of tasks. Let $S_{\text{par}}$ be the initial empty partial solution set, and $S_{ext}$ be the set that stores the task sets of the last workstation of all the partial solutions. For the first workstation, the two sets are the same. After each exploration, the task assignment for the first workstation, which is different to those in $S_{ext}$ and its lower bound (will be described later) is less than $|S_{bsf}|$, which is the number of workstations needed in the best-so-far solution, are put into $S_{\text{par}}$ and $S_{ext}$. Because the task assignment for the first workstation is also a partial solution.

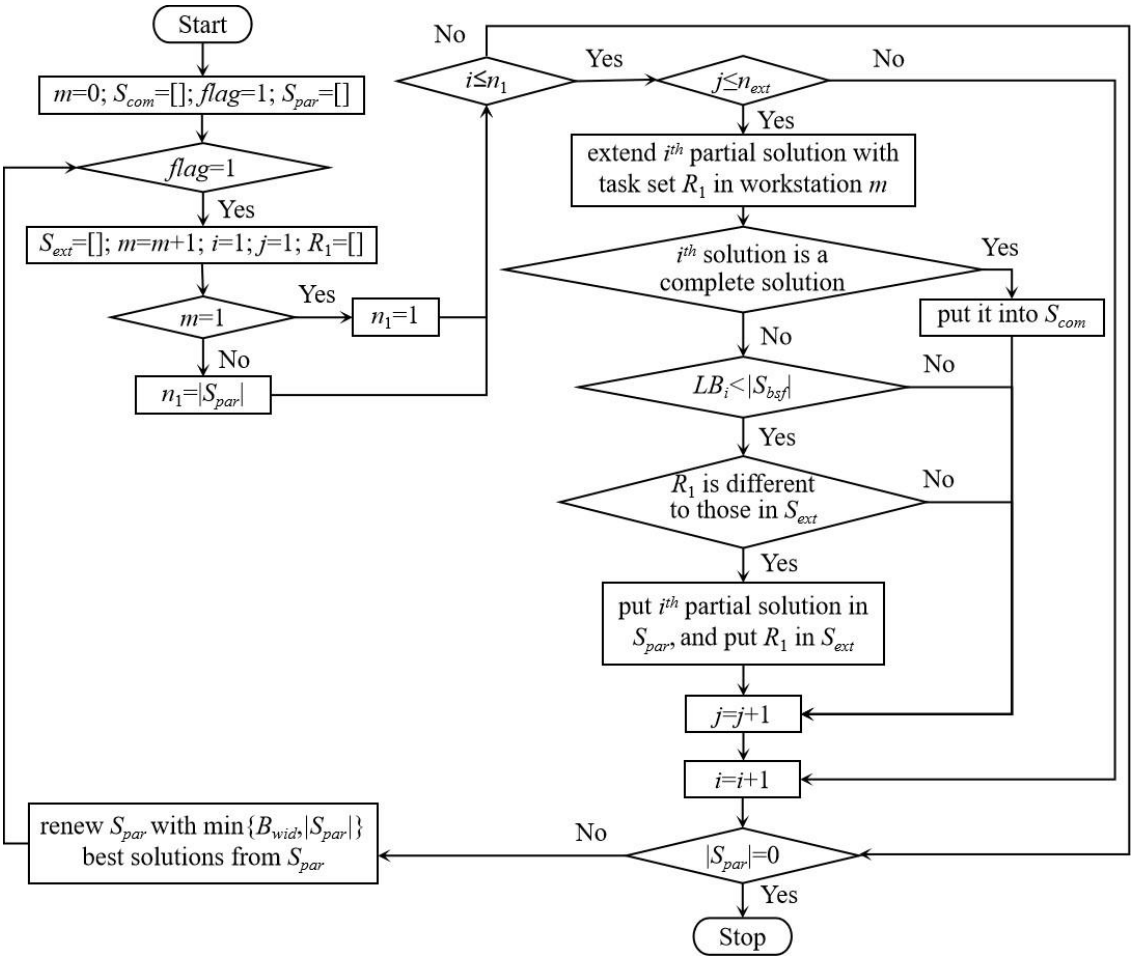


**Figure 3-5:** Flow chart for the solution generation process of ACO-BS

After the assignment of the first workstation, there will be at most $n_{\text{ext}}$ partial solutions in $S_{\text{par}}$. One partial solution is picked one time, and then the following steps are repeated until the partial solution extends for $n_{\text{ext}}$ times (Let $m$ denotes the workstation which is currently considered; $S_{ext} = \phi$):

(1) $ext = 1$. $R_1 = \phi$, and it is the task set for workstation $m$.

(2) assign tasks to workstation $m$, and put these tasks into $R_1$.

(3) extend the partial solution with $R_1$ for workstation $m$. If the extended solution is a complete solution, go to step (4); otherwise, go to step (5).

(4) put the extended partial solution to $S_{com}$ if it is a complete solution.

(5) if the lower bound (described below) of the workstation needed after the extension of the partial solution is less than $|S_{bsf}|$ and $R_1$ is different from all the factors in $S_{\text{ext}}$, the partial solution will be put into $S_{\text{par}}$.

(6) if $ext = n_{ext}$, end this procedure; otherwise, $\text{ext} = ext + 1$ and $R_1 = \phi$, and go to step (2).

Only partial solutions which will not lead to solutions worse than $S_{bsf}$ can be extended in the next round. Let $N_{rem}$ be the set of tasks not involved in the partial solution $s$, and the lower bound on the workstations needed is calculated as follows (Scholl and Becker, 2006):

$$LB_s = \left\lceil \frac{\sum_{j \in N_{rem}} t_j}{C} \right\rceil . \tag{3-9}$$

Partial solutions are ranked by increasing the lower bound defined above. If there are ties, our preference goes to partial solutions with less idle time in the last workstation (further ties are broken randomly). Finally, for each workstation considered, there will be $\min\{B_{wid}, |S_{par}|\}$ partial solutions generated, and $B_{wid}$

denotes the width of the beam and $|S_{par}|$ denotes the number of partial solutions obtained.

This step is ended when no partial solution can be extended.

**Step 3: Choose the iteration best solution and update pheromone values**

Since in step2, if the lower bound of a partial solution is no less than $|S_{bsf}|$, the partial solution will be aborted. Thus, it is possible that there is no solution obtained in step 2. If so, $S_{bsf}$ is used to update the pheromone values.

If there is a solution obtained in step 2, an iteration best solution, $S_{ib}$, is chosen with the two criteria introduced in section 3.1.3 to update the pheromone values. Pheromone values $\tau_{ij}$ between task $j$ and workstation $i$ ( $i = 1,\ldots,|S_{ib}|$ ; $j = 1,\ldots,n$ ) are updated with the following two steps: (1) pheromone evaporation. For each $\tau_{ij}$ needs to be updated, there is $(1-\rho)\cdot\tau_{ij}$ left after evaporation. $\rho \in (0,1]$ is the evaporation rate, assigned as 0.1 in this study. (2) $\tau_{ij}$ increases by $\rho$ when task $j$ is assigned to workstation $i$ in $S_{ib}$.

When $\tau_{ij}$ is too small, task $j$ tends never to be assigned to workstation $i$; When the value is too large, task $j$ tends always to be assigned to workstation $i$. Consequently, the solution space is small, and this may lead to bad quality of the

solutions generated. Thus, the pheromone values are restricted to the interval $[\tau_{\min}, \tau_{\max}]$ to prevent stagnation (López-Ibáñez et al., 2016), and $\tau_{\min} = 0.01$ and $\tau_{\max} = 0.99$. If a pheromone value is larger than $\tau_{\max}$, it will be replaced by $\tau_{\max}$; if the value is smaller than $\tau_{\min}$, it is replaced by $\tau_{\min}$.

If $S_{ib}$ is better than $S_{bsf}$ (based on the criteria in section 3.1.3), $S_{bsf}$ is updated by $S_{ib}$.

**Step 4: Calculating the convergence value**

According to Kong et al. (2008), pheromone re-initialization is an important strategy to avoid premature convergence by preventing the algorithm from searching around a local optima continuously with low effectiveness. The convergence value is calculated as follows (Blum, 2008):

$$c = 2 \cdot \left( \frac{\sum_{j=1}^{n} \sum_{i=1}^{|S_{bsf}|} \min\{\tau_{\max} - \tau_{ij}, \tau_{ij} - \tau_{\min}\}}{n \cdot |S_{bsf}| \cdot (\tau_{\max} - \tau_{\min})} \right). \quad (3\text{-}10)$$

After step 1, the convergence value is 1. All the pheromone values are initialized to be 0.5 when the convergence value is less than 0.05.

## 3.3 Computational Results of Benchmark Instances

The computational results are obtained by running MATLAB, using an Intel Core i7-6700 (3.40 gigahertz) processor with 32 gigabytes of available memory. The computation times and the standard deviations are reported in CPU time seconds.

### 3.3.1 Results by ACO-BS

In order to exhibit the superior performance of the algorithm developed in this paper, we tested the algorithm with benchmark instances (SALBP-I) published on https://assembly-line-balancing.de/.

There are 269 benchmark instances of SALBP-I. Optimal solutions can be obtained for 170 instances by using the priority rule only. After ten runs of ACO-BS (360 CPU time seconds for each run, there are 87 more instances whose optimal solutions can be obtained by the ACO-BS algorithm. There are 12 instances whose optimal solutions cannot be found by ACO-BS ( $n_{ext} = 10$ , $B_{wid} = 20$ ), however, when the time limit increases, the results are better. For example, for the instance Warnecke (with task number of 58 and cycle time of 60), the optimal result can be found, with the average solution found to be 27.7 (standard variation is 0.483); there are three runs in ten in which the optimal solution can be found, with the running times of 2931.790, 2687.077 and 3570.333.

Tables 3-1, 3-2 and 3-3 show the results of the benchmark instances. For each instance, the given cycle time, best solution ever found, solution found by priority rule, the best solution found by ACO-BS, the difference between solution found by priority rule and the best solution found by ACO-BS are reported. Besides, the average and standard variation of solutions found in ten runs and the running times are also reported. The running time here is the computational time to find the best solution by ACO-BS for the first time. We can see from Tables 3-1, 3-2 and 3-3 that the algorithm performs well in most instances, but there are some tricky instances whose optimal results cannot be found or cannot be found in every run.

**Table 3-1:** Results of benchmark instances 1~33

| Instances | C | Optimal solution | Priority rule | ACO-BS | Difference | Solution | | Running time [s] | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | avg. | std. | avg. | std. |
| Arcus1 | 3786 | 21 | 22 | 21* | 1 | 21 | 0 | 3.437 | 0.077 |
| Arcus2 | 11570 | 13 | 14 | 13* | 1 | 13 | 0 | 65.417 | 60.831 |
| Barthol2 | 84 | 51 | 52 | 51* | 1 | 51 | 0 | 7.575 | 0.127 |
| Barthol2 | 85 | 50 | 51 | 51 | 0 | $51^{T}$ | 0 | 0.254 | 0.022 |
| Barthol2 | 87 | 49 | 50 | 49* | 1 | 49 | 0 | 7.818 | 0.161 |
| Barthol2 | 89 | 48 | 49 | 48* | 1 | 48 | 0 | 8.277 | 0.182 |
| Barthol2 | 91 | 47 | 48 | 47* | 1 | 47 | 0 | 8.333 | 0.257 |
| Barthol2 | 93 | 46 | 47 | 46* | 1 | 46 | 0 | 8.034 | 0.201 |
| Barthol2 | 95 | 45 | 46 | 45* | 1 | 45 | 0 | 8.193 | 0.153 |
| Barthol2 | 97 | 44 | 45 | 44* | 1 | 44 | 0 | 8.013 | 0.175 |
| Barthol2 | 99 | 43 | 44 | 43* | 1 | 43 | 0 | 9.656 | 3.344 |
| Barthol2 | 101 | 42 | 43 | 42* | 1 | 42 | 0 | 11.605 | 7.768 |
| Barthol2 | 104 | 41 | 42 | 41* | 1 | 41 | 0 | 8.338 | 0.182 |
| Barthol2 | 106 | 40 | 41 | 40* | 1 | 40 | 0 | 10.094 | 3.506 |
| Barthol2 | 109 | 39 | 40 | 39* | 1 | 39 | 0 | 8.675 | 0.203 |
| Barthol2 | 112 | 38 | 39 | 38* | 1 | 38 | 0 | 8.583 | 0.197 |
| Barthol2 | 115 | 37 | 38 | 37* | 1 | 37 | 0 | 8.758 | 0.174 |
| Barthol2 | 118 | 36 | 37 | 36* | 1 | 36 | 0 | 8.864 | 0.127 |
| Barthol2 | 121 | 35 | 36 | 35* | 1 | 35 | 0 | 11.590 | 4.274 |
| Barthol2 | 125 | 34 | 35 | 34* | 1 | 34 | 0 | 8.886 | 0.153 |
| Barthol2 | 129 | 33 | 34 | 33* | 1 | 33 | 0 | 9.145 | 0.144 |
| Barthol2 | 133 | 32 | 33 | 32* | 1 | 32 | 0 | 9.107 | 0.175 |
| Barthol2 | 137 | 31 | 32 | 31* | 1 | 31 | 0 | 9.190 | 0.115 |
| Barthol2 | 146 | 29 | 30 | 29* | 1 | 29 | 0 | 10.081 | 2.874 |
| Barthol2 | 152 | 28 | 29 | 28* | 1 | 28 | 0 | 9.309 | 0.130 |
| Barthol2 | 157 | 27 | 28 | 27* | 1 | 27 | 0 | 9.233 | 0.117 |
| Barthol2 | 163 | 26 | 27 | 26* | 1 | 26 | 0 | 9.232 | 0.118 |
| Barthol2 | 170 | 25 | 26 | 25* | 1 | 25 | 0 | 9.306 | 0.109 |
| Barthold | 403 | 14 | 15 | 14* | 1 | 14 | 0 | 8.093 | 0.255 |
| Barthold | 434 | 13 | 14 | 13* | 1 | 13 | 0 | 8.597 | 0.108 |
| Barthold | 470 | 12 | 13 | 12* | 1 | 12 | 0 | 8.691 | 0.130 |
| Barthold | 513 | 11 | 12 | 11* | 1 | 11 | 0 | 8.618 | 0.131 |
| Barthold | 626 | 9 | 10 | 9* | 1 | 9 | 0 | 8.423 | 0.103 |

Notes: * indicates an optimal solution is found; $^{T}$ indicates trickiness of a solution

**Table 3-2:** Results of benchmark instances 34~66

| Instances | C | Optimal solution | Priority rule | ACO-BS | Difference | Solution | | Running time [s] | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | avg. | std. | avg. | std. |
| Buxey | 41 | 8 | 9 | 8* | 1 | 8 | 0 | 0.812 | 0.014 |
| Buxey | 47 | 7 | 8 | 7* | 1 | 7 | 0 | 16.977 | 4.762 |
| Gunther | 54 | 9 | 10 | 9* | 1 | 9 | 0 | 1.035 | 0.011 |
| Heskiaoff | 205 | 5 | 6 | 5* | 1 | 5 | 0 | 1.050 | 0.031 |
| Jackson | 10 | 5 | 6 | 5* | 1 | 5 | 0 | 0.081 | 0.015 |
| Kilbridge | 69 | 8 | 9 | 8* | 1 | 8 | 0 | 1.784 | 0.065 |
| Kilbridge | 79 | 7 | 8 | 7* | 1 | 7 | 0 | 1.758 | 0.022 |
| Kilbridge | 92 | 6 | 7 | 6* | 1 | 6 | 0 | 1.736 | 0.050 |
| Kilbridge | 138 | 4 | 5 | 4* | 1 | 4 | 0 | 1.567 | 0.035 |
| Lutz2 | 11 | 49 | 50 | 49* | 1 | 49 | 0 | 8.561 | 6.649 |
| Lutz2 | 12 | 44 | 47 | 44* | 3 | $44.5^{T}$ | 0.527 | 99.253 | 84.952 |
| Lutz2 | 13 | 40 | 42 | 40* | 2 | 40 | 0 | 7.149 | 6.477 |
| Lutz2 | 14 | 37 | 38 | 37* | 1 | $37.7^{T}$ | 0.483 | 61.574 | 105.705 |
| Lutz2 | 15 | 34 | 35 | 34* | 1 | 34 | 0 | 7.567 | 4.152 |
| Lutz2 | 16 | 31 | 33 | 31* | 2 | 31 | 0 | 8.813 | 5.541 |
| Lutz2 | 17 | 29 | 30 | 29* | 1 | 29 | 0 | 14.619 | 6.265 |
| Lutz2 | 18 | 28 | 29 | 28* | 1 | 28 | 0 | 27.185 | 11.602 |
| Lutz2 | 19 | 26 | 27 | 26* | 1 | 26 | 0 | 8.493 | 5.250 |
| Lutz2 | 20 | 25 | 26 | 25* | 1 | 25 | 0 | 2.688 | 0.081 |
| Lutz2 | 21 | 24 | 25 | 24* | 1 | 24 | 0 | 2.748 | 0.078 |
| Lutz3 | 110 | 15 | 16 | 15* | 1 | $15.8^{T}$ | 0.422 | 31.939 | 42.531 |
| Lutz3 | 118 | 14 | 15 | 14* | 1 | $14.4^{T}$ | 0.516 | 82.378 | 107.850 |
| Mansoor | 62 | 3 | 4 | 3* | 1 | 3 | 0 | 0.130 | 0.007 |
| Mukherje | 201 | 22 | 23 | 22* | 1 | 22 | 0 | 4.698 | 0.094 |
| Sawyer | 41 | 8 | 9 | 8* | 1 | 8 | 0 | 0.959 | 0.020 |
| Sawyer | 47 | 7 | 8 | 7* | 1 | $7.1^{T}$ | 0.316 | 225.300 | 145.082 |
| Tonge | 251 | 14 | 15 | 14* | 1 | $14.7^{T}$ | 0.483 | 17.315 | 36.100 |
| Tonge | 320 | 11 | 12 | 11* | 1 | 11 | 0 | 3.575 | 0.079 |
| Warnecke | 54 | 31 | 32 | 31* | 1 | 31 | 0 | 50.685 | 47.707 |
| Warnecke | 56 | 29 | 30 | 29* | 1 | 29 | 0 | 20.740 | 14.768 |
| Warnecke | 60 | 27 | 29 | 28 | 1 | $28^{T}$ | 0 | 7.998 | 5.720 |
| Warnecke | 62 | 27 | 28 | 27* | 1 | 27 | 0 | 21.701 | 16.420 |
| Warnecke | 65 | 25 | 27 | 25* | 1 | 25 | 0 | 4.152 | 2.172 |

**Table 3-3:** Results of benchmark instances 67~99

| Instances | C | Optimal solution | Priority rule | ACO-BS | Difference | Solution | | Running time [s] | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | avg. | std. | avg. | std. |
| Warnecke | 68 | 24 | 25 | 24* | 1 | 24 | 0 | 13.874 | 11.894 |
| Warnecke | 71 | 23 | 24 | 23* | 1 | 23 | 0 | 14.076 | 13.767 |
| Warnecke | 82 | 20 | 21 | 20* | 1 | 20 | 0 | 3.393 | 2.080 |
| Warnecke | 92 | 17 | 18 | 17* | 1 | 17 | 0 | 4.433 | 3.124 |
| Warnecke | 104 | 15 | 16 | 15* | 1 | 15 | 0 | 2.266 | 0.037 |
| Warnecke | 111 | 14 | 15 | 14* | 1 | 14 | 0 | 3.053 | 1.544 |
| Wee-mag | 30 | 62 | 63 | 62* | 1 | 62 | 0 | 6.444 | 2.356 |
| Wee-mag | 46 | 34 | 35 | 34* | 1 | 34 | 0 | 8.894 | 6.803 |
| Wee-mag | 47 | 32 | 33 | 33 | 0 | $33^{T}$ | 0 | 10.019 | 5.176 |
| Wee-mag | 52 | 31 | 32 | 31* | 1 | 31 | 0 | 4.056 | 0.040 |
| Wee-mag | 56 | 30 | 31 | 30* | 1 | 30 | 0 | 4.113 | 0.034 |
| Scholl | 1394 | 50 | 51 | 51 | 0 | $51^{T}$ | 0 | 7.531 | 9.124 |
| Scholl | 1452 | 48 | 49 | 48* | 1 | $48.9^{T}$ | 0.316 | 24.050 | 7.363 |
| Scholl | 1483 | 47 | 48 | 48 | 0 | $48^{T}$ | 0 | 12.256 | 19.041 |
| Scholl | 1515 | 46 | 47 | 47 | 0 | $47^{T}$ | 0 | 13.600 | 10.953 |
| Scholl | 1584 | 44 | 45 | 44* | 1 | $44.9^{T}$ | 0.316 | 75.227 | 118.097 |
| Scholl | 1659 | 42 | 43 | 43 | 0 | $43^{T}$ | 0 | 3.568 | 0.032 |
| Scholl | 1742 | 40 | 41 | 40* | 1 | $40.8^{T}$ | 0.422 | 47.947 | 66.518 |
| Scholl | 1787 | 39 | 40 | 39* | 1 | $39.8^{T}$ | 0.422 | 174.619 | 130.467 |
| Scholl | 1834 | 38 | 39 | 38* | 1 | $38.2^{T}$ | 0.422 | 200.751 | 111.410 |
| Scholl | 1883 | 37 | 38 | 38 | 0 | $38^{T}$ | 0 | 149.149 | 160.662 |
| Scholl | 1935 | 36 | 37 | 37 | 0 | $37^{T}$ | 0 | 185.443 | 149.573 |
| Scholl | 1991 | 35 | 36 | 35* | 1 | $35.1^{T}$ | 0.316 | 200.308 | 113.994 |
| Scholl | 2049 | 34 | 35 | 35 | 0 | $35^{T}$ | 0 | 45.256 | 18.299 |
| Scholl | 2111 | 33 | 34 | 34 | 0 | $34^{T}$ | 0 | 50.165 | 74.052 |
| Scholl | 2177 | 32 | 33 | 32* | 1 | $32.7^{T}$ | 0.483 | 156.656 | 103.493 |
| Scholl | 2247 | 31 | 32 | 32 | 0 | $32^{T}$ | 0 | 47.580 | 24.497 |
| Scholl | 2322 | 30 | 31 | 30* | 1 | $30.9^{T}$ | 0.316 | 75.187 | 89.942 |
| Scholl | 2402 | 29 | 30 | 29* | 1 | $29.4^{T}$ | 0.516 | 141.810 | 110.054 |
| Scholl | 2488 | 28 | 29 | 28* | 1 | $28.7^{T}$ | 0.483 | 60.101 | 69.595 |
| Scholl | 2580 | 27 | 28 | 27* | 1 | $27.2^{T}$ | 0.422 | 142.472 | 102.649 |
| Scholl | 2680 | 26 | 27 | 26* | 1 | 26 | 0 | 35.804 | 16.163 |
| Scholl | 2787 | 25 | 26 | 25* | 1 | 25 | 0 | 49.832 | 28.863 |

### 3.3.2 Comparative Results of ACO, GA and PSO

According to the results shown in Tables 3-1, 3-2 and 3-3, ACO-BS can achieve significantly better results than those obtained by the priority rule. However, further comparative experiments are needed to show the superiority of ACO-BS. ACO, which has a similar framework with ACO-BS except for the beam search part, GA in Leu et al. (1994) and PSO in Dou et al. (2013) are compared with ACO-BS.

Since ACO-BS begins with a solution obtained by the priority rule, the other algorithms will also use the priority rule to get the initial solutions. Specifically, for ACO, the best-so-far solution is initialized by the best solution obtained by the priority rule, and the number of ants is set to be 20 since the beam width is 20 in ACO-BS. As to GA, the population size is 50, and initial solutions are obtained by the priority rule and four heuristic methods in Leu et al. (1994) (except the third one in Leu et al. (1994)). For PSO, there are 30 initial solutions, with 10 solutions obtained the same way as the 10 initial solutions for GA and the other initial solutions generated randomly. Thus, the best solutions found by ACO, GA and PSO will not be worse than those found by the priority rule as well. The other parameters in GA and PSO are the same as those in the corresponding two published papers.

Figure 3-6 shows the comparative results of ACO, GA and PSO on the 99 benchmark instances whose optimal solutions cannot be reached by the priority rule. On the x-axis are the 99 instances, and on the y-axis are the differences between the best solutions found in 10 runs within 360 CPU time seconds and the corresponding optimal solutions. The numbers of instances that can reach optimal solutions by ACO, GA and PSO are 33, 24 and 23, respectively. For GA and PSO, the difference between the best solution found and the corresponding optimal one ranges from 0 to 2. However, the difference between the best solution found by ACO and the corresponding optimal solution ranges from 0 to 1. Thus, among the three algorithms, ACO has comparatively better performance in solving ALBP.

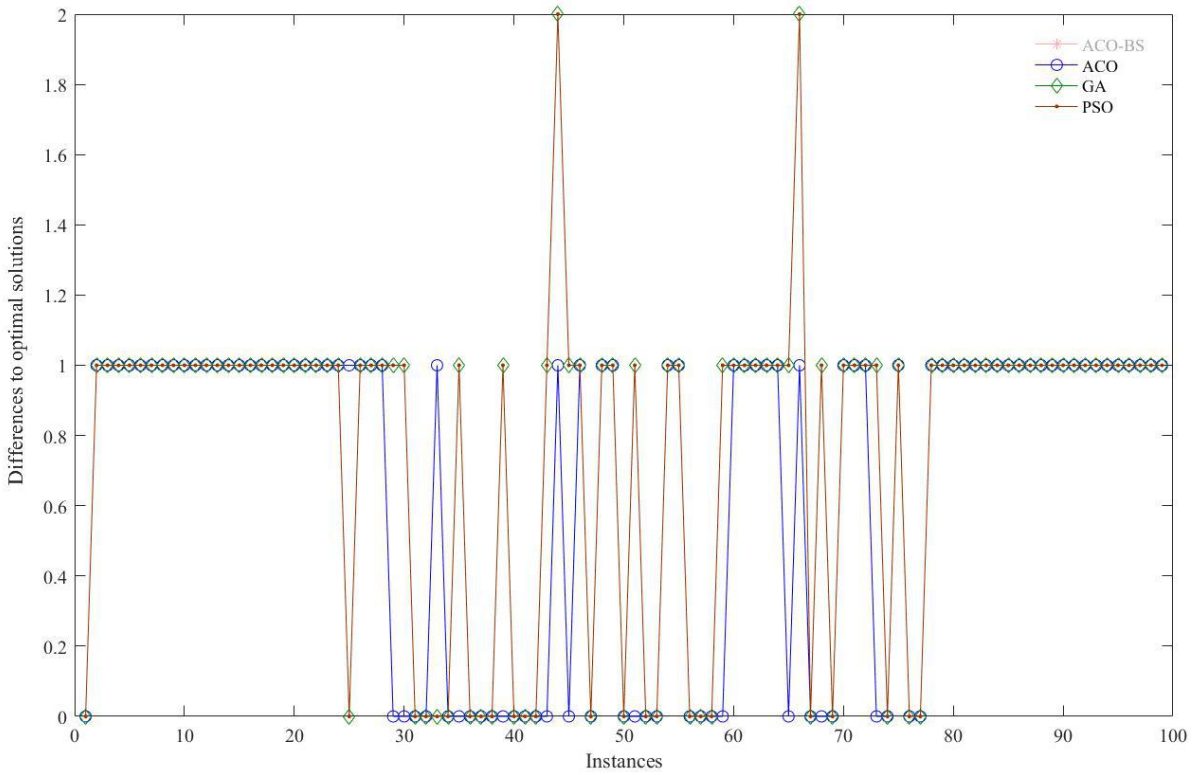


**Figure 3-6:** Comparative results of ACO, GA and PSO on benchmark instances

In order to highlight the benefits brought by the integration of beam search and ACO, the performance of ACO is compared with that of ACO-BS. With the increase of beam width in ACO-BS from 20 to 100, the number of ants used in ACO is increased to be 100, and the running time limit is set to be 720 CPU time seconds. The comparative results are shown in Figure 3-7. The numbers of instances whose optimal solutions can be reached by ACO and ACO-BS are 33 and 87, respectively. Thus, the performance of ACO-BS is better than that of ACO.

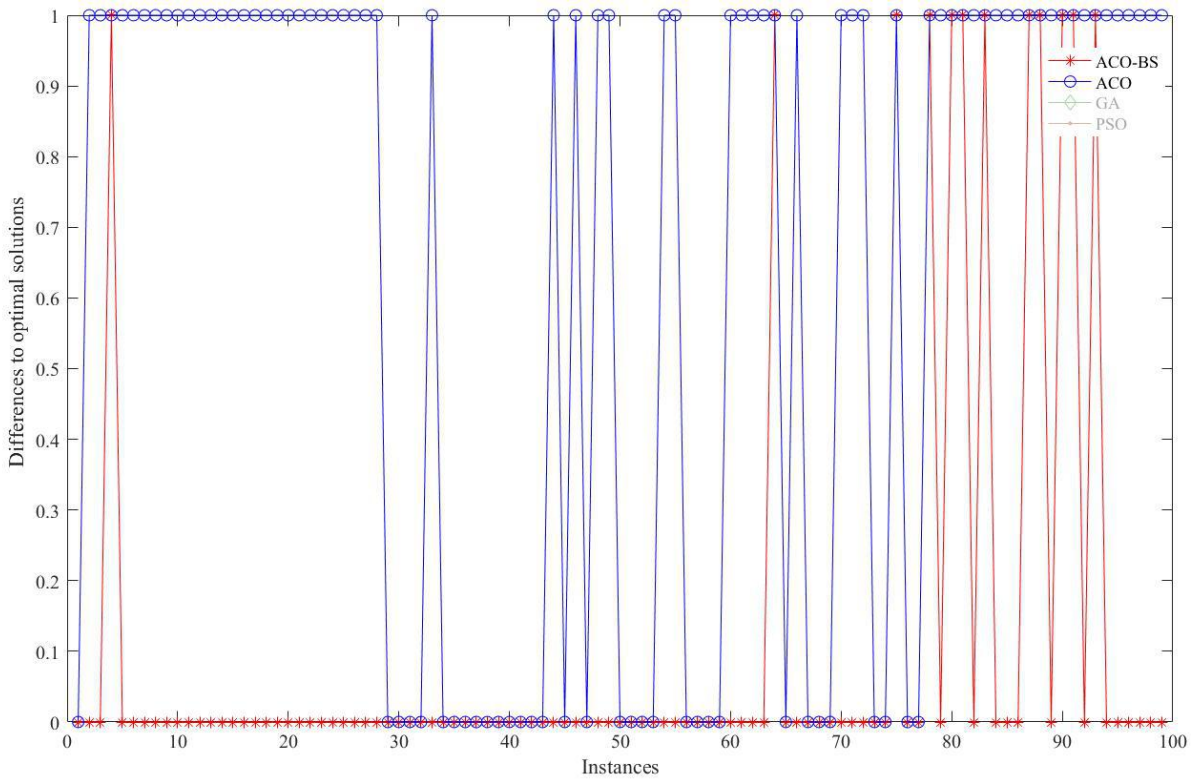


**Figure 3-7:** Comparative results between ACO and ACO-BS on benchmark instances

Therefore, ACO shows superiority compared with GA in Leu et al. (1994) and PSO in Dou et al. (2013), and ACO-BS improves ACO by integrating beam search and ACO.

## 3.4 Computational Results of Randomly Generated Instances

According to Scholl (1993), the following three indicators can be used to measure the complexity of the ALBP instances:

(1) Order Strength (OS). OS is defined as the number of arcs in the transitive closure of the precedence graph divided by $\frac{n \cdot (n-1)}{2}$, that is, the maximal number of arcs in an acyclic graph with $n$ nodes. The middle values of OS make an instance more complicated, compared with the low or high order strength values (Morrison et al., 2014). Nevertheless, when OS is 1 there is only one task sequence feasible; when OS is 0, SALBP-I becomes the bin packing problem, which is also NP-hard (Scholl, 1993).

(2) Time Variability (TV). TV is measured by $\frac{t_{\max}}{t_{\min}}$, and reflects the structure of processing times. $t_{\max}$ and $t_{\min}$ denote the longest and shortest processing time, respectively. A smaller TV suggests a higher complexity.

(3) Time Interval (TI). TI is defined as $\left[\frac{t_{\min}}{C}, \frac{t_{\max}}{C}\right]$, and indicates the relation between the cycle time and the processing times. Instances with a time interval that is small and near to 1 are expected to be relatively complicated.

To have a better understanding of the complex instances marked in Tables 3-1, 3-2

and 3-3, the characteristics of these instances are analyzed and shown in Table 3-4. For instances of Scholl, only the statistical information of tricky instances with the smallest and largest cycle time is reported. Table 3-4 shows that $t_{\min}$ is quite small (usually 1, with 5 and 7 as larger values). OS is around 0.2, 0.6 or 0.8. TV ranges from 7.571 to 277.2. Additionally, $\frac{t_{\min}}{C}$ tends to be less than 0.1, and $\frac{t_{\max}}{C}$ ranges from 0.532 to 0.994. The ratio of mean processing time to cycle time varies from 0.157 to 0.445. The variation of processing times ranges from 8.205 to 38911.047.

**Table 3-4:** Characteristics of tricky instances

| Instances | $C$ | Mean | Var | $t_{min}$ | $t_{max}$ | TV | $t_{min}/C$ | $t_{max}/C$ | Mean/C | OS |
|---|---|---|---|---|---|---|---|---|---|---|
| Barthol2 | 85 | 28.608 | 356.716 | 1 | 83 | 83 | 0.012 | 0.977 | 0.337 | 0.258 |
| Lutz2 | 12 | 5.449 | 8.205 | 1 | 10 | 10 | 0.083 | 0.833 | 0.454 | 0.776 |
| Lutz2 | 14 | 5.449 | 8.205 | 1 | 10 | 10 | 0.071 | 0.714 | 0.389 | 0.776 |
| Lutz3 | 110 | 18.472 | 184.525 | 1 | 74 | 74 | 0.009 | 0.673 | 0.168 | 0.776 |
| Lutz3 | 118 | 18.472 | 184.525 | 1 | 74 | 74 | 0.009 | 0.627 | 0.157 | 0.776 |
| Sawyer | 47 | 10.800 | 36.855 | 1 | 25 | 25 | 0.021 | 0.532 | 0.230 | 0.448 |
| Tonge | 251 | 50.143 | 1505.400 | 1 | 156 | 156 | 0.004 | 0.622 | 0.200 | 0.200 |
| Warnecke | 60 | 26.690 | 206.077 | 7 | 53 | 7.571 | 0.117 | 0.883 | 0.445 | 0.591 |
| Wee-mag | 47 | 19.987 | 46.419 | 2 | 27 | 13.5 | 0.043 | 0.575 | 0.425 | 0.227 |
| Scholl | 1394 | 234.529 | 38911.047 | 5 | 1386 | 277.2 | 0.004 | 0.994 | 0.168 | 0.582 |
| Scholl | 2247 | 234.529 | 38911.047 | 5 | 1386 | 277.2 | 0.002 | 0.617 | 0.104 | 0.582 |

Finally, we choose OS and TV to measure the tricky level of each instance, and set three levels of OS (0.2, 0.6, 0.9) and three levels of TV (5-15, 65-75, 135-145). As we want to explore the large-scale instance, we choose the problem size of 400. Since the tricky level and the problem size level are high, time limit of one run is

enlarged from 360 CPU time seconds to 720 CPU time seconds. Besides, the width of beam increases to be 100, and the number of extensions increases to 30.

### 3.4.1 Generation of Random Instances

The random instance generation consists of two parts: arc generation and task time generation.

(1) Arc generation. According to Otto et al. (2013), the concept of stages allows for direct manipulation of stages characteristics. Following Otto et al. (2013) and Kolisch et al. (1995), we use three steps to generate arcs. Firstly, the average number of tasks per stage is selected, and then the number of tasks per stage is generated following a truncated normal distribution (that is, the number is generated following a normal distribution iteratively until the number is no less than 1). Next, each beginning node (nodes have no predecessor) is assigned one successor, and each of other nodes is assigned one predecessor. Then, one successor is chosen randomly for those having no successor. The second step is repeated until the expected complexity is reached. During the above-mentioned procedure, the following aspects should be considered: First, there should not be redundant arcs. According to Kolisch et al. (1995), let $N=(V,A)$ be a network with node set $V$ and arc set $A$, and an arc $(i_0,i_s)$ is called redundant if there

are arcs $(i_0, i_1)$, ..., $(i_{s-1}, i_s) \in A$ and $s \geq 2$. The redundant arcs should be deleted. Second, predecessors and successors of nodes can only be chosen from the previous stage and the next stage, respectively. Last, tasks are always considered in the order of increasing order, and the added precedence relationships follow the topological rule.

(2) Task time generation. Kilbridge and Wester (1961) found that task times usually follow a unimodal or a bimodal distribution. Following Morrison et al. (2014), processing times of tasks are generated randomly according to some pre-specified normal distribution. Three kinds of task times are generated: peak at the bottom: tasks times are drawn from a normal distribution with the mean centered around a small value; peak in the middle: task times are drawn from a normal distribution with the mean of half of the cycle time; bimodal: task times are drawn from a combination of two normal distribution with means centered around two values. Besides, task times are rounded to the next integer and possible rounding effects are compensated by setting the default cycle time to 1000, which is large enough to allow flexible time structures (Otto et al., 2013).

### 3.4.2 Results of Random Instances

Tables 3-5 and 3-6 show the statistical description of the precedence graph and processing times of random instances, respectively. For the OS levels of 0.2 and 0.4, the number of stages is 40, while for the OS level of 0.9, the number of stages is 50. The number of stages is tuned by hand, and we find that when the number of stages is large, the number of iterations to add arcs to increase OS is smaller.

**Table 3-5:** Statistical description of precedence graphs of random instances

| OS level | OS | Number of stages | Number of beginning nodes | Number of ending nodes | Number of pairs precedence relations |
|---|---|---|---|---|---|
| 0.2 | 0.208 | 40 | 3 | 2 | 551 |
| 0.6 | 0.607 | 40 | 3 | 4 | 862 |
| 0.9 | 0.904 | 50 | 2 | 7 | 1079 |

**Table 3-6:** Statistical description of processing times of random instances

| TV level | Statistical information | | | | | Types of processing times | | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Var | $t_{min}$ | $t_{max}$ | TV | Type | mean | std. |
| 5-15 | 253.902 | 5899.542 | 35 | 457 | 13.057 | bottom | 250 | 80 |
| | 501.938 | 22247.106 | 63 | 855 | 13.572 | central | 500 | 150 |
| | 825.698 | 19925.720 | 69 | 999 | 14.478 | bimodal | 250 (750) | 100 (250) |
| 65-75 | 251.145 | 6878.485 | 8 | 539 | 67.375 | bottom | 250 | 80 |
| | 502.035 | 21738.550 | 14 | 985 | 70.357 | central | 500 | 150 |
| | 816.795 | 23440.324 | 14 | 999 | 71.357 | bimodal | 250 (750) | 100 (250) |
| 135-145 | 253.125 | 6818.070 | 4 | 577 | 144.25 | bottom | 250 | 80 |
| | 500.297 | 25632.059 | 7 | 978 | 139.714 | central | 500 | 150 |
| | 832.472 | 19895.062 | 7 | 999 | 142.714 | bimodal | 250 (750) | 100 (250) |

Note: figures in the brackets of the last two columns are the standard deviations used to generate a bimodal distribution.

Figure 3-8 shows the comparative results of ACO and ACO-BS. For 11 instances, the solutions found by ACO are the same with those by ACO-BS, while for the other 16 instances, ACO-BS performs better than ACO.

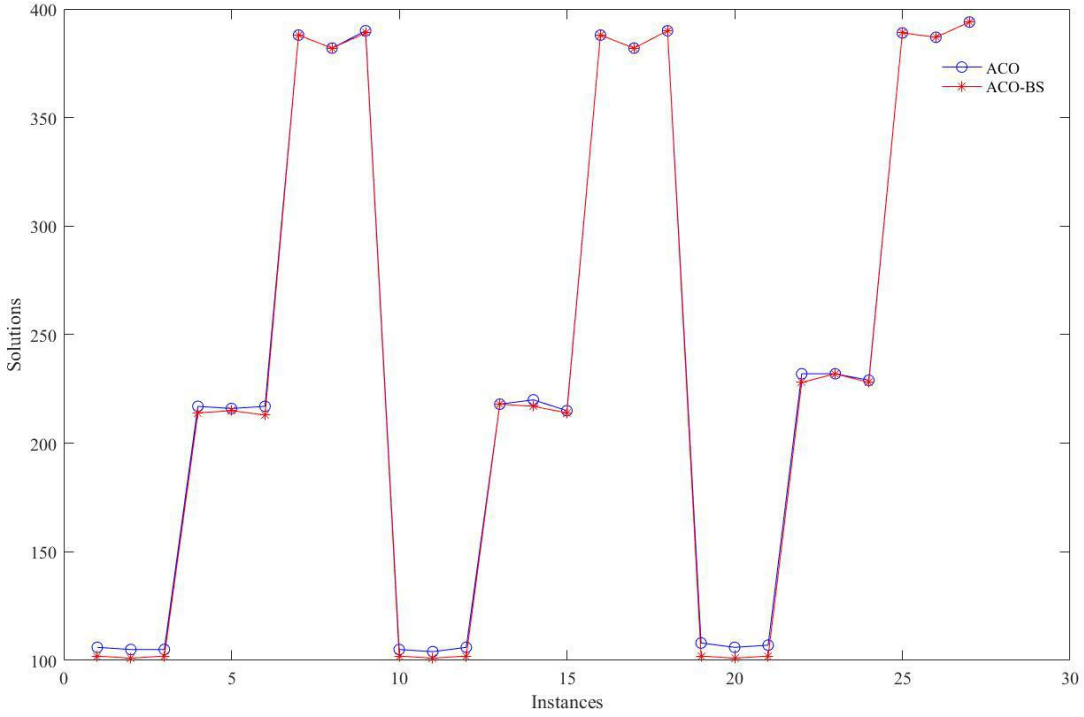


**Figure 3-8:** Comparative results between ACO and ACO-BS on 27 random instances

Table 3-7 shows the computational results of random instances. Surprisingly, the solutions found by PR are the same as those found by ACO. The column of difference refers to the differences between the solutions obtained by priority rule and those obtained by ACO-BS. We can see from Table 3-7 that the most significant

tendency is that instances whose processing times follow the bimodal distribution are more difficult to solve, since when the bimodal distribution is used to generate task times, compared with ACO, ACO-BS improves the solution quality for only one instance (OS level is 0.2, TV level is 135-145). This finding is consistent with Morrison et al. (2014).

Besides, there are two instances where there is no improvement in solution quality, with OS levels of 0.6 and 0.9 respectively and TV levels of 5-15 and 65-75 respectively. However, the similarity of these two instances is that their processing times are generated following the distribution peaking in the middle.

In addition, it seems that the standard deviations of the solutions obtained by ACO-BS for instances with processing times generated following the distribution peaking at the bottom tend to be 0. This implies that these kinds of instances are easiest to be solved.

**Table 3-7:** Computational results of random instances

| OS level | TV level | Type | PR/ACO | ACO-BS | Difference | Solution | | Running time [s] | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | avg. | std. | avg. | std. |
| 0.2 | 5-15 | bottom | 106 | 102 | 4 | 102 | 0 | 466.411 | 10.825 |
| | 65-75 | bottom | 105 | 101 | 4 | 101 | 0 | 569.084 | 2.131 |
| | 135-145 | bottom | 105 | 102 | 3 | 102 | 0 | 559.794 | 6.917 |
| | 5-15 | middle | 217 | 214 | 3 | 215 | 1 | 466.249 | 318.705 |
| | 65-75 | middle | 216 | 215 | 1 | 215.4 | 0.548 | 369.356 | 155.936 |
| | 135-145 | middle | 217 | 213 | 4 | 213.8 | 1.304 | 305.772 | 106.824 |
| | 5-15 | bimodal | 388 | 388 | 0 | 388 | 0 | 14.027 | 0.055 |
| | 65-75 | bimodal | 382 | 382 | 0 | 382 | 0 | 13.947 | 0.067 |
| | 135-145 | bimodal | 390 | 389 | 1 | 389.6 | 0.548 | 1048.546 | 946.530 |
| 0.6 | 5-15 | bottom | 105 | 102 | 3 | 102 | 0 | 535.052 | 2.433 |
| | 65-75 | bottom | 104 | 101 | 3 | 101 | 0 | 546.670 | 5.939 |
| | 135-145 | bottom | 106 | 102 | 4 | 102 | 0 | 535.687 | 4.069 |
| | 5-15 | middle | 218 | 218 | 0 | 218 | 0 | 111.478 | 133.562 |
| | 65-75 | middle | 220 | 217 | 3 | 217.6 | 0.548 | 435.175 | 135.772 |
| | 135-145 | middle | 215 | 214 | 1 | 214.8 | 0.447 | 452.758 | 102.803 |
| | 5-15 | bimodal | 388 | 388 | 0 | 388 | 0 | 12.368 | 0.929 |
| | 65-75 | bimodal | 382 | 382 | 0 | 382 | 0 | 13.208 | 1.178 |
| | 135-145 | bimodal | 390 | 390 | 0 | 390 | 0 | 12.976 | 1.496 |
| 0.9 | 5-15 | bottom | 108 | 102 | 6 | 102.8 | 0.447 | 543.545 | 236.263 |
| | 65-75 | bottom | 106 | 101 | 5 | 101 | 0 | 482.360 | 115.202 |
| | 135-145 | bottom | 107 | 102 | 5 | 102 | 0 | 275.540 | 1.023 |
| | 5-15 | middle | 232 | 228 | 4 | 230.2 | 1.643 | 558.983 | 261.882 |
| | 65-75 | middle | 232 | 232 | 0 | 232 | 0 | 14.075 | 0.045 |
| | 135-145 | middle | 229 | 228 | 1 | 228.8 | 0.447 | 264.277 | 348.637 |
| | 5-15 | bimodal | 389 | 389 | 0 | 389 | 0 | 14.141 | 0.0592 |
| | 65-75 | bimodal | 387 | 387 | 0 | 387 | 0 | 1343.375 | 743.422 |
| | 135-145 | bimodal | 394 | 394 | 0 | 394 | 0 | 13.1387 | 1.227 |

## 3.4 Summary

ACO-BS is proposed in this chapter, and its effectiveness is examined with the benchmark instances and random instances.

A method based on the priority rule is used to obtain a solution to initialize the best-so-far solution. With priority rule, 63.20% of the total benchmark instances can reach the optimal results. After ten runs (360 CPU time seconds for each run) of ACO-BS, 95.54% of the total benchmark instances can reach the optimal solutions. We can conclude that the algorithm of ACO-BS is good at solving SALBP-I.

With the development of the manufacturing industry and the transformation from mass production to customization, obtaining good solutions for assembly line balancing of complex products within an acceptable period is worth to explore. In order to further examine the performance of ACO-BS on more complicated instances, we generate large-scale SALBP-I instances randomly. OS and TV are chosen to measure the complexity of random instances. Compared with solutions obtained by priority rule or ACO, the solutions obtained by ACO-BS are significantly better. This shows that ACO-BS is efficient to solve small-scale instances as well as large-scale instances. Therefore, ACO-BS is a promising tool for solving SALBP-I, especially the complicated instances.

# Chapter 4. Smart Control of the Assembly Process with a Fuzzy Control System in the Context of Industry 4.0

In this chapter, a real-time control problem in the context of Industry 4.0 is stated, and a fuzzy control system is proposed to monitor and control an assembly line in real time. Finally, the impact of the fuzzy control system on the performance of the assembly line, in terms of average buffer level, starvation ratio and blockage ratio, is examined by numerical experiments.

## 4.1 Problem Statements and Assumptions

Workload balance of an assembly line is affected by uncertainties, such as changes of processing times, blockage, starvation and maintenance. The existing literature considering the disruptions examines the robust solutions or re-balancing solutions, however, there is limited research on the real-time reactions to the disruptions due to the lack of real-time information in the past. In the era of Industry 4.0, real-time data of the assembly process can be obtained by sensors, and the assembly line is more reconfigurable. Nevertheless, the novel context brought by Industry 4.0 tends to be ignored. Thus, the novel problem setting of ALBP with useful information monitored is introduced next.

Figure 4-1 shows the structure of an assembly line consisting of $M$ workstations and $M-1$ buffers between these workstations, and this is the assembly line considered in this study. $B_i$ denotes buffer $i$ ($i = 0,1,2,...,M$).

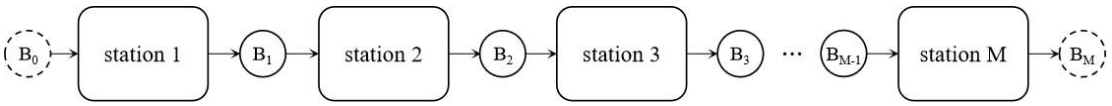


**Figure 4-1:** An assembly line with *M* workstations

The whole assembly process is divided into production cycles by assembly line re-balancing. After the preparation for assembly line re-balancing is completed, a new production cycle will begin. Let $P_i$ denote the cumulative production in the $i^{th}$ production cycle. Figures 4-2 and 4-3 are used to illustrate the related variables for two scenarios. Let $P$, $D$ and $T_d$ denote the cumulative production, the demand quantity of products and the delivery time left for the current production cycle, respectively. Let $t_i$ ($i > 0$, $t_0 = 0$) denote the time when the preparation for the $i^{th}$ re-balancing is completed and a new production cycle is about to begin. At $t_i$, $D$ and $T_d$ are updated, and $P$ will be initialized to be 0.

At first, $D = D_0$ and $T_d = T_{total}$. If there is no re-balancing since $t_0$, $D$ and $T_d$ will not be updated, and $P$ is the cumulative production since $t_0$ (see Figure 4-2). Let $T_i$ ($i > 0$) denote the duration between $t_{i\text{-}1}$ and the beginning of the $i^{th}$

re-balancing, and let $T_{r_i}$ denote the preparation time for the $i^{th}$ re-balancing. If assembly line re-balancing has been implemented for $k$ times ($k > 0$),

$$D = D_0 - \sum_{i=1}^{k} P_k \,, \tag{4-1}$$

$$T_d = T_{total} - \sum_{i=1}^{k} \left( T_i + T_{r_i} \right), \tag{4-2}$$

and $P$ is the cumulative production since $t_k$ (see Figure 4-3).

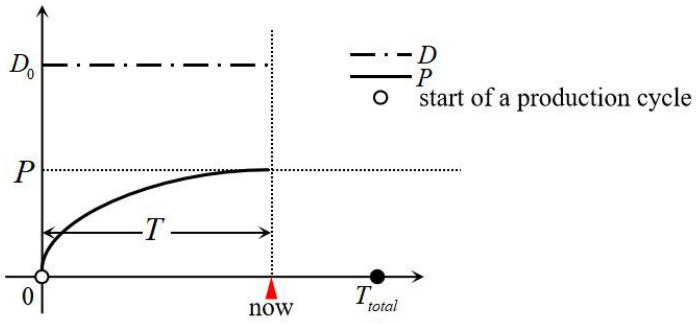


**Figure 4-2:** An illustration of variables when no re-balancing has been conducted

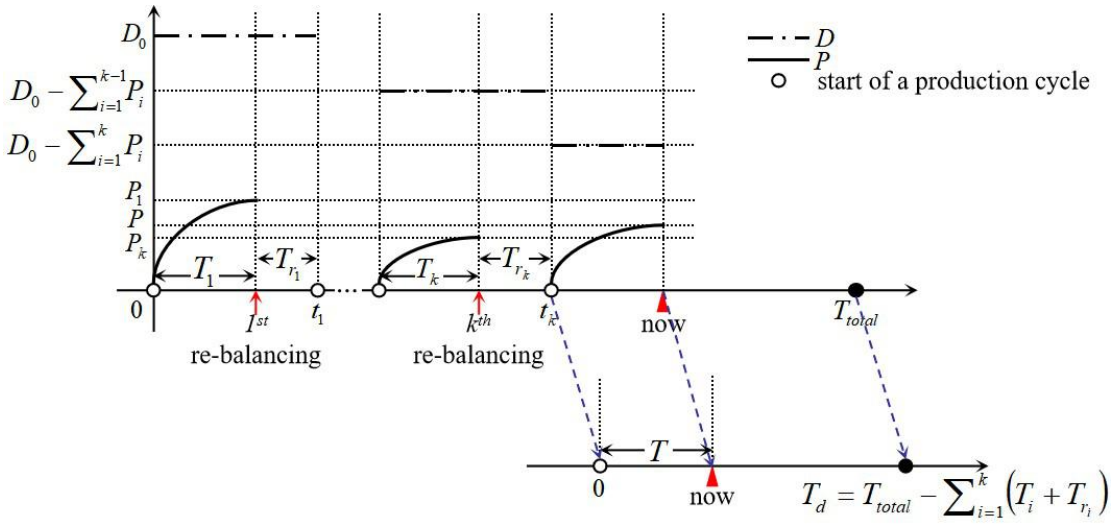


**Figure 4-3:** An illustration of variables after the $k^{th}$ re-balancing

The main assumptions in this study are presented as follows:

(1) Let $pr_i$ denote the production rate of workstation $i$ ( $pr_i \le pr_{\max}^i$ ), and $pr_{\max}^i$ is the largest production rate of workstation $i$ . It is assumed that $pr_i$ can be adjusted between 0 and $pr_{\max}^i$ . Thus, workstation $i$ can operate at a minimum processing time $\frac{1}{pr_{\max}^i}$.

(2) It is assumed that $B_i (1 \le i \le M-1)$ has finite capacity. $B_0$ is assumed to be an infinite source of raw material so that station 1 is never starved. $B_M$ is assumed to have infinite storage capacity so that station $M$ is never blocked.

(3) The financial cost for assembly line re-balancing is assumed to be negligible, but the time cost due to the preparation for all the re-assigned tasks is considered.

(4) Machines are assumed to break down and be repaired randomly with different probabilities. The failure rate of machines is $\lambda$ , and the repair rate of machines is $\mu$ . The uptimes and downtimes of machines are assumed to follow exponential distributions with the means of $\frac{1}{\lambda}$ and $\frac{1}{\mu}$, respectively.

(5) The automated assembly line, where more automated technologies are adopted and the assembly tasks are done by robots rather than human workers, is considered in this study. According to Li (2017), there are two characteristics of the automated assembly line in which robots are the primary agents in assembly tasks: learning automata, and control architecture and collaborative learning. The first characteristic means manufacturing techniques can be refined based on

the prior manufacturing experience and this leads to task time reductions. As to the second characteristic, task time reductions of one robot can be realized by the other robots since the learned skills can be transferred to the other robots. In this study, the above learning effect is assumed to be possible. Due to the learning effect, the processing ability of each workstation will be improved. The processing time reductions of all tasks are assumed to occur simultaneously. The task time reduction rate of the task $i$ is defined in equation (4-3):

$$r_i = \frac{t_i - t_i'}{t_i} \tag{4-3}$$

where $t_i$ is the initial processing time of the task $i$, and $t_i'$ is the decreased processing time of task $i$ after a learning effect occurs.

(6) Blockage of workstation $i$ is assumed to occur when it finishes one workpiece, but $B_i$ is full. Workstation $i$ will be blocked until there is space in $B_i$. Starvation of workstation $i$ is assumed to occur when it is idle, but $B_{i-1}$ is empty. The starvation will end when there is inventory in $B_{i-1}$.

## 4.2 Fuzzy Control Model

Due to the randomness and nonlinearity of disruptions to an assembly line, it is challenging to design a control system for an assembly line, and it is difficult to develop a mathematical model for such a control system. However, fuzzy controllers

can incorporate data and human knowledge in a systematic way, and it is efficient for decision-making. In the context of Industry 4.0, each workstation can exchange production information with the upstream and downstream workstations and get access to the information of buffer levels. Thus, in this study, for each workstation, one fuzzy controller is developed to analyse useful information for its production and determine whether and how it should adjust the production rate. As a result, quicker response because of the decentralized decision-making and better collaboration of workstations due to the more transparent information can be expected. Besides, in order to determine the trigger point of assembly line re-balancing, another fuzzy controller needs to be developed to analyse the global information. Therefore, with the novel context brought by Industry 4.0 considered, a fuzzy control system composed of fuzzy controllers will be used to monitor and control the assembly process and make real-time adjustments.

Figure 4-4 shows the fuzzy control system for the assembly line shown in Figure 4-1. $FC_i$ denotes the $i^{th}$ fuzzy controller. There are two types of fuzzy controllers in the fuzzy control system. $FC_1$ is used to deal with global information and determine whether to re-balance the assembly line. Fuzzy controllers $FC_2$ to $FC_{M+1}$ are used to process the local information and make decisions on how to adjust the production rate of each workstation when re-balancing is not needed.

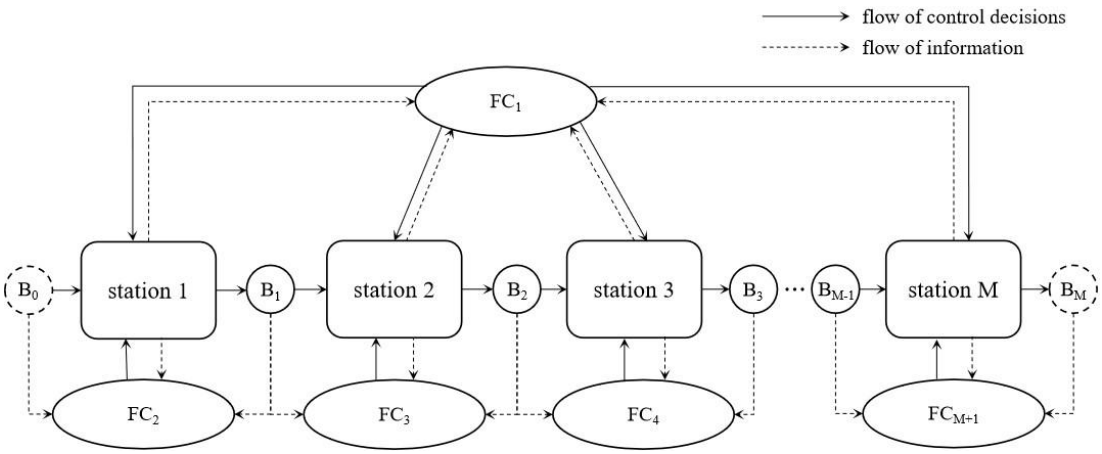


**Figure 4-4:** The framework of the fuzzy control system

A fuzzy controller is an inference system to mimic human thinking, which consists of a fuzzifier, some fuzzy IF-THEN rules, a fuzzy inference engine and a defuzzifier (AI-Ebbini et al., 2016).

### 4.2.1 Fuzzification

In the fuzzification process, the input data set is converted into fuzzy sets by fuzzy membership functions. Each of the fuzzy subsets represents one linguistic term that allows its members to have different grades of membership.

### 4.2.2 Inputs and the Output of Type 1 Fuzzy Controller

Two factors, which are important in determining whether to re-balance the assembly line, are the inputs of the fuzzy controller, and the output is the necessity of

assembly line re-balancing.

(1) Urgency of the assembly job, $urg$

To keep the stability of an assembly line and prevent overacting to disruptions, the urgency of the assembly job is considered before a re-balancing decision is made. The assembly job urgency is defined in equation (4-4) as follows:

$$urg = \begin{cases} \dfrac{D \cdot \dfrac{T}{T_d} - P}{D \cdot \dfrac{T}{T_d}}, & D \cdot \dfrac{T}{T_d} \geq \dfrac{P}{2} \\ -1, & D \cdot \dfrac{T}{T_d} < \dfrac{P}{2} \end{cases} \tag{4-4}$$

where $T$ is the assembly time used in the current production cycle. $D \cdot \dfrac{T}{T_d}$ is the amount of production that should be finished at present, and if it is larger than $P$, there is a risk that the demand cannot be satisfied. $urg$ ranges from -1 to 1, and the fuzzy term set is {very small, small, medium, large, very large}.

(2) Time cost to re-balance the assembly line, $T_c$

Assembly line re-balancing should not be considered when there is not a large possibility that the production after re-balancing is more than that without re-balancing, even though it is quite urgent to increase production. The cost of re-balancing is defined in equation (4-5):

$$T_c = \frac{pr_{ini} \cdot \left(T_{total} - T - T_m\right) - pr_{new} \cdot \left(T_{total} - T - T_r - \frac{1}{pr_{new}} \cdot n_{avail}\right)}{pr_{\max} \cdot \left(T_{total} - T\right)} \tag{4-5}$$

where $T_m$ denotes the maintenance time of the assembly line and is defined as

$$T_m = \max_{i \in T_A} \left\{T_{m_i}\right\}. \tag{4-6}$$

$T_A$ is a set consisting of the stations involved in the current assembly plan (some workstations may be closed for maintenance), and $T_{m_i}$ is the maintenance time needed by station $i$. $T_r$ designates the preparation time for re-balancing the assembly line and is defined as the sum of the preparation times of all the re-assigned tasks. The term $pr_{ini}$ denotes the largest production ability according to the current assembly plan, defined as

$$pr_{ini} = \min_{i \in T_A} \left\{pr_{\max}^{i}\right\}. \tag{4-7}$$

Besides, $pr_{new}$ is the production rate after re-balancing, and $pr_{\max}$ designates the largest production rate that can be achieved by the current assembly line with all stations operative. $n_{avail}$ denotes the number of stations used in the re-balancing plan. $T_r + \frac{1}{pr_{new}} \cdot n_{avail}$ denotes the shortest time from the beginning of the preparation for the re-balancing to obtaining one finished product. A small $T_c$ less than 0 indicates that there is some possibility that the production, after re-balancing during the simulation time remaining, is increased. The smaller $T_c$ is, the larger the possibility becomes. The fuzzy term set is {very small, small, medium, large, very large}.

(3) Output variable: the necessity of re-balancing, $N$

The set of fuzzy terms is {very small, small, medium, large, very large}. The necessity is between 0% and 100%, and the fuzzy terms of 'very small' and 'very large' indicate the range from 0% to 100%. Assembly line re-balancing is conducted when $N$ is larger than a predetermined threshold.

### 4.2.3 Inputs and the Output of Type 2 Fuzzy Controller

Three factors, which affect decision making on the production rate adjustment of a workstation, are the inputs of type 2 fuzzy controllers, and the output is the production rate adjustment of the corresponding workstation.

(1) Upstream buffer level $BL_{i\text{-}1}$ and downstream buffer level $BL_i$ of station $i$

The upstream buffer of station $i$ is $B_{i-1}$, and the downstream buffer is $B_i$. The buffer level of buffer $i$ is defined in equation (4-8):

$$BL_i = \frac{w_i}{C_i}, \quad i = 0,1,2,...,M \tag{4-8}$$

where $w_i$ is the inventory of buffer $i$, and $C_i$ is the capacity of buffer $i$. Buffer levels range from 0 to 1. The set of fuzzy terms is {very small, small, medium, large, very large}.

(2) Production surplus rate, $S_i$

The third factor that affects the production rate adjustment is the production surplus rate, which is defined in equation (4-9) as follows:

$$S_i = \begin{cases} \dfrac{\dfrac{P_{S_i}}{T} - d}{pr^i_{\max} - d}, & P_{S_i} \geq d \cdot T \\ \dfrac{\dfrac{P_{S_i}}{T} - d}{d}, & P_{S_i} < d \cdot T \end{cases} \tag{4-9}$$

where $P_{S_i}$ is the cumulative production of workstation $i$. $d$ denotes the demand production rate that is updated at the beginning of each production cycle. The production rate of workstation $i$ should be around $d$ to satisfy the demand.

If the percentage of the operative time of a workstation is $\dfrac{\frac{1}{\lambda}}{\frac{1}{\lambda}+\frac{1}{\mu}}$, the operative time of the whole assembly line is no larger than $T_{total} \cdot \dfrac{\mu}{\lambda+\mu}$. Thus, a relatively safe demand rate $d$ at $t_0$ should be at least $\dfrac{D_0}{T_{total} \cdot \dfrac{\mu}{\lambda+\mu}}$. In addition, the re-balancing decision is made only when there is a risk that the demand cannot be satisfied, thus, after re-balancing, $d$ is set to be a small value (it is $10^{-6}$ in this study) smaller than $pr_{ini}$ in order to make up for the production loss due to assembly plan modification as soon as possible.

The surplus rate ranges from -1 to 1. When it is larger than 0, there is more inventory; Otherwise, there are backlogs. The fuzzy set of production surplus rate is {very small, small, medium, large, very large}.

(3) Output variable: production rate adjustment of workstation $i$, $adj\_pr_i$

The output $adj\_pr_i$ is the adjustment suggestion for $pr_i$, and ranges from -1 to 1. The fuzzy terms set is {very small, small, medium, large, very large}. When $adj\_pr_i$ is larger than 0, $pr_i$ should be increased toward $pr_{\max}^i$. Otherwise, $pr_i$ should be decreased toward 0. The production rate after adjustment is defined in equation (4-10) as follows:

$$pr_i' = \begin{cases} pr_i + adj\_pr_i \cdot \left(pr_{\max}^i - pr_i\right), & adj\_pr_i \geq 0 \\ pr_i + adj\_pr_i \cdot \left(pr_i - 0\right), & adj\_pr_i < 0 \end{cases}. \quad (4\text{-}10)$$

### 4.2.4 Fuzzy Rules

Fuzzy rules are the base of the fuzzy inference engine, and they can be utilized to make decisions and generate control actions. The rules are in the form of if-then statements. There are 25 fuzzy rules for type 1 fuzzy controller (see Table 4-1). When there is a risk that the demand cannot be satisfied and $T_c$ is smaller than 0, assembly line re-balancing may take place. Otherwise, re-balancing will not take place. Therefore, assembly line re-balancing takes place only when $N$ is large

enough, so as to adjust the assembly line in time and prevent overreaction.

**Table 4-1:** Fuzzy rule base for the type 1 fuzzy controller

| *urg* | $T_c$ | | | | |
|---|---|---|---|---|---|
| | VS | S | ME | L | VL |
| VS | S | S | S | VS | VS |
| S | L | ME | ME | VS | VS |
| ME | VL | VL | L | VS | VS |
| L | VL | VL | L | S | VS |
| VL | VL | VL | L | S | VS |

Note: VS, S, ME, L and VL denote very small, small, medium, large and very large, respectively.

There are 125 rules for the type 2 fuzzy controllers (see Table 4-2). If there is no risk of starvation or blockage, $pr_i$ should be adjusted mainly based on $S_i$. Otherwise, since the adverse impacts of starvation and blockage propagate throughout the assembly line, $pr_i$ should be adjusted to eliminate starvation and blockage.

**Table 4-2:** Fuzzy rule base for type 2 fuzzy controllers

| $S_i$=VS | | | | | |
|---|---|---|---|---|---|
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | L | S | S | S | VS |
| S | VL | VL | VL | L | VS |
| ME | VL | VL | VL | VL | S |
| L | VL | VL | VL | VL | S |
| VL | VL | VL | VL | VL | ME |
| $S_i$=S | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | L | S | S | VS | VS |
| S | VL | VL | L | L | VS |
| ME | VL | VL | VL | L | S |
| L | VL | VL | VL | L | S |
| VL | VL | VL | VL | L | ME |
| $S_i$=ME | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | ME | S | VS | VS | VS |
| S | VL | ME | ME | ME | VS |
| ME | VL | L | ME | ME | S |
| L | VL | VL | ME | ME | S |
| VL | VL | VL | L | ME | ME |
| $S_i$=L | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | S | VS | VS | VS | VS |
| S | ME | S | S | S | VS |
| ME | ME | S | S | S | VS |
| L | L | ME | S | S | S |
| VL | VL | L | L | ME | ME |
| $S_i$=VL | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | VS | VS | VS | VS | VS |
| S | S | VS | VS | VS | VS |
| ME | ME | VS | VS | VS | VS |
| L | L | ME | VS | VS | VS |
| VL | VL | L | L | ME | ME |

### 4.2.5 Defuzzification

The output generated by the fuzzy inference engine is a set of fuzzy membership values (Nakandala et al., 2013). Fuzziness helps rule evaluation during the intermediate steps. However, the final desired output is generally a single number. Therefore, all the outputs are transferred into the final crisp value by a widely used defuzzification method: the centroid method, which assesses the center of gravity of the possible distribution of the fuzzy output (Nakandala et al., 2013), and is defined in equation (4-11) as follows (AI-Ebbini et al., 2016):

$$y = \frac{\sum_{r=1}^{R_l} A^{\alpha r} C_{A^{\alpha r}}}{\sum_{r=1}^{R_l} A^{\alpha r}} \tag{4-11}$$

where $A^{\alpha r}$ denotes the area of consequent's fuzzy subset, which is obtained by $\alpha$ membership determined by the $r^{th}$ rule. $C_{A^{\alpha r}}$ is the center of area $A^{\alpha r}$. $R_l$ designates the number of fuzzy rules.

## 4.3 Numerical Results

### 4.3.1 Settings of Experiments

The assembly line used to test the fuzzy control system is defined by KILBRID (45 tasks), and the task times and precedence relationship information can be found in the SALBP data sets shown on https://assembly-line-balancing.de/salbp/benchmark-data-sets-1993/. Table 4-3

shows the original information of the task times and the precedence graph of the chosen instance. However, the total simulation time is set to be 1000, and in order to be consistent with this setting, all the task times in this study are made to be 100 times smaller so that $t_{\min}$, $t_{\max}$ and $t_{avg}$ become to 0.03, 0.55 and 0.12267. The first column is the total number of tasks, and the second to the fifth columns are the minimal, maximal, total and average task times, respectively. The sixth column shows the order strength of the precedence graph, which is calculated by the ratio of the number of all precedence relations to $n \cdot (n-1)$. $TV$ is the time variability ratio defined by $\frac{t_{\max}}{t_{\min}}$, and $div$ and $conv$ are the divergence degree and convergence degree of the precedence graph.

**Table 4-3:** Original information of KILBRID

| $n$ | $t_{\min}$ | $t_{\max}$ | $t_{sum}$ | $t_{avg}$ | $OS$ | $TV$ | $div$ | $conv$ |
|---|---|---|---|---|---|---|---|---|
| 45 | 3 | 55 | 552 | 12.267 | 44.550 | 18.330 | 0.670 | 0.690 |

There are 8 workstations in total, and the assembly line is balanced initially with all tasks assigned to these 8 workstations. Based on the fuzzy control system, the assembly line will be re-balanced when necessary. Although we only chose one instance from the SALBP data set, different characteristics of machine states and different production rates required by the demand are considered to examine the effectiveness of the proposed system. For each experiment, there are 10 random runs,

and different seeds are used to guarantee the independent states of the machines.

Since the problem defined in this study is novel, there are no benchmark instances in the existing literature. To model different levels of information transparency and make comparisons, we set three kinds of comparative assembly lines. Table 4-4 shows the characteristics of the four assembly lines discussed in this study. The length of the time from the breakdown of the machine in station $i$ to the recognition of the breakdown follows a normal distribution with a mean which is a multiple of $T_{S_i}$, which denotes the sum of the processing times of tasks assigned to station $i$. Since the real-time information is not only collected but also analyzed, the assembly line with the proposed fuzzy system ($AS_1$) achieves higher-level information transparency compared with the other three assembly lines. From $AS_2$ to $AS_4$, the level of information transparency decreases.

**Table 4-4:** Assembly lines to be compared in this study

| Name | With a fuzzy system | Production rate | With a maintenance delay | Description of the delay when applicable |
|---|---|---|---|---|
| $AS_1$ | Yes | $0 \le pr_i \le pr_{\max}^i$ | No | - |
| $AS_2$ | No | $pr_i = pr_{\max}^i$ | No | - |
| $AS_3$ | No | $pr_i = pr_{\max}^i$ | Yes, and it follows a normal distribution | $N(5T_{S_i},3)$ |
| $AS_4$ | No | $pr_i = pr_{\max}^i$ | Yes, and it follows a normal distribution | $N(10T_{S_i},5)$ |

Figure 4-5 shows the assembly process with the proposed fuzzy system. If it is decided to re-balance the assembly line by $FC_1$, then the re-balancing plan is prepared and undertaken. Otherwise, $FC_2$ to $FC_9$ are activated to adjust the production rate of each workstation.

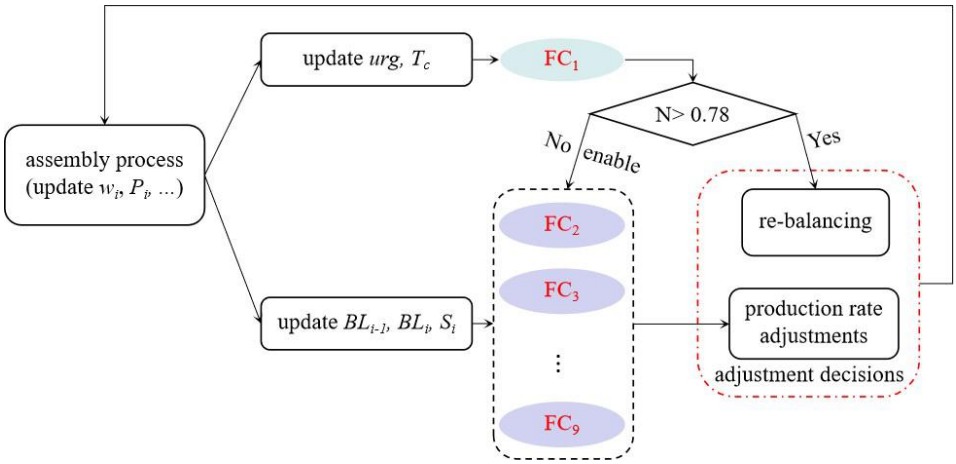


**Figure 4-5:** Assembly process with the proposed fuzzy system

As solution generation for the assembly line re-balancing problem is not the main contribution of this study, the algorithm of ACO-BS developed by Huo et al. (2018) to solve ALBP is used iteratively to generate the re-balancing solution, given the number of available workstations and the initial assembly plan. The flow chart of this method is shown in Figure 4-6. $LB$ denotes the lower bound of cycle time, given the number of available workstations $m^*$, and is initialized to be $\max\left\{t_{\max}, \frac{t_{sum}}{m^*}\right\}$. $tr_i$ denotes the sum of the preparation times of all the reassigned tasks in solution $s_i$. The given cycle time is increased by 1 step by step to find all

the possible cycle times, with all the available stations utilized. Finally, the re-balancing solution is obtained by considering both the cycle time and the corresponding preparation time for re-balancing.

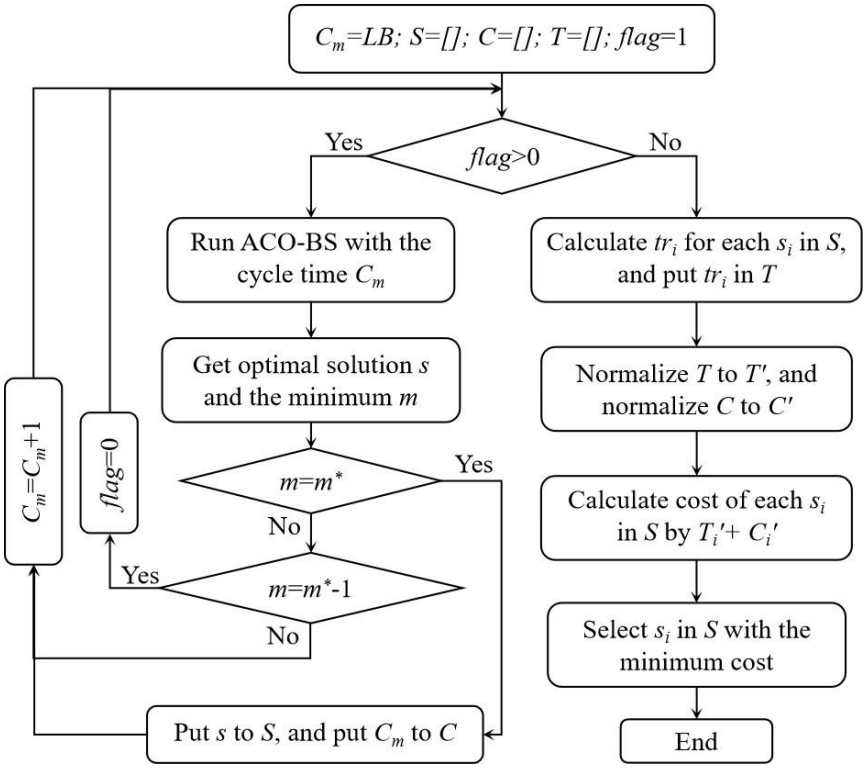


**Figure 4-6:** Flow chart of the method to generate the re-balancing solution

### 4.3.2 Results

The numerical experiments were done with Simulink in MATLAB (R2016a). The learning effect was set to occur at a simulation time of 500. The task time reduction rate $r_i$ was generated following the uniform distribution in the interval (0, 0.1), with mean 0.050, and standard deviation 0.027. For comparison reasons, the same

setting related to the learning effect was used for all the cases. Additionally, the capacity of each buffer between workstations was set to be 25 units, and the preparation time for each task was set to be 0.01. The experiment stops when the simulation time is used up or the demand is satisfied.

There are 5 different combinations of $\lambda$ and $\mu$, 3 levels of $d_0$ (measured by $\frac{D_0}{T_{total}}$). For each assembly line, there are 15 different experiments to conduct. For each experiment, the means and the standard deviations of the ten random runs are shown in Tables 4-5 and 4-6, and six indicators are shown, that is, blockage ratio (ratio of the duration of blockage to the total simulation time), average buffer level, starvation ratio (ratio of the duration of starvation to the total simulation time), simulation time used, total production and number of times of assembly line re-balancing.

**Table 4-5:** Results of the numerical experiments (Part 1)

| λ | μ | μ/λ | No. | Mean | $d_0$=1 | | | | | | $d_0$=0.7 | | | | | | $d_0$=0.3 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | and std | b_r | BL | st_r | t | P | no_r | b_r | BL | st_r | t | P | no_r | b_r | BL | st_r | t | P | no_r |
| 0.1 | 0.5 | 5 | $AS_1$ | mean | 0.000 | 0.347 | 0.047 | 949.409 | 1000.000 | 0.000 | 0.000 | 0.123 | 0.062 | 803.144 | 700.000 | 0.000 | 0.000 | 0.089 | 0.031 | 559.940 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.064 | 0.008 | 21.739 | 0.000 | 0.000 | 0.000 | 0.018 | 0.012 | 9.844 | 0.000 | 0.000 | 0.000 | 0.002 | 0.004 | 5.607 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.016 | 0.362 | 0.071 | 918.023 | 1000.000 | 0.000 | 0.011 | 0.327 | 0.083 | 671.585 | 700.000 | 0.000 | 0.007 | 0.244 | 0.119 | 315.004 | 300.000 | 0.000 |
| | | | | std | 0.010 | 0.085 | 0.023 | 22.647 | 0.000 | 0.000 | 0.006 | 0.085 | 0.026 | 17.781 | 0.000 | 0.000 | 0.006 | 0.065 | 0.023 | 15.842 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.053 | 0.428 | 0.118 | 1000.000 | 725.900 | 0.000 | 0.053 | 0.428 | 0.117 | 964.831 | 699.600 | 0.000 | 0.039 | 0.363 | 0.153 | 466.546 | 300.000 | 0.000 |
| | | | | std | 0.017 | 0.075 | 0.035 | 0.000 | 31.370 | 0.000 | 0.017 | 0.078 | 0.029 | 22.982 | 1.265 | 0.000 | 0.020 | 0.086 | 0.032 | 30.299 | 0.000 | 0.000 |
| | | | $AS_4$ | mean | 0.087 | 0.444 | 0.155 | 1000.000 | 512.900 | 0.000 | 0.079 | 0.430 | 0.153 | 1000.000 | 525.000 | 0.000 | 0.063 | 0.384 | 0.184 | 639.721 | 300.000 | 0.000 |
| | | | | std | 0.032 | 0.077 | 0.036 | 0.000 | 39.159 | 0.000 | 0.023 | 0.057 | 0.032 | 0.000 | 27.793 | 0.000 | 0.025 | 0.062 | 0.026 | 29.157 | 0.000 | 0.000 |
| 0.01 | 0.1 | 10 | $AS_1$ | mean | 0.003 | 0.285 | 0.090 | 967.039 | 999.200 | 0.333 | 0.000 | 0.126 | 0.062 | 863.241 | 700.000 | 0.000 | 0.000 | 0.095 | 0.041 | 594.412 | 300.000 | 0.000 |
| | | | | std | 0.011 | 0.051 | 0.025 | 34.312 | 1.751 | 0.707 | 0.000 | 0.016 | 0.028 | 29.816 | 0.000 | 0.000 | 0.000 | 0.003 | 0.016 | 16.654 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.067 | 0.435 | 0.098 | 911.229 | 1000.000 | 0.000 | 0.066 | 0.429 | 0.103 | 652.414 | 700.000 | 0.000 | 0.032 | 0.307 | 0.136 | 295.451 | 300.000 | 0.000 |
| | | | | std | 0.024 | 0.089 | 0.032 | 44.328 | 0.000 | 0.000 | 0.022 | 0.080 | 0.029 | 26.334 | 0.000 | 0.000 | 0.036 | 0.123 | 0.036 | 25.368 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.092 | 0.462 | 0.120 | 976.718 | 982.600 | 0.000 | 0.092 | 0.440 | 0.134 | 724.977 | 700.000 | 0.000 | 0.054 | 0.332 | 0.160 | 325.486 | 300.000 | 0.000 |
| | | | | std | 0.029 | 0.101 | 0.036 | 27.294 | 33.662 | 0.000 | 0.030 | 0.094 | 0.034 | 37.823 | 0.000 | 0.000 | 0.050 | 0.139 | 0.038 | 37.421 | 0.000 | 0.000 |
| | | | $AS_4$ | mean | 0.114 | 0.460 | 0.154 | 1000.000 | 904.400 | 0.000 | 0.111 | 0.454 | 0.145 | 777.045 | 700.000 | 0.000 | 0.088 | 0.367 | 0.186 | 372.435 | 300.000 | 0.000 |
| | | | | std | 0.023 | 0.078 | 0.042 | 0.000 | 59.024 | 0.000 | 0.029 | 0.074 | 0.032 | 52.693 | 0.000 | 0.000 | 0.073 | 0.140 | 0.039 | 47.908 | 0.000 | 0.000 |
| 0.01 | 0.5 | 50 | $AS_1$ | mean | 0.000 | 0.082 | 0.017 | 924.738 | 1000.000 | 0.000 | 0.000 | 0.088 | 0.017 | 835.108 | 700.000 | 0.000 | 0.000 | 0.085 | 0.022 | 590.860 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.001 | 0.001 | 2.036 | 0.000 | 0.000 | 0.000 | 0.001 | 0.001 | 3.102 | 0.000 | 0.000 | 0.000 | 0.000 | 0.001 | 0.544 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.000 | 0.138 | 0.029 | 727.136 | 1000.000 | 0.000 | 0.000 | 0.107 | 0.031 | 519.023 | 700.000 | 0.000 | 0.000 | 0.054 | 0.045 | 228.329 | 300.000 | 0.000 |
| | | | | std | 0.001 | 0.027 | 0.006 | 3.864 | 0.000 | 0.000 | 0.000 | 0.028 | 0.006 | 6.212 | 0.000 | 0.000 | 0.000 | 0.027 | 0.014 | 5.763 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.011 | 0.324 | 0.051 | 787.344 | 1000.000 | 0.000 | 0.009 | 0.265 | 0.067 | 572.001 | 700.000 | 0.000 | 0.001 | 0.143 | 0.088 | 254.766 | 300.000 | 0.000 |
| | | | | std | 0.005 | 0.046 | 0.009 | 15.836 | 0.000 | 0.000 | 0.006 | 0.055 | 0.016 | 14.634 | 0.000 | 0.000 | 0.002 | 0.042 | 0.022 | 8.706 | 0.000 | 0.000 |
| | | | $AS_4$ | mean | 0.041 | 0.408 | 0.077 | 861.475 | 1000.000 | 0.000 | 0.031 | 0.350 | 0.089 | 621.326 | 700.000 | 0.000 | 0.014 | 0.237 | 0.117 | 286.753 | 300.000 | 0.000 |
| | | | | std | 0.015 | 0.056 | 0.015 | 22.755 | 0.000 | 0.000 | 0.013 | 0.072 | 0.020 | 18.745 | 0.000 | 0.000 | 0.011 | 0.065 | 0.027 | 12.734 | 0.000 | 0.000 |

**Table 4-6:** Results of the numerical experiments (Part 2)

| λ | μ | μ/λ | No. | Mean | $d_0$=1 | | | | | | $d_0$=0.7 | | | | | | $d_0$=0.3 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | and std | b_r | BL | st_r | t | P | no_r | b_r | BL | st_r | t | P | no_r | b_r | BL | st_r | t | P | no_r |
| 0.001 | 0.1 | 100 | $AS_1$ | mean | 0.000 | 0.093 | 0.039 | 938.273 | 1000.000 | 0.000 | 0.000 | 0.091 | 0.028 | 840.511 | 700.000 | 0.000 | 0.000 | 0.086 | 0.024 | 594.759 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.015 | 0.022 | 11.883 | 0.000 | 0.000 | 0.000 | 0.005 | 0.014 | 2.664 | 0.000 | 0.000 | 0.000 | 0.002 | 0.008 | 4.481 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.011 | 0.203 | 0.048 | 750.031 | 1000.000 | 0.000 | 0.013 | 0.163 | 0.051 | 540.471 | 700.000 | 0.000 | 0.012 | 0.099 | 0.070 | 240.230 | 300.000 | 0.000 |
| | | | | std | 0.010 | 0.097 | 0.024 | 29.281 | 0.000 | 0.000 | 0.013 | 0.080 | 0.034 | 31.280 | 0.000 | 0.000 | 0.022 | 0.088 | 0.066 | 29.834 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.015 | 0.238 | 0.054 | 763.129 | 1000.000 | 0.000 | 0.016 | 0.182 | 0.059 | 553.254 | 700.000 | 0.000 | 0.014 | 0.113 | 0.078 | 243.959 | 300.000 | 0.000 |
| | | | | std | 0.013 | 0.089 | 0.028 | 31.488 | 0.000 | 0.000 | 0.015 | 0.070 | 0.036 | 32.275 | 0.000 | 0.000 | 0.023 | 0.084 | 0.066 | 29.451 | 0.000 | 0.000 |
| | | | $AS_4$ | mean | 0.023 | 0.280 | 0.060 | 781.053 | 1000.000 | 0.000 | 0.021 | 0.200 | 0.065 | 562.290 | 700.000 | 0.000 | 0.018 | 0.128 | 0.089 | 251.713 | 300.000 | 0.000 |
| | | | | std | 0.015 | 0.092 | 0.033 | 34.417 | 0.000 | 0.000 | 0.017 | 0.067 | 0.038 | 32.991 | 0.000 | 0.000 | 0.031 | 0.084 | 0.077 | 36.822 | 0.000 | 0.000 |
| 0.001 | 0.3 | 300 | $AS_1$ | mean | 0.000 | 0.081 | 0.017 | 933.622 | 1000.000 | 0.000 | 0.000 | 0.087 | 0.017 | 842.267 | 700.000 | 0.000 | 0.000 | 0.085 | 0.022 | 594.838 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.000 | 0.001 | 2.134 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.353 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.095 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.000 | 0.036 | 0.013 | 697.191 | 1000.000 | 0.000 | 0.000 | 0.026 | 0.012 | 494.844 | 700.000 | 0.000 | 0.000 | 0.013 | 0.018 | 214.962 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.021 | 0.003 | 3.281 | 0.000 | 0.000 | 0.000 | 0.023 | 0.005 | 3.809 | 0.000 | 0.000 | 0.000 | 0.016 | 0.005 | 3.626 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.000 | 0.090 | 0.020 | 709.865 | 1000.000 | 0.000 | 0.000 | 0.060 | 0.019 | 503.267 | 700.000 | 0.000 | 0.000 | 0.034 | 0.028 | 219.521 | 300.000 | 0.000 |
| | | | | std | 0.000 | 0.051 | 0.004 | 6.178 | 0.000 | 0.000 | 0.000 | 0.038 | 0.007 | 5.083 | 0.000 | 0.000 | 0.000 | 0.031 | 0.011 | 6.124 | 0.000 | 0.000 |
| | | | $AS_4$ | mean | 0.001 | 0.130 | 0.024 | 721.407 | 1000.000 | 0.000 | 0.001 | 0.108 | 0.029 | 516.427 | 700.000 | 0.000 | 0.001 | 0.061 | 0.038 | 225.219 | 300.000 | 0.000 |
| | | | | std | 0.003 | 0.081 | 0.006 | 10.219 | 0.000 | 0.000 | 0.001 | 0.088 | 0.012 | 13.785 | 0.000 | 0.000 | 0.002 | 0.058 | 0.019 | 10.527 | 0.000 | 0.000 |

To show the findings of this study, the numerical results clustered by the performance indicators are shown in Figures 4-7 to 4-11. For each kind of assembly line, the results of 15 cases are shown. The demand rates are 1, 0.7 and 0.3 for cases 1 to 5, cases 6 to 10 and cases 11 to 15, respectively.

Figures 4-7 to 4-9 show the results in regards of the average starvation ratio, blockage ratio and idle ratio (the sum of starvation ratio and blockage ratio). As these three figures show, the assembly line with the proposed fuzzy system has significantly less blockage (almost no blockage for all the cases), less starvation and a higher level of machine utilization. Additionally, when machine breakdown is recognized more slowly, there is more idle time for an assembly line. Thus, there is less blockage, less starvation and a higher level of machine utilization when the level of information transparency is higher.

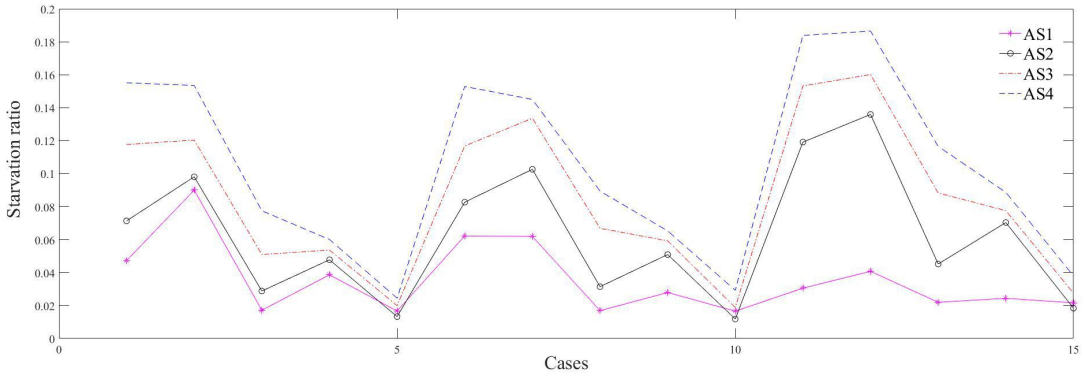


**Figure 4-7:** Average starvation ratio for the four assembly lines

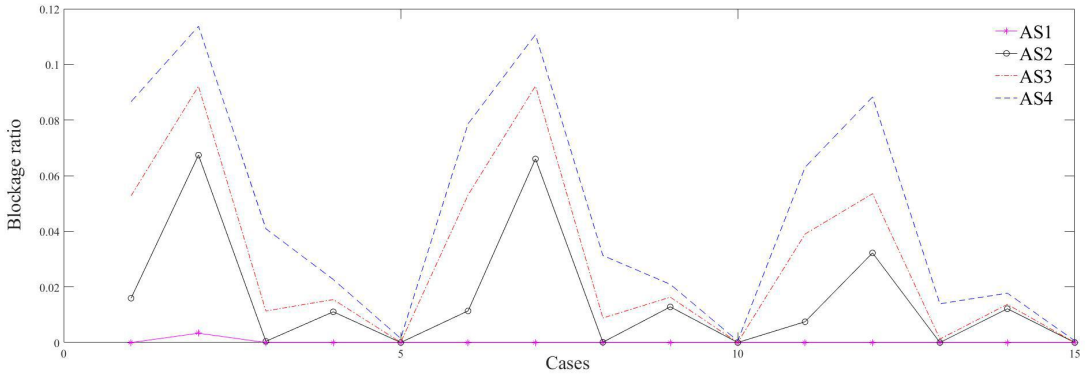


**Figure 4-8:** Average blockage ratio for the four assembly lines

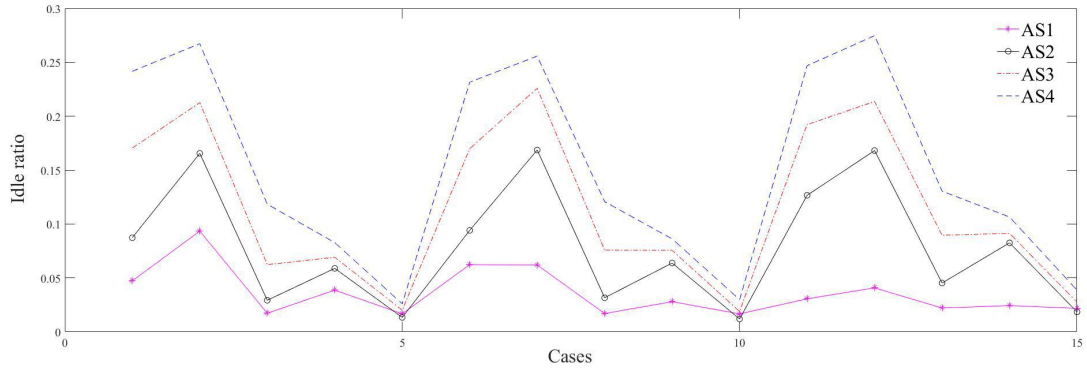


**Figure 4-9:** Average idle ratio for the four assembly lines

Figure 4-10 shows the results related to the buffer level. For $AS_1$, except for the first two cases, the buffer level is about 0.1. In general, the buffer level of $AS_1$ is significantly lower than that of the other three assembly lines, and stays at a stable level. Not surprisingly, the buffer levels of $AS_2$, $AS_3$ and $AS_4$ rank the second, third and fourth, which indicates that the buffer level decreases significantly with the increase of information transparency level. Therefore, timely recognition of the disruptions and in-time adjustment of the assembly line are helpful to keep the work-in-progress at a low level.

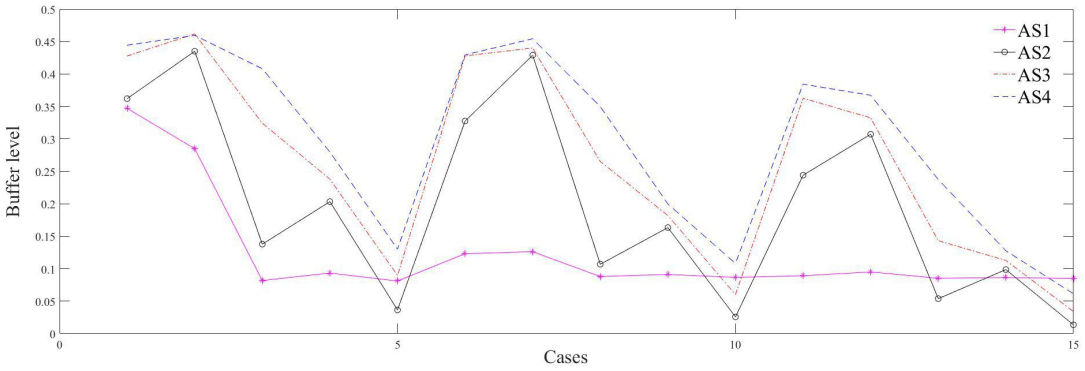


**Figure 4-10:** Average buffer level for the four assembly lines

Figure 4-11 shows the total production information for the four assembly lines. For cases 1 to 5, the demand quantity is 1000 units, and there is a large backlog for $AS_3$ and $AS_4$. For cases 6 to 10, the demand quantity is 700 units, and there is a large backlog for $AS_4$. For cases 11 to 15, outputs of the four assembly lines satisfy the demand (300 units). $AS_3$ and $AS_4$ show worse production ability, which suggests that the information transparency positively affects the production ability of an assembly line. When disruptions are recognized and dealt with in time, production losses caused by disruptions can be reduced.

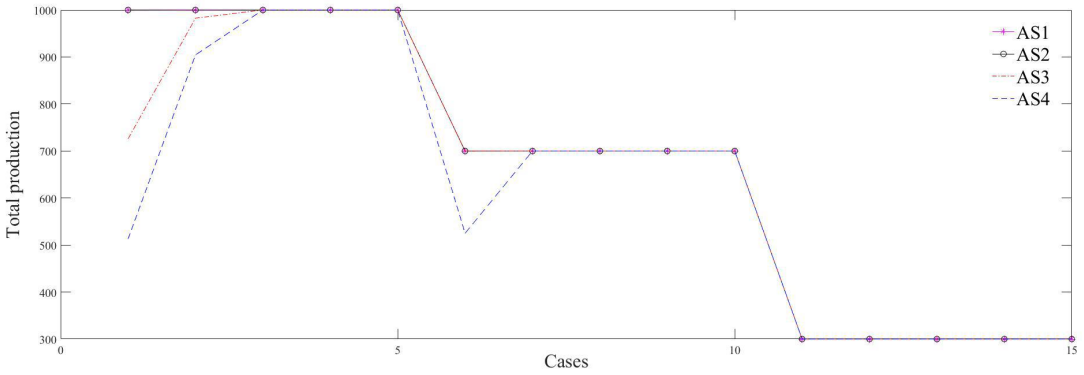


**Figure 4-11:** Average total production for the four assembly lines

Thus, the higher the information transparency level is, the better the performance of an assembly line becomes. For an assembly line with the proposed fuzzy system, real-time information can be analyzed, and in-time adjustments are undertaken accordingly. The performance is better due to the right decisions made by the proposed fuzzy system.

It can be seen from Tables 4-5 and 4-6 that assembly line re-balancing is conducted only for one case ( $\lambda = 0.01$ , $\mu = 0.1$ , $d_0 = 1$ ). For ten runs of that case, assembly line re-balancing takes place in two runs. In order to explore the impact of the preparation time of each task which is reassigned to another workstation, the preparation time is increased to 0.10 from 0.01, and the numerical results are shown in Table 4-7. This change does not affect the results of those cases where assembly line re-balancing has not taken place, thus, only the results for the two special cases are discussed further. For $AS_1$ without $FC_1$, the production rates of the workstations

are adjusted in time, but whether the assembly line should be re-balanced is not examined.

**Table 4-7:** Numerical results of the two random cases

| No. | random cases | b_r | BL | st_r | t | P | no_r |
|---|---|---|---|---|---|---|---|
| 1 | $AS_1$ | 0.000 | 0.314 | 0.074 | 999.620 | 997 | 2 |
| | $AS_1$ (with increased $T_r$) | 0.000 | 0.315 | 0.072 | 997.530 | 994 | 1 |
| | $AS_1$ (without $FC_1$) | 0.000 | 0.326 | 0.067 | 1000.000 | 986 | 0 |
| | $AS_2$ | 0.105 | 0.482 | 0.060 | 902.590 | 1000 | - |
| | $AS_3$ | 0.124 | 0.517 | 0.074 | 979.800 | 1000 | - |
| | $AS_4$ | 0.131 | 0.561 | 0.085 | 1000.000 | 982 | - |
| 2 | $AS_1$ | 0.033 | 0.294 | 0.094 | 996.870 | 1000 | 1 |
| | $AS_1$ (with increased $T_r$) | 0.033 | 0.294 | 0.094 | 996.870 | 1000 | 1 |
| | $AS_1$ (without $FC_1$) | 0.000 | 0.260 | 0.127 | 1000.000 | 963 | 0 |
| | $AS_2$ | 0.077 | 0.379 | 0.145 | 969.670 | 1000 | - |
| | $AS_3$ | 0.093 | 0.402 | 0.170 | 1000.000 | 945 | - |
| | $AS_4$ | 0.106 | 0.390 | 0.235 | 1000.000 | 799 | - |

As seen in Table 4-7, when the preparation time for each task is increased to 0.10 from 0.01, there are no significant changes in the performance of $AS_1$. For the first special case, the total production is 986 units when there are production rate adjustments but no assembly line re-balancing, and the production increases to 997 units when both production rate adjustments and re-balancing are allowed. There is a relatively large difference between the total production of $AS_4$ and the demand quantity. For $AS_2$ and $AS_3$, although the outputs can satisfy the demand within the simulation time, the buffer levels are 153.503% and 164.650% of the buffer level of $AS_1$, and the utilization of machines is significantly smaller than that for $AS_1$.

Similar rules are found for the second special case.

## 4.4 Summary

A fuzzy control system, which is composed of two types of fuzzy controllers, is proposed to deal with uncertainties of an assembly line. If the output of type 1 fuzzy controller is larger than 0.78, which is the threshold tuning by hand, the assembly line will be re-balanced. Otherwise, the production rate of each workstation will be adjusted by a type 2 fuzzy controller. In order to avoid overreacting to the uncertainties, the assembly line will be re-balanced when it is urgent to increase the total production to satisfy the demand and there is a possibility that more production is obtained after re-balancing. Meanwhile, if there is a sign of blockage or starvation, production rates will be adjusted to eliminate blockage or starvation. Otherwise, the production rates will be adjusted mainly based on the surplus rate. Compared with the assembly lines without the proposed fuzzy control system, the assembly line with the fuzzy control system achieves better performance. It can be concluded from the research findings that the higher the information transparency level is, the better the performance of an assembly line becomes.

# Chapter 5. A Fuzzy Control System for Assembly Line Balancing with a Three-state Degradation Process

In this chapter, ALBP in the context of Industry 4.0 is described, with the three-state degradation process of machines considered. Then, a fuzzy control system is proposed to deal with uncertainties involving the changes in machines' health states. The structures of the two types of fuzzy controllers in the fuzzy control system are presented, and the effectiveness of the proposed fuzzy control system is examined by comparing the performance of the assembly line with the fuzzy control system and two other assembly lines without the fuzzy control system.

## 5.1 Problem Description and Main Assumptions

The assembly line shown in Figure 5-1 is considered in this study, and it consists of $M$ workstations and $M-1$ buffers between workstations. $B_i$ $(0 \le i \le M)$ denotes buffer $i$. $B_i$ $(1 \le i \le M-1)$ has a finite capacity. $B_0$ is an infinite source of raw material so that station 1 is never starved, and $B_M$ has infinite storage capacity so that station $M$ is never blocked.

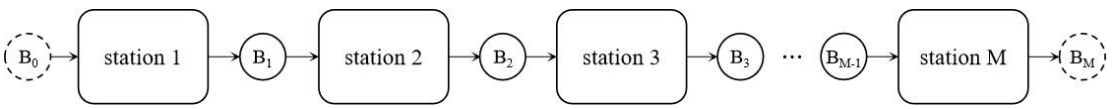


**Figure 5-1:** Structure of the assembly line considered in this study

Machines in workstations are unreliable and can be operative or down. For a brand-new machine, its initial processing time is around the expected value, however, its processing ability will be degraded with the increase of production time. Before the failure of the machine, there are several health states with different defective levels (Peng et al., 2019), which depend on the number of signs of potential failure of the machine. Changes in health states of machines bring significant changes in processing abilities of workstations, and in turn, the initial workload balance will be broken. However, there is little literature on the real-time workload balance of the assembly process considering the degradation process of machines. To fill this gap, the degradation process of each machine is considered in this research, and the operative period of a machine is divided into three health states: normal stage (denoted by $ns$), minor defective stage (denoted by $md$) and severe defective stage (denoted by $sd$) (see Figure 5-2). Machine failure comes after the severe defective stage. After maintenance, a machine will enter the normal stage.

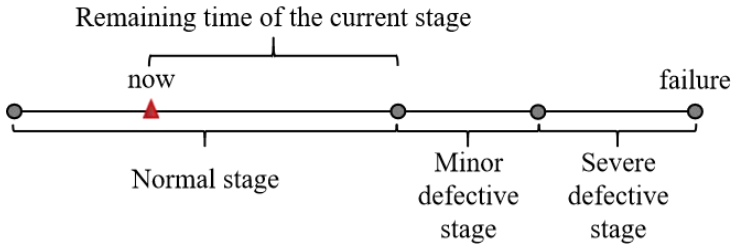


**Figure 5-2:** Three stages in the operative period

After the assembly line is re-balanced, a new production cycle will begin. Then, the demand quantity of products will be updated by the difference between the initial demand quantity of the current production cycle and the number of products finished already.

Main assumptions are stated as follows:

(1) It is assumed that machines fail randomly with a probability and are repaired randomly with another probability. The failure rate and the repair rate of machines are $\lambda$ and $\mu$, respectively. Following Tamani et al. (2011), the uptime and downtime of machines follow the exponential distributions with the means of $\frac{1}{\lambda}$ and $\frac{1}{\mu}$, respectively.

(2) Compared with the average processing time of a task on a machine at the normal stage, the average processing times on the machine at the stages of minor defective stage and severe defective stage are assumed to be $er_{md}$ and $er_{sd}$ times longer, respectively. Let $state_i$ denote the health state of the machine in workstation $i$, and $state_i \in \{ns, md, sd\}$. At the health state of $ns$, $er_{ns}$ =1 and the processing time of task $i$ is $t_i$. Thus, when the machine in workstation $j$ is at the health state of $state_i$, the processing time of task $i$ assigned to workstation $j$ is $t_i \cdot er_{state_j}$.

(3) There are significant fluctuations in the processing times during the degradation process. Thus, for each health state of a machine, the corresponding production rate is assumed to follow a normal distribution: $N\left(\frac{1}{T_{s_i}}, 0.1 \cdot T_{s_i}\right)$, $i = 1,2,3$. $T_{s_i}$ denotes the sum of processing times of tasks assigned to workstation $i$.

(4) The degradation process can be divided into two or more phases, and following Peng et al. (2019), a machine's health state is assumed to follow a three-state Markov chain.

(5) Sensors can be used to collect real-time information related to the degradation process, and some researchers have developed methods to identify the transition of the health states quickly and exactly with a narrow estimation variance (e.g. Peng et al., 2019). Thus, it is assumed that the health state of each machine is always monitored and known.

(6) According to Dong and He (2007), the duration of each health state can be estimated with high accuracy level. Besides, Hu et al. (2018) proposed a real-time remaining useful life (RUL) prediction method for wind turbine bearings, and the prediction can be determined with the probability density distribution of the RUL after the monitoring period. Peng et al. (2019) dynamically tracked the health state of a real bearing and a hard disk drives, and then accurately predicted their RUL. Thus, it is assumed that the remaining time of each health state can be predicted. Besides, data related to the remaining time

of the current stage are assumed to be available, and the prediction of remaining time is based on the historical data and real-time data obtained by sensors. In this study, the prediction of remaining time is assumed to be available and follow a normal distribution $N(r, 0.1 \cdot r)$, where $r$ is the real remaining time.

(7) If task $i$ is reassigned to another workstation, the preparation time for task $i$ is assumed to follow the uniform distribution: $Uniform\left(1, \frac{\sum_{i=1}^{n} t_i}{n}\right)$ following the way to generate the setup time for each task by Hamta et al. (2013).

## 5.2 The Fuzzy Control System

With the change of health states of machines, the actual operation times of workstations will deviate from the initial ones, and the initial workload balance will be broken. To deal with the novel problem described in section 5.1, a fuzzy control system (see Figure 5-3) is proposed to improve the production efficiency by real-time monitoring and controlling of the assembly process. $FC_i$ denotes the $i^{th}$ fuzzy controller, and there are $M+1$ fuzzy controllers in total. Type 1 fuzzy controller: $FC_1$, is used to analyze the information of the whole assembly line, and to determine whether to re-balance the current assembly line. Each of type 2 fuzzy controllers: $FC_2$ to $FC_{M+1}$, is used to process local information of a workstation and make a decision on how to adjust the production rate of the workstation.

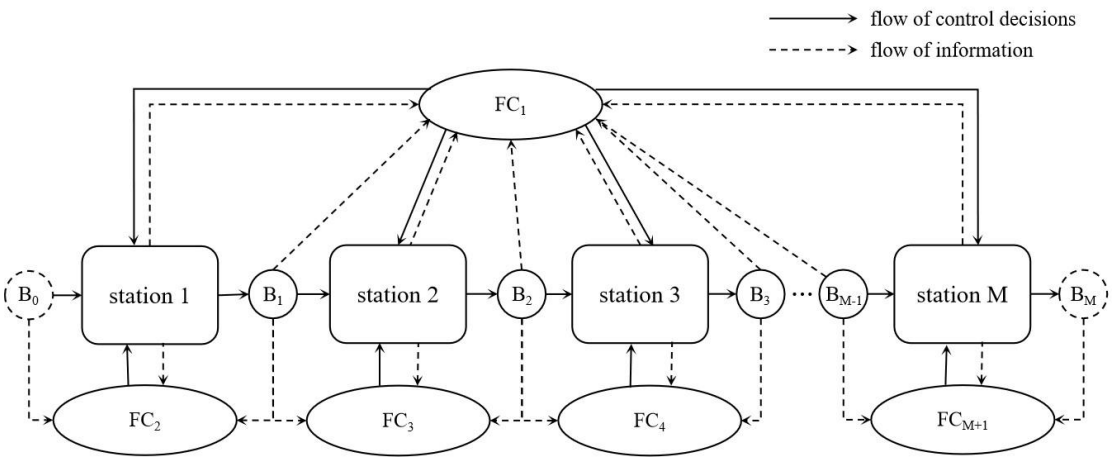


**Figure 5-3:** Framework of the fuzzy control system

A fuzzy controller mimics human thinking and consists of a fuzzifier, some fuzzy IF-THEN rules, a fuzzy inference engine and a defuzzifier (AI-Ebbini et al., 2016). The input data set is transferred into fuzzy sets by fuzzy membership functions in the fuzzification process. For each type of fuzzy controller, there are three inputs and one output.

### 5.2.1 Inputs and the Output of Type 1 Fuzzy Controller

(1) Risk of starvation or blockage

Let $pr^{i}_{state_i}$ denote the average production rate of workstation $i$ at the health state of $state_i$. For each buffer $i$ $(1 \le i \le M-1)$, if $pr^{i}_{state_i} = pr^{i+1}_{state_{i+1}}$, buffer $i$ will be 'safe' since there is no risk of starvation or blockage. Otherwise, the buffer will tend

to be empty or full. When machines are in different health states, the production rates will differ from the initial ones to different extents, and the risk that buffers are starved or blocked will increase. The risk of starvation or blockage of buffer $i$ is defined as follows:

$$R_i = \begin{cases} \dfrac{BL_i - 0.5}{0.5} \cdot \dfrac{pr^i_{state_i} - pr^{i+1}_{state_{i+1}}}{pr^i_{state_i}}, pr^i_{state_i} > pr^{i+1}_{state_{i+1}} \\ \dfrac{0.5 - BL_i}{0.5} \cdot \dfrac{pr^{i+1}_{state_{i+1}} - pr^i_{state_i}}{pr^{i+1}_{state_{i+1}}}, pr^i_{state_i} < pr^{i+1}_{state_{i+1}} \\ 0, pr^i_{state_i} = pr^{i+1}_{state_{i+1}} \end{cases} \quad (5\text{-}1)$$

where $BL_i$ is the inventory level of buffer $i$ and

$$BL_i = \frac{w_i}{BC_i}. \quad (5\text{-}2)$$

$w_i$ denotes the amount of inventory in buffer $i$ and $BC_i$ is the capacity of buffer $i$. When $BL_i$ is close to 1 or 0 and machines in workstations $i$ and $i+1$ are in different health states, $R_i$ will be large.

The risk of the whole assembly line is the average of $R_i$, that is:

$$R = \frac{1}{M-1} \cdot \sum_{i=1}^{M-1} R_i. \quad (5\text{-}3)$$

When $R$ is larger, re-balancing will be needed more urgently. $R$ varies in the range of -1 to 1, and the fuzzy term set is {very small, small, medium, large, very large}.

(2) The urgency of increasing production

When there is a signal that the demand cannot be satisfied, it is urgent to increase total production. The urgency of increasing production is defined in equation (5-4) as follows:

$$Urg = \begin{cases} \dfrac{D \cdot \dfrac{T}{T_d} - P}{D \cdot \dfrac{T}{T_d}}, & D \cdot \dfrac{T}{T_d} \geq P \\ -1, & D \cdot \dfrac{T}{T_d} < P \end{cases} \tag{5-4}$$

where $T$ is the assembly time used in the current production cycle. $D$ and $T_d$ denote the demand quantity of products and delivery time left for the current production cycle, respectively. $D \cdot \dfrac{T}{T_d}$ is the amount of products that should be finished after the production time of $T$ in the current production cycle, and if it is larger than $P$, it will be urgent to increase the total production. $Urg$ varies from -1 to 1, and the fuzzy term set is {small, medium, large}.

(3) Possible production increase after re-balancing

Let station $j$ be the station that has the largest station number among the stations needing maintenance. The possible production increase is defined as follows:

$$P_{incre}=\begin{cases}\dfrac{pr_{new}\cdot\left(\min\limits_{i\in S_{oper}}Rt_i-T_r-\dfrac{n_{avail}}{pr_{new}}\right)-\min\limits_{i\in S_{oper}}pr^{i}_{state_i}\cdot\min\limits_{i\in S_{oper}}Rt_i}{D\cdot\dfrac{\min\limits_{i\in S_{oper}}Rt_i}{T_d}}, T_m=0\\ \dfrac{pr_{new}\cdot\left[\min\left(\min\limits_{i\in S_{oper}}Rt_i,T_m\right)-T_r-\dfrac{n_{avail}}{pr_{new}}\right]-P_a}{D\cdot\dfrac{\min\left(\min\limits_{i\in S_{oper}}Rt_i,T_m\right)}{T_d}}, T_m\neq 0, S_{oper}\neq\phi\\ -1, S_{oper}=\phi\end{cases} \quad (5\text{-}5)$$

where $T_m$ denotes the maintenance time of the assembly line, and is defined as

$$T_m=\max_{1\le i\le M}\{T_{m_i}\}. \quad (5\text{-}6)$$

$T_{m_i}$ denotes the maintenance time needed by workstation $i$. $pr_{new}$ is the production rate of the re-balanced assembly line. $S_{oper}$ denotes the set of operative stations, and $Rt_i$ denotes the remaining time of the current health state of station $i$, $i\in S_{oper}$. The total preparation time $T_r$ for re-balancing is defined as follows:

$$T_r=\sum_{i=1}^{M}T^{i}_{pre}, \quad (5\text{-}7)$$

where $T^{i}_{pre}$ denotes the sum of preparation times for tasks re-assigned to workstation $i$. $n_{avail}$ denotes the number of stations used in the re-balancing plan, and $T_r+\dfrac{1}{pr_{new}}\cdot n_{avail}$ is the shortest time from the beginning of the preparation for the re-balancing plan to obtaining one finished product.

If $j=M$ and $S_{oper}\neq\phi$, $P_a=0$. If $S_{oper}=\phi$, there is no need to re-balance the

assembly line and $P_{incre} = -1$. If $1 \le j < M$ and $S_{oper} \neq \phi$, $P_a$ will be calculated by the method shown in Figure 5-4. $pr_i$ in the figure denotes the average production rate of workstation $i$ in the health state of $state_i$, that is, $pr^i_{state_i}$.

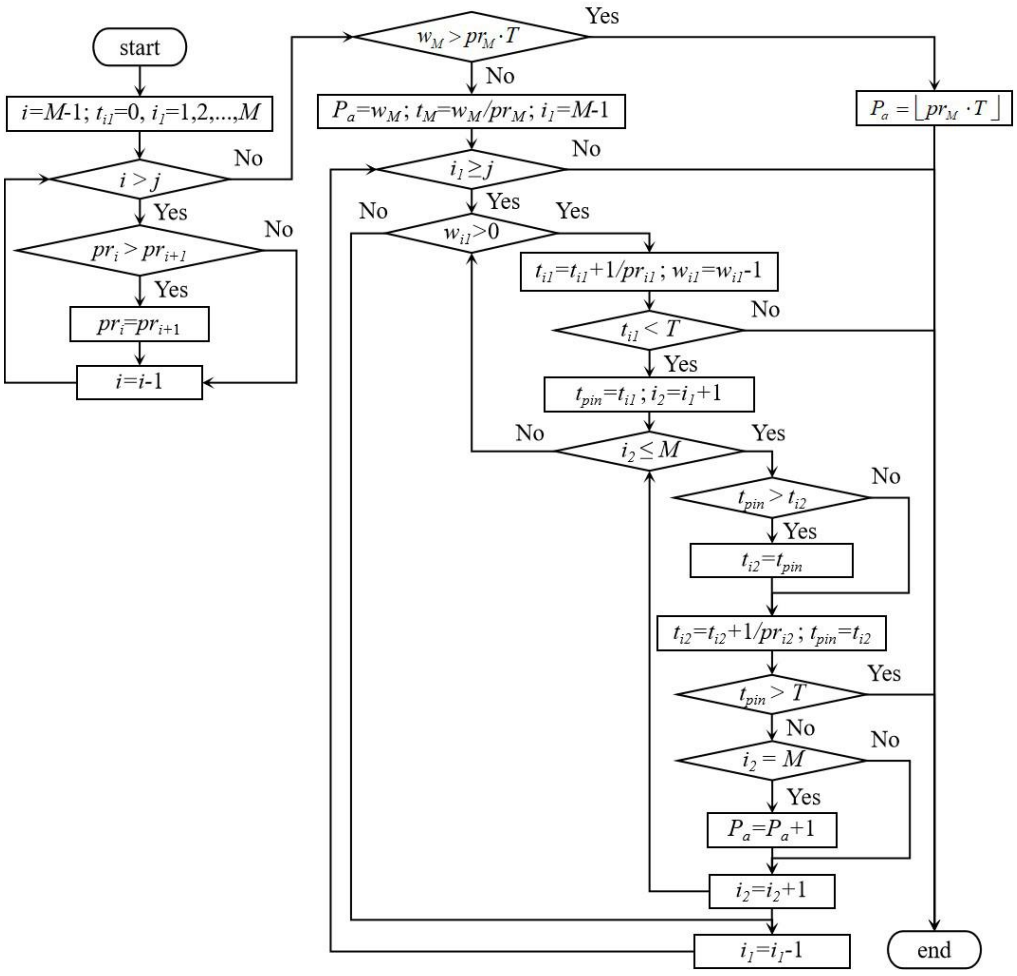


**Figure 5-4:** The method to calculate $P_a$

$P_{incre}$ is restricted to the range of -1 to 1. If $P_{incre}$ is larger than 0, then it will be beneficial to production increase if the assembly line is re-balanced. The fuzzy term set is {very small, small, medium, large, very large}.

(4) Output variable: the necessity of re-balancing, $N$

The fuzzy term set is {very small, small, medium, large, very large}. $N$ is between 0 and 1, and an assembly line is re-balanced only when $N$ is larger than a predetermined threshold.

### 5.2.2 Inputs and the Output of Type 2 Fuzzy Controller

(1) Upstream buffer level and downstream buffer level

For each workstation $i$, the upstream buffer level $BL_{i-1}$ and the downstream buffer level $BL_i$ will be considered. These two buffer levels are closely related to the production rate adjustments. $BL_i$ ($1 \le i \le M-1$) ranges from 0 to 1, and the fuzzy term set is {very small, small, medium, large, very large}.

(2) Production surplus rate

Production surplus is the difference between production and the demand. The production surplus rate in this study is defined as follows:

$$S_i = \begin{cases} \dfrac{\dfrac{P_i}{T} - d}{pr_{normal}^{i} - d}, P_i \ge d \cdot T \\ \dfrac{\dfrac{P_i}{T} - d}{d}, P_i < d \cdot T \end{cases} \tag{5-8}$$

where $P_i$ is the cumulative production of workstation $i$ in the current production

cycle, and $pr^{i}_{normal}$ designates the production rate that can be achieved by workstation $i$ at the normal stage. $d$ denotes the production rate required by demand. It is initialized by $\frac{D}{T_d}$, and is set to a small value smaller than $pr_{new}$ after the decision of assembly line re-balancing to make up for the production loss as soon as possible. Therefore, the production rate of workstation $i$ is always adjusted to be around $d$. $S_i$ is restricted to the range of -1 to 1, and the fuzzy term set is {very small, small, medium, large, very large}.

(3) Output variable: production rate adjustment of workstation $i$

Let $adj\_pr_i$ denote the production rate adjustment of workstation $i$. $adj\_pr_i$ is between -1 and 1, and the fuzzy term set is {very small, small, medium, large, very large}.

If $adj\_pr_i$ is larger than 0, the current production rate of workstation $i$, $pr_i$, should be increased toward $\max\left(pr'_i, pr^{i}_{state_i}\right)$ where $pr'_i$ denotes a production rate generated randomly following the normal distribution $N\left(pr^{i}_{state_i}, 0.1 \cdot pr^{i}_{state_i}\right)$. Otherwise, $pr_i$ should be decreased toward 0. The production rate after adjustment is defined as follows:

$$pr''_i = \min(\max(pr'_i, pr^{i}_{state_i}), \max(0, pr_i \cdot adj\_pr_i + pr'_i)) . \quad (5\text{-}9)$$

The production rate of workstation $i$ is based on the current value. Additionally,

the maximum value after adjustment can reach the average level considering the health state, and some randomness is also allowed by considering the randomly generated $pr_i'$ at the same time. Thus, the production rate is adjusted to a reasonable extent.

### 5.2.3 Fuzzy Rules of Fuzzy Controllers

As shown in Table 5-1, there are 75 fuzzy If-Then rules for type 1 fuzzy controller. Re-balancing is done to reduce the risk that the whole assembly line is stopped by starvation or blockage and to reduce production loss during the time that not all workstations are operative. Thus, fuzzy rules are set to make sure that re-balancing is implemented only when two requirements are met at the same time. The first is that it is likely that the production after re-balancing during the considered period is more than that without re-balancing. The second is that there is a risk that many workstations are idle due to blockage or starvation or there is a risk that the demand cannot be satisfied based on the present production progress. Therefore, re-balancing is conducted only when necessary, and stability of the assembly process is considered.

**Table 5-1:** Fuzzy rules for type 1 fuzzy controller

| *Urg=S* | | | | | |
|---|---|---|---|---|---|
| | $P_{incre}$ | | | | |
| *R* | VS | S | ME | L | VL |
| VS | VS | VS | S | N | N |
| S | VS | VS | S | N | N |
| ME | VS | VS | N | L | L |
| L | VS | S | L | VL | VL |
| VL | VS | S | L | VL | VL |
| *Urg=ME* | | | | | |
| | $P_{incre}$ | | | | |
| *R* | VS | S | ME | L | VL |
| VS | VS | S | N | N | VL |
| S | VS | S | N | L | VL |
| ME | VS | S | L | VL | VL |
| L | VS | S | L | VL | VL |
| VL | S | N | L | VL | VL |
| *Urg=L* | | | | | |
| | $P_{incre}$ | | | | |
| *R* | VS | S | ME | L | VL |
| VS | VS | S | L | VL | VL |
| S | VS | S | L | VL | VL |
| ME | VS | N | L | VL | VL |
| L | S | N | L | VL | VL |
| VL | S | N | L | VL | VL |

Note: VS, S, ME, L and VL denote very small, small, medium, large and very large, respectively.

Table 5-2 shows the 125 fuzzy rules for type 2 fuzzy controllers. When there is no sign of blockage or starvation, fuzzy rules are set so that production rates are adjusted mainly based on surplus rates. Otherwise, efforts will be made to eliminate blockage and starvation. Thus, for each workstation, the production rate is adjusted based on the production progress when there is no risk of starvation or blockage, and in-time acceleration or deceleration on the production rate are implemented to prevent the propagation of the adverse impacts of blockage and starvation.

**Table 5-2:** Fuzzy rules for type 2 fuzzy controllers

| $S_i$=*VS* | | | | | |
|---|---|---|---|---|---|
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | L | S | S | S | VS |
| S | VL | VL | VL | L | VS |
| ME | VL | VL | VL | VL | S |
| L | VL | VL | VL | VL | S |
| VL | VL | VL | VL | VL | ME |
| $S_i$=*S* | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | L | S | S | VS | VS |
| S | VL | VL | L | L | VS |
| ME | VL | VL | VL | L | S |
| L | VL | VL | VL | L | S |
| VL | VL | VL | VL | L | ME |
| $S_i$=*ME* | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | ME | S | VS | VS | VS |
| S | VL | ME | ME | ME | VS |
| ME | VL | L | ME | ME | S |
| L | VL | VL | ME | ME | S |
| VL | VL | VL | L | ME | ME |
| $S_i$=*L* | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | S | VS | VS | VS | VS |
| S | ME | S | S | S | VS |
| ME | ME | S | S | S | VS |
| L | L | ME | S | S | S |
| VL | VL | L | L | ME | ME |
| $S_i$=*VL* | | | | | |
| | $BL_i$ | | | | |
| $BL_{i-1}$ | VS | S | ME | L | VL |
| VS | VS | VS | VS | VS | VS |
| S | S | VS | VS | VS | VS |
| ME | ME | VS | VS | VS | VS |
| L | L | ME | VS | VS | VS |
| VL | VL | L | L | ME | ME |

### 5.2.4 Defuzzification

After the fuzzification process, input data can be interpreted by the fuzzy inference system, whose output is a set of fuzzy membership values. Since the final output should be a single number, the defuzzification process is necessary to transfer the fuzzy outputs into a final crisp output. The centroid method is used for defuzzification (AI-Ebbini et al., 2016).

## 5.3 Numerical Results

### 5.3.1 Settings of Experiments

The proposed fuzzy control system is tested by the SALBP instance of KILBRID (45 tasks), whose parameters (i.e. task times and precedence relationship) can be found in the data set shown on https://assembly-line-balancing.de/salbp/benchmark-data-sets-1993/. Uncertainties, such as blockage, starvation, maintenance and re-balancing, are considered in this study. Since whether an assembly line can produce enough products to satisfy the demand is an important performance indicator, all the task times are divided by 100 to differentiate the outputs of different assembly lines. There are four workstations in total, and the average operation time is 1.38.

Since the problem defined in this study is novel, there are no existing benchmark instances, and thus, the performance of two comparative assembly lines is compared with that of the assembly line with the proposed fuzzy control system. Descriptions of the three assembly lines considered in this study are shown in Table 5-3. When no re-balancing has been conducted, the production rates of $AS_1$ are the same as those of $AS_2$. After re-balancing, the production rates of $AS_1$ will be generated randomly following the normal distribution based on the updated workload of each workstation. Production rates of $AS_2$ and $AS_3$ are the same if there is no delay before the recognition of machine breakdown for $AS_3$.

**Table 5-3:** Assembly lines considered in this study

| No. | With a fuzzy system | With a maintenance delay | Description of the delay when applicable |
|---|---|---|---|
| $AS_1$ | Yes | No | - |
| $AS_2$ | No | No | - |
| $AS_3$ | No | Yes, and it follows a normal distribution | $N\left(5 \cdot T_{S_i}, 0.5 \cdot T_{S_i}\right)$ |

Real-time information of an assembly line is monitored and inputs of the two types of fuzzy controllers will be updated. If the output of type 1 fuzzy controller is larger than 0.79, the assembly line will be re-balanced. Otherwise, the production rate of each workstation will be adjusted by type 2 fuzzy controller.

Since solving the assembly line re-balancing problem is not related to the main contributions of this study, ACO-BS developed by Huo et al. (2018) is used iteratively to obtain the optimal assembly line re-balancing solution when re-balancing is necessary. The method to work out the assembly line re-balancing problem is shown in Figure 5-5. Let $m^*$ denote the number of available workstations, and let $tr_i$ designate the preparation time needed by solution $s_i$. Let $C$ denote the cycle time, and then,

$$\sum_{i=1}^{m^*} \frac{C}{er_{state_i}} \geq \sum_{i=1}^{n} t_i \,. \tag{5-10}$$

$LB$ denotes the lower bound of cycle time, given the number of available workstations $m^*$, and is defined as

$$LB = \max\left\{ \max_{1 \leq i \leq n}(t_i), \frac{\sum_{i=1}^{n} t_i}{\sum_{i=1}^{m^*} \frac{1}{er_{state_i}}} \right\}. \tag{5-11}$$

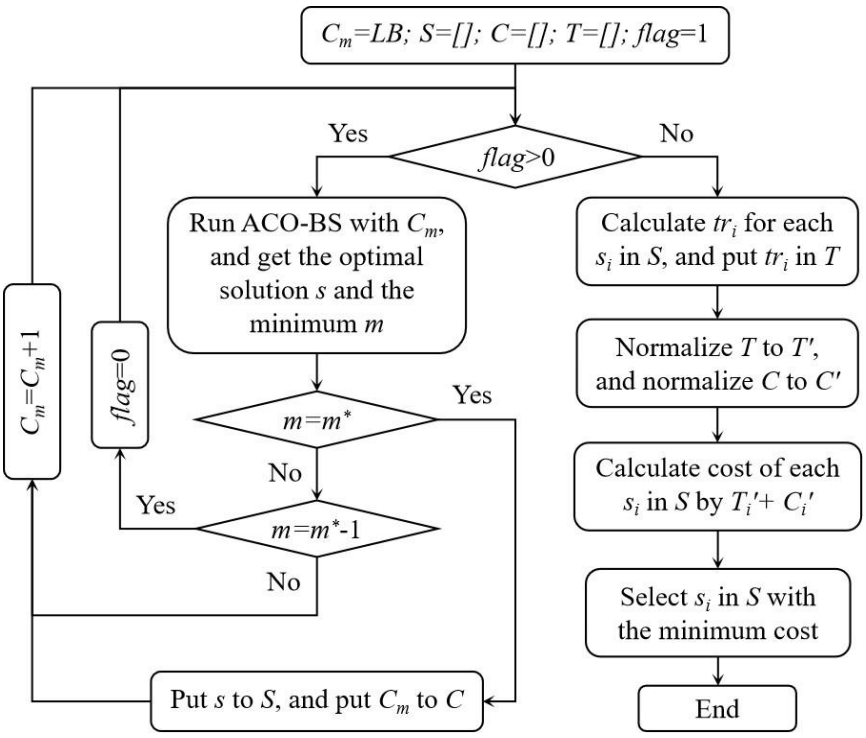


**Figure 5-5:** The method to generate the re-balancing solution

### 5.3.2 Results

All the experiments in this study are conducted by Simulink in Matlab (R2016a), and the total simulation time is 1000. The transition matrix to generate the three-state Markov Chain is [0.999,0.001,0;0,0.999,0.001;0,0,1].

For the pair of parameters $(\lambda, \mu)$ to generate uptimes and downtimes of machines, there are three different combinations: (0.001,0.1), (0.01,0.5) and (0.01,0.1). The average production rate of a workstation at the normal stage is 0.725, which is the largest average production rate. Since the machines are not reliable and the

degradation process is considered, reasonable demand production rate should be at least smaller than $0.725 \cdot \min_{1 \le i \le 3} \frac{\mu_i}{\lambda_i + \mu_i}$, that is, 0.659. Thus, three levels of demand production rate are set to be 0.5, 0.3, 0.1. According to Yan et al. (2018), if the processing time exceeds the expectation by 30%, a machining center can be treated as failed. Thus, an extension rate of 1.3 is relatively high, and 1.3 is set to be the extension rate in the health state of severe defective. In this study, two sets of extension rates $(er_n, er_{md}, er_{sd})$ are set to be (1,1.15,1.3) and (1,1.5,2), so that different characteristics of the degradation process are considered. Therefore, there are 18 experiments in total, and ten random runs, with different seeds to grantee the independent states of workstations, are conducted for each experiment. The means and standard deviations of the ten runs are shown in Tables 5-4 and 5-5. There are six performance indicators: average blockage ratio (ratio of the duration of blocking time to the total simulation time), average buffer level, average starvation ratio (ratio of the duration of starving time to the total simulation time), total simulation time, total production and number of assembly line re-balancing.

**Table 5-4:** Results of the numerical experiments with the extension rates of (1,1.15,1.3)

| λ | μ | μ/λ | No. | Mean | d=0.5 | | | | | | d=0.3 | | | | | | d=0.1 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | and std | b_r | st_r | BL | t | P | no_r | b_r | st_r | BL | t | P | no_r | b_r | st_r | BL | t | P | no_r |
| 0.001 | 0.1 | 100 | $AS_1$ | mean | 0.000 | 0.032 | 0.087 | 962.836 | 472.800 | 0.400 | 0.000 | 0.033 | 0.082 | 678.811 | 300.000 | 0.400 | 0.001 | 0.026 | 0.092 | 248.598 | 100.000 | 0.100 |
| | | | | std | 0.000 | 0.046 | 0.037 | 26.760 | 54.634 | 0.699 | 0.000 | 0.047 | 0.034 | 65.542 | 0.000 | 0.699 | 0.005 | 0.020 | 0.053 | 9.664 | 0.000 | 0.316 |
| | | | $AS_2$ | mean | 0.003 | 0.020 | 0.172 | 939.482 | 500.000 | 0.000 | 0.003 | 0.021 | 0.129 | 567.069 | 300.000 | 0.000 | 0.005 | 0.032 | 0.097 | 197.921 | 100.000 | 0.000 |
| | | | | std | 0.004 | 0.014 | 0.179 | 18.322 | 0.000 | 0.000 | 0.006 | 0.012 | 0.186 | 15.863 | 0.000 | 0.000 | 0.011 | 0.022 | 0.168 | 17.184 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.006 | 0.031 | 0.218 | 959.098 | 500.000 | 0.000 | 0.005 | 0.031 | 0.168 | 579.641 | 300.000 | 0.000 | 0.007 | 0.047 | 0.127 | 205.019 | 100.000 | 0.000 |
| | | | | std | 0.007 | 0.018 | 0.180 | 22.756 | 0.000 | 0.000 | 0.010 | 0.017 | 0.189 | 17.683 | 0.000 | 0.000 | 0.017 | 0.033 | 0.188 | 18.012 | 0.000 | 0.000 |
| 0.01 | 0.5 | 50 | $AS_1$ | mean | 0.000 | 0.013 | 0.107 | 933.312 | 500.000 | 0.000 | 0.000 | 0.014 | 0.098 | 639.733 | 300.000 | 0.000 | 0.000 | 0.024 | 0.091 | 241.920 | 100.000 | 0.000 |
| | | | | std | 0.000 | 0.003 | 0.010 | 4.817 | 0.000 | 0.000 | 0.000 | 0.002 | 0.008 | 2.671 | 0.000 | 0.000 | 0.000 | 0.004 | 0.011 | 2.166 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.000 | 0.021 | 0.184 | 911.951 | 500.000 | 0.000 | 0.000 | 0.027 | 0.137 | 552.131 | 300.000 | 0.000 | 0.000 | 0.043 | 0.073 | 188.753 | 100.000 | 0.000 |
| | | | | std | 0.001 | 0.006 | 0.082 | 4.728 | 0.000 | 0.000 | 0.000 | 0.009 | 0.074 | 5.498 | 0.000 | 0.000 | 0.000 | 0.012 | 0.044 | 3.591 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.007 | 0.035 | 0.361 | 994.703 | 497.500 | 0.000 | 0.005 | 0.049 | 0.277 | 608.979 | 300.000 | 0.000 | 0.003 | 0.086 | 0.151 | 213.113 | 100.000 | 0.000 |
| | | | | std | 0.005 | 0.013 | 0.089 | 6.776 | 4.301 | 0.000 | 0.007 | 0.019 | 0.103 | 16.109 | 0.000 | 0.000 | 0.006 | 0.034 | 0.079 | 13.567 | 0.000 | 0.000 |
| 0.01 | 0.1 | 10 | $AS_1$ | mean | 0.005 | 0.069 | 0.240 | 998.341 | 436.900 | 0.600 | 0.003 | 0.070 | 0.167 | 754.292 | 300.000 | 0.900 | 0.000 | 0.104 | 0.130 | 293.422 | 100.000 | 0.400 |
| | | | | std | 0.006 | 0.032 | 0.102 | 5.246 | 77.182 | 0.699 | 0.005 | 0.034 | 0.069 | 58.313 | 0.000 | 0.876 | 0.000 | 0.056 | 0.044 | 43.372 | 0.000 | 0.516 |
| | | | $AS_2$ | mean | 0.025 | 0.074 | 0.388 | 1000.000 | 472.400 | 0.000 | 0.026 | 0.076 | 0.353 | 651.717 | 300.000 | 0.000 | 0.018 | 0.128 | 0.185 | 238.106 | 100.000 | 0.000 |
| | | | | std | 0.015 | 0.036 | 0.094 | 0.000 | 21.557 | 0.000 | 0.016 | 0.026 | 0.081 | 23.855 | 0.000 | 0.000 | 0.033 | 0.051 | 0.122 | 29.140 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.047 | 0.105 | 0.389 | 1000.000 | 425.300 | 0.000 | 0.046 | 0.100 | 0.388 | 717.525 | 300.000 | 0.000 | 0.032 | 0.152 | 0.242 | 268.318 | 100.000 | 0.000 |
| | | | | std | 0.017 | 0.038 | 0.095 | 0.000 | 19.079 | 0.000 | 0.019 | 0.020 | 0.069 | 30.944 | 0.000 | 0.000 | 0.040 | 0.061 | 0.123 | 40.335 | 0.000 | 0.000 |

**Table 5-5:** Results of the numerical experiments with the extension rates of (1,1.5,2)

| λ | μ | μ/λ | No. | Mean and std | d=0.5 | | | | | | d=0.3 | | | | | | d=0.1 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | b_r | st_r | BL | t | P | no_r | b_r | st_r | BL | t | P | no_r | b_r | st_r | BL | t | P | no_r |
| 0.001 | 0.1 | 100 | $AS_1$ | mean | 0.000 | 0.017 | 0.161 | 1000.000 | 368.000 | 0.000 | 0.000 | 0.016 | 0.120 | 856.739 | 300.000 | 0.000 | 0.000 | 0.032 | 0.100 | 324.829 | 100.000 | 0.000 |
| | | | | std | 0.000 | 0.010 | 0.103 | 0.000 | 4.595 | 0.000 | 0.000 | 0.009 | 0.024 | 8.414 | 0.000 | 0.000 | 0.000 | 0.020 | 0.019 | 4.940 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.001 | 0.017 | 0.184 | 1000.000 | 355.200 | 0.000 | 0.001 | 0.019 | 0.172 | 844.791 | 300.000 | 0.000 | 0.000 | 0.046 | 0.126 | 286.422 | 100.000 | 0.000 |
| | | | | std | 0.002 | 0.010 | 0.126 | 0.000 | 4.290 | 0.000 | 0.002 | 0.011 | 0.119 | 12.725 | 0.000 | 0.000 | 0.001 | 0.028 | 0.088 | 9.597 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.001 | 0.021 | 0.212 | 1000.000 | 350.600 | 0.000 | 0.001 | 0.023 | 0.199 | 856.846 | 300.000 | 0.000 | 0.001 | 0.054 | 0.142 | 292.282 | 100.000 | 0.000 |
| | | | | std | 0.002 | 0.012 | 0.129 | 0.000 | 6.132 | 0.000 | 0.002 | 0.014 | 0.124 | 19.525 | 0.000 | 0.000 | 0.003 | 0.034 | 0.095 | 13.691 | 0.000 | 0.000 |
| 0.01 | 0.5 | 50 | $AS_1$ | mean | 0.000 | 0.028 | 0.263 | 1000.000 | 380.000 | 0.000 | 0.000 | 0.023 | 0.150 | 828.318 | 300.000 | 0.000 | 0.000 | 0.037 | 0.104 | 310.112 | 100.000 | 0.000 |
| | | | | std | 0.000 | 0.010 | 0.103 | 0.000 | 5.055 | 0.000 | 0.000 | 0.005 | 0.035 | 9.386 | 0.000 | 0.000 | 0.000 | 0.011 | 0.017 | 5.551 | 0.000 | 0.000 |
| | | | $AS_2$ | mean | 0.007 | 0.035 | 0.288 | 1000.000 | 368.400 | 0.000 | 0.006 | 0.039 | 0.259 | 818.242 | 300.000 | 0.000 | 0.001 | 0.070 | 0.143 | 278.446 | 100.000 | 0.000 |
| | | | | std | 0.008 | 0.013 | 0.113 | 0.000 | 6.818 | 0.000 | 0.009 | 0.011 | 0.106 | 12.992 | 0.000 | 0.000 | 0.003 | 0.017 | 0.070 | 5.099 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.005 | 0.041 | 0.297 | 1000.000 | 343.700 | 0.000 | 0.005 | 0.045 | 0.282 | 873.533 | 300.000 | 0.000 | 0.001 | 0.085 | 0.182 | 302.867 | 100.000 | 0.000 |
| | | | | std | 0.005 | 0.014 | 0.073 | 0.000 | 6.165 | 0.000 | 0.005 | 0.014 | 0.069 | 23.757 | 0.000 | 0.000 | 0.004 | 0.025 | 0.066 | 8.430 | 0.000 | 0.000 |
| 0.01 | 0.1 | 10 | $AS_1$ | mean | 0.011 | 0.055 | 0.336 | 1000.000 | 336.500 | 0.100 | 0.003 | 0.065 | 0.235 | 910.310 | 296.100 | 0.100 | 0.000 | 0.081 | 0.143 | 353.471 | 100.000 | 0.300 |
| | | | | std | 0.010 | 0.017 | 0.122 | 0.000 | 18.822 | 0.316 | 0.005 | 0.020 | 0.069 | 35.365 | 12.333 | 0.316 | 0.000 | 0.036 | 0.039 | 30.898 | 0.000 | 0.483 |
| | | | $AS_2$ | mean | 0.022 | 0.060 | 0.360 | 1000.000 | 330.600 | 0.000 | 0.020 | 0.061 | 0.348 | 905.633 | 300.000 | 0.000 | 0.010 | 0.106 | 0.204 | 324.660 | 100.000 | 0.000 |
| | | | | std | 0.014 | 0.017 | 0.119 | 0.000 | 12.076 | 0.000 | 0.015 | 0.015 | 0.117 | 36.820 | 0.000 | 0.000 | 0.013 | 0.040 | 0.053 | 30.731 | 0.000 | 0.000 |
| | | | $AS_3$ | mean | 0.034 | 0.070 | 0.393 | 1000.000 | 306.200 | 0.000 | 0.032 | 0.072 | 0.386 | 967.798 | 296.500 | 0.000 | 0.016 | 0.121 | 0.245 | 354.410 | 100.000 | 0.000 |
| | | | | std | 0.031 | 0.022 | 0.099 | 0.000 | 16.943 | 0.000 | 0.029 | 0.019 | 0.096 | 28.030 | 11.068 | 0.000 | 0.013 | 0.044 | 0.067 | 35.458 | 0.000 | 0.000 |

In order to highlight the characteristics of the main performance indicators, the means of idle ratio (sum of the average blockage ratio and average starvation ratio), buffer level and total production are shown in Figures 5-6, 5-7 and 5-8. Data of the first 9 cases are extracted from Table 5-4 where $er_1 = (1,1.15,1.3)$, and data of the last 9 cases are extracted from Table 5-5 where $er_2 = (1,1.5,2)$. Values in the brackets are the pairs of $(\lambda, \mu)$, and points which have the same horizontal coordinate have the same $\lambda$ and $\mu$. For cases 1 to 3, 4 to 6, 7 to 9, the corresponding demand rates are 0.5, 0.3 and 0.1, respectively. For cases 10 to 18, the same rule applies.

As Figure 5-6 shows, the idle ratio of $AS_1$ is the lowest for almost all the cases, which implies that with the proposed fuzzy control system, the utilization of machines is higher. The idle ratio of $AS_2$ is higher, and the idle ratio of $AS_3$ where the breakdown of a machine cannot be recognized immediately is the highest. This is because if there is no timely repair, the prolonged downtime can quickly affect the upstream and downstream stations via blockage and starvation. For the first 9 cases, there are large differences between the idle ratios of different assembly lines. However, for the last 9 cases, these differences become smaller and the idle ratios become smaller in general.

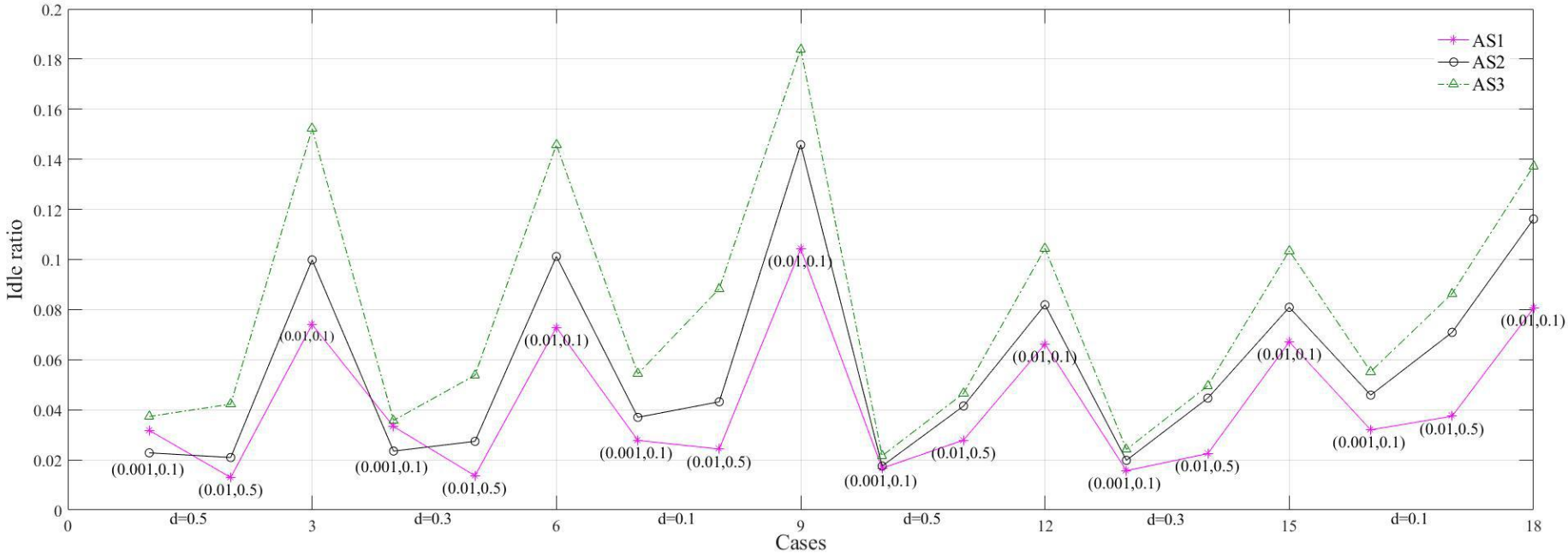


**Figure 5-6:** Average idle ratio for the three assembly lines

Figure 5-7 shows that $AS_1$ has significant strength in reducing work-in-progress. By contrast, the buffer levels of $AS_3$ are the highest for all the cases. Interestingly, for cases which have the same demand rate and extension rate (i.e. cases 1 to 3, cases 4 to 6, cases 7 to 9, cases 10 to 12, cases 13 to 15, cases 16 to 18), the buffer level increase monotonically with the decrease of $\frac{\mu}{\lambda}$. This suggests that the buffer level will be higher, when there is shorter expected uptime or longer expected downtime and other parameters stay the same. This finding can be used to explain the worst performance of $AS_3$, where the downtime is longer due to the low level of information transparency. In addition, when the extension rate changes from $er_1$ to $er_2$ and the other parameters stay the same, for each of the three assembly lines, the buffer level increases for most cases.

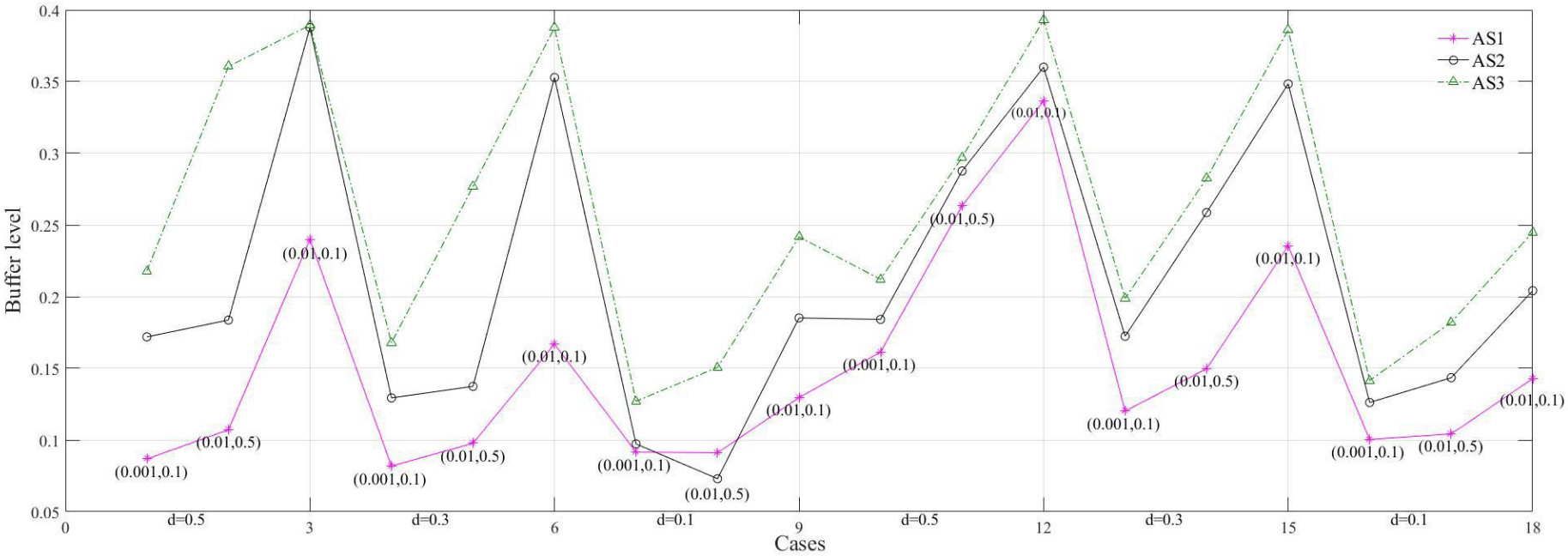


**Figure 5-7:** Average buffer level for the three assembly lines

Although the adjustment of production rate to eliminate starvation and blockage will bring production loss to $AS_1$, the proposed fuzzy control system helps $AS_1$ maintain a low average buffer level, which is not at the expense of production reduction (see Figure 5-8). In general, production of $AS_3$ is the least. When the extension rate is $er_2$ and $d = 0.5$, the demand quantity is 500, but there is a relatively large difference between the total production of each assembly line and the demand. Actually, when the extension rate changes from $er_1$ to $er_2$, the processing abilities of workstations changes with larger extents and the final output of an assembly line decreases, which can also be used to explain the finding from Figure 5-7 that there is an increasing tendency for buffer levels when the extension rate changes from $er_1$ to $er_2$.

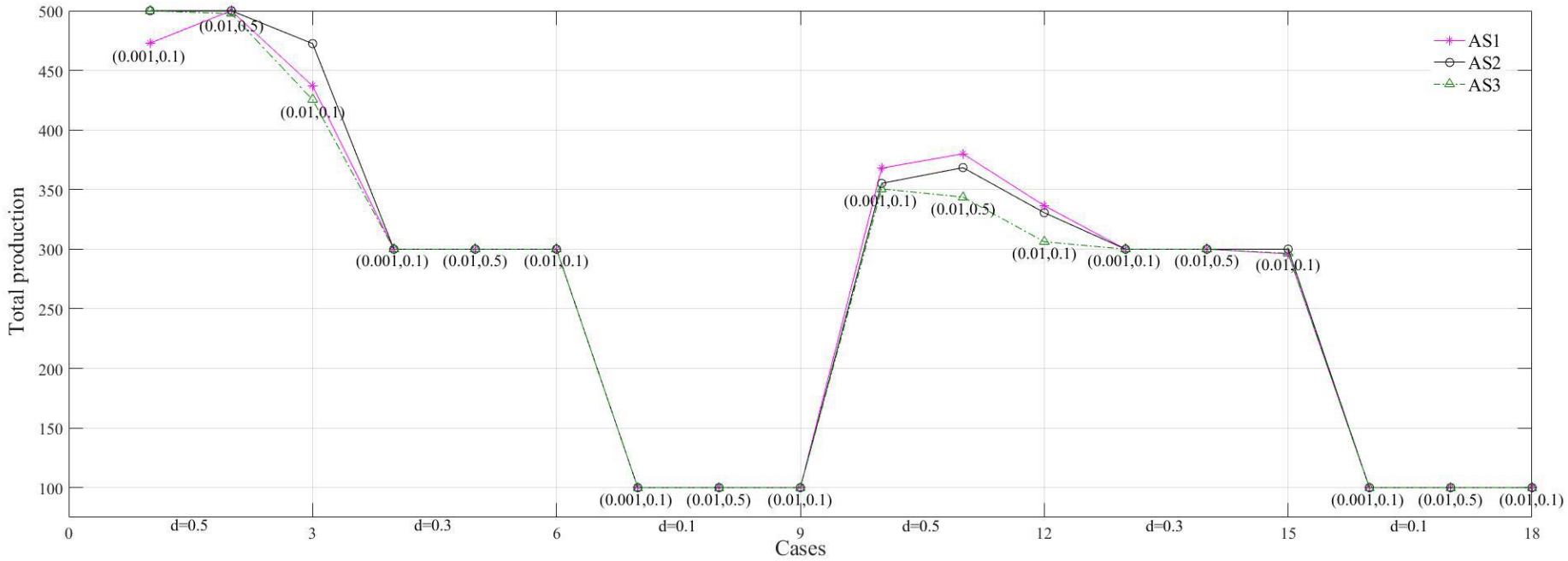


**Figure 5-8:** Average total production for the three assembly lines

## 5.4 Summary

In this chapter, the degradation process of a machine is divided into the normal stage, minor defective stage and severe defective stage. The prediction of remaining time of each health state is assumed to be available, and the production rates in each health state are generated following normal distributions. A real-time fuzzy control system is utilized to utilize the real-time information of the assembly line and support the decision-making on when to re-balance the assembly line and how to adjust the production rates of workstations. As the results of the numerical experiments show, an assembly line with the proposed fuzzy control system achieves lower buffer level and higher utilization of machines, without the expense of production reduction.

# Chapter 6. Conclusions and Future Work

In this chapter, the main contributions and conclusions are presented. Besides, the limitations of this research are analyzed, and future work is provided accordingly.

## 6.1 Main Contributions

Although many explorations have been made by researchers, with the increasing complexity of SALBP, the development of new approaches to suit the complex assembly environment is urgent. Industry 4.0 breaks the information barriers among the different parts of an assembly line, since smart, connected products enabled by advanced information and communication technology, can intelligently interact and communicate with each other and collect, process and produce information. On-line monitoring of the assembly lines becomes possible with real-time information obtained by advanced information and network technologies in the era of Industry 4.0, thus the assembly lines will be more re-configurable. However, there is little literature on the assembly process considering the new attributes and opportunities brought by Industry 4.0. A wide range of disruptions can break the current workload balance, but the link between disruptions to an assembly line and the corresponding reactions tends to be ignored. Due to the randomness and nonlinearity caused by unpredictable disruptions, accurate analysis of an automotive assembly line is difficult. Therefore, to fill the above research gaps, the following works are

conducted. In order to deal with SALBP, an algorithm based on ant colony algorithm combined by beam search, ACO-BS, is developed to improve the solution quality and speed up the searching process. Besides, real-time information of an assembly line is monitored by a fuzzy control system, and when to re-balance the current assembly line and how to adjust the production rate of each workstation are determined. The number of open workstations will change based on the availability of workstations and the 'decisions' of the fuzzy control system, and task re-assignments will be implemented after the re-balancing decision is made. Consequently, an assembly line can be adjusted with the proposed fuzzy control system to adapt to the dynamic environment. The main contributions of this research are addressed as follows:

(1) ACO-BS based on beam ant colony algorithm is proposed to deal with the large-scale ALBP, and the effectiveness of the algorithm is demonstrated by the results of benchmark instances and random instances. This will contribute to the assembly process of complex products by generating satisfactory solutions within acceptable computation time.

(2) The real-time balance control of an assembly line is explored in a novel context created by Industry 4.0, and a fuzzy control system is used to process the real-time production information of an assembly line efficiently and make

on-line adjustment decisions.

(3) The link between disruptions and the corresponding reactions is created by the proposed fuzzy system. The production rate of each workstation is adjusted in time to eliminate blockage and starvation, and the assembly line is re-balanced when necessary. Thus, this research provides references for smart control of the automated assembly lines.

(4) The degradation process of machines is considered when developing the fuzzy control system, and the health states of a machine are divided into the normal stage, minor defective stage and severe defective stage. Thus, the imbalance caused by the transitions of health states of machines is modeled and dealt with by the fuzzy control system.

(5) The proposed fuzzy system is used to model the benefits brought by industry 4.0, and the performance of assembly lines with different levels of information transparency is compared to examine the impact of information transparency on assembly line balancing.

## 6.2 Conclusions

The main conclusions are summarized and discussed as follows:

(1) The proposed algorithm in Chapter 3, ACO-BS, shows advantages in solving the benchmark instances and the large-scale random instances. The good results of ACO-BS compared with the results of ACO, GA and PSO also demonstrate that the algorithm can jump out of local optimum and prevent premature convergence. Premature convergence is a challenging problem for ACO, and the good performance of ACO-BS is due to several strategies. First, the pheromone values are restricted to the interval of [0.01, 0.99] to prevent stagnation. This is because some tasks tend to be assigned to the same workstation if the pheromone values are too large, and some tasks tend to avoid being assigned to a workstation if the pheromone values are too small. There is also an evaporation part for each pheromone value so that it is discouraged to assign one task to the same position. Second, one task will be chosen from the available task set by maximizing the probability or by the roulette-wheel method, with equal probability. Thus, tasks with a higher probability have a larger chance to be selected, but tasks with a lower probability still have chances to be selected. Third, the convergence value is calculated in every iteration. Since the pheromone values are initialized to be 0.5, the convergence

value is 1 at the beginning. When all the pheromone values are close to 0.99 or 0.01, the convergence value will be close to zero. The pheromone values will be reinitialized to be 0.5 when the convergence value is less than 0.05, and this will prevent the stagnation. Therefore, the strategies above drive ACO-BS to search for better solutions rather than staying at the stagnation state.

(2) In Chapter 4, the assembly process is discussed in a dynamic environment, where there are task time reductions due to the learning effect, maintenance due to machine failures, starvation and blockage. A fuzzy control system is developed to deal with the unpredictable disruptions. The numerical results show that the assembly line with the proposed fuzzy control system performs much better in terms of blockage ratio, starvation ratio and the buffer level, with the satisfaction of demand considered. There are two benefits of the proposed fuzzy control system. On one hand, based on the fuzzy control system, an assembly line is re-balanced with tasks re-assigned to the available workstations when necessary to decrease the adverse impacts of failed workstations. On the other hand, the production rate of each workstation tends to be maximum when there is no risk of blockage or starvation, and it is adjusted when necessary to prevent the propagation of the adverse impacts of starvation and blockage on the assembly line.

(3) It assumed that the health states of machines are constant in Chapter 4. To make the definition of the problem more reasonable, in chapter 5, the degradation process of a machine is divided into the normal stage, minor defective stage and severe defective stage, and failure comes after severe defective stage. The processing ability of a machine varies with its health state. Production rates of workstations are generated following the three-state Markov Chain. Before the coming of the next health stage, no information about the duration is available, however, the current health state and the prediction of the duration of the current health state are assumed to be available with the related historical data obtained by sensors. Then, a fuzzy control system is proposed to deal with uncertainties in such a context. As the results of numerical experiments show, compared with the two comparative assembly lines, the assembly line with the proposed fuzzy control system achieves higher utilization of machines and less work-in-progress, without the expense of production reduction. This highlights the importance of short-term assembly line balancing, by which necessary modifications on the assembly plan are made in time. Besides, when the extension rates are larger, the production ability of a machine changes to a larger extent, and the probability of workload imbalance of workstations becomes larger. Then, it is more challenging to keep the workload balance of an assembly line, and then the proposed fuzzy control system will be needed more urgently.

(4) In order to explore real-time workload balance of the assembly line in the context of Industry 4.0, a fuzzy control system is developed to model the benefits brought by Industry 4.0 on information transparency. It is verified by the numerical results of Chapter 4 and Chapter 5 that with the increase of information transparency level, the performance of an assembly line becomes better. That is because with a higher level of information transparency, decisions are made based on the real-time information of the assembly process and in-time adjustments are undertaken accordingly. Thus, information transparency positively affects the performance of an assembly line, and Industry 4.0 will lead us to a more intelligent and efficient era. Practitioners should devote more effort to the adoption and application of new advanced technologies to improve the information transparency level and the intelligence level of the assembly process.

## 6.2 Limitations and Future work

Limitations of this research and future work are presented as follows:

(1) The proposed algorithm, ACO-BS, is used to deal with SALBP, and it cannot be used directly to solve GALBP with more restrictions on task assignments. In the

future, the characteristics of assembly lines, such as the U-shaped assembly line, mixed-model assembly line and two-sided assembly line, will be discussed, and the current algorithm will be improved accordingly to suit to the more complex assembly process.

(2) The proposed fuzzy control system shows advantages in maintaining a low average buffer level without the expense of production reduction. However, it is assumed in Chapters 4 and 5 that there are infinite resources for maintenance and maintenance will take place whenever a machine breaks down. However, finite resources for maintenance will be more possible in practice. In the future, real-time information of the health state of a machine obtained by sensors and the prediction of the remaining useful life will be used to develop maintenance prioritization strategies to take full use of finite maintenance resources.

(3) It is assumed in Chapter 4 and Chapter 5 that the production rates can be adjusted. But the failure rates of machines are assumed to be constant. However, machines working at their maximum production rates have higher failure rates. Thus, failure rates associated with the production rates will be considered in the future.

# References


[1] 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(6), 1405-1429.

[2] Al-Ebbini, L., Oztekin, A., & Chen, Y. (2016). FLAS: Fuzzy lung allocation system for US-based transplantations. *European Journal of Operational Research*, 248(3), 1051-1065.

[3] AkpıNar, 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(1), 574-589.

[4] Andres, C., Miralles, C., & Pastor, R. (2008). Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research*, 187(3), 1212-1223.

[5] Antoine, M., Hind, B. E. H., Wahiba, R. C. K., & Lounes, B. M. (2016). Iterated Local Search for dynamic assembly line rebalancing problem. *IFAC-PapersOnLine*, 49(12), 515-519.

[6] Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, 54(15), 2787-2805.

[7] Bandyopadhyay, D., & Sen, J. (2011). Internet of things: Applications and challenges in technology and standardization. *Wireless Personal Communications*, 58(1), 49-69.

[8] Battaïa, O., & Dolgui, A. (2012). Reduction approaches for a generalized line balancing problem. *Computers & Operations Research*, 39(10), 2337-2345.

[9] Battaïa, O., & Dolgui, A. (2013). A taxonomy of line balancing problems and their solutionapproaches. *International Journal of Production Economics*, 142(2), 259-277.

[10] Bautista, J., & Pereira, J. (2007). Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research*, 177(3), 2016-2032.

[11] Bautista, J., Batalla-García, C., & Alfaro-Pozo, R. (2016). Models for assembly line balancing by temporal, spatial and ergonomic risk attributes. *European Journal of Operational Research*, 251(3), 814-829.

[12] Bautista, J., & Pereira, J. (2002, September). Ant algorithms for assembly line balancing. *In: Dorigo M., Di Caro G., Sampels M. (eds) Ant Algorithms. ANTS 2002. Lecture Notes in Computer Science (vol. 2463, pp. 65-75)*. Springer, Berlin, Heidelberg.

[13] Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32(8), 909-932.

[14] Baykasoğlu, A., & Dereli, T. (2009). Simple and U-type assembly line balancing by using an ant colony based algorithm. *Mathematical and Computational Applications*, 14(1), 1-12.

[15] Becker, C., & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168(3), 694-715.

[16] Blum, C. (2008). Beam-ACO for simple assembly line balancing. *INFORMS Journal on Computing*, 20(4), 618-627.

[17] Boysen, N., Fliedner, M., & Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2), 674-693.

[18] Boysen, N., Fliedner, M., & Scholl, A. (2008). Assembly line balancing: Which model to use when?. *International Journal of Production Economics*, 111(2), 509-528.

[19] Celik, E., Kara, Y., & Atasagun, Y. (2014). A new approach for rebalancing of U-lines with stochastic task times using ant colony optimisation algorithm. *International Journal of Production Research*, 52(24), 7262-7275.

[20] Chang, Q., Pan, C., Xiao, G., & Biller, S. (2013). Integrated modeling of automotive assembly line with material handling. *Journal of Manufacturing Science and Engineering*, 135(1), 1-10.

[21] Cheshmehgaz, H. R., Haron, H., Kazemipour, F., & Desa, M. I. (2012). Accumulated risk of body postures in assembly line balancing problem and modeling through a multi-criteria fuzzy-genetic algorithm. *Computers & Industrial Engineering*, 63(2), 503-512.

[22] Chica, M., Cordón, Ó., 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(18), 3465-3487.

[23] Chica, M., Cordón, 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(1), 709-720.

[24] Dong, M., & He, D. (2007). A segmental hidden semi-Markov model (HSMM)-based diagnostics and prognostics framework and methodology. *Mechanical Systems and Signal Processing*, 21(5), 2248-2266

[25] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1), 53-66.

[26] 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(1), 29-41.

[27] Dou, J., Li, J., & Su, C. (2013). A novel feasible task sequence-oriented discrete particle swarm algorithm for simple assembly line balancing problem of type 1. *The International Journal of Advanced Manufacturing Technology*, 69(9-12), 2445-2457.

[28] Ege, Y., Azizoglu, M., & Ozdemirel, N. E. (2009). Assembly line balancing with station paralleling. *Computers & Industrial Engineering*, 57(4), 1218-1225.

[29] ElMaraghy, H., & ElMaraghy, W. (2016). Smart adaptable assembly systems. *Procedia CIRP*, 44, 4-13.

[30] Fattahi, P., Roshani, A., & Roshani, A. (2011). A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. *The International Journal of Advanced Manufacturing Technology*, 53(1-4), 363-378.

[31] Feng, J., Li, F., Xu, C., & Zhong, R. Y. (2018). Data-Driven Analysis for RFID-Enabled Smart Factory: A Case Study. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, DOI: 10.1109/TSMC.2018.2882838.

[32] Gamberini, R., Grassi, A., & Rimini, B. (2006). A new multi-objective heuristic algorithm for solving the stochastic assembly line re-balancing problem. *International Journal of Production Economics*, 102(2), 226-243.

[33] Hamta, N., Ghomi, S. F., Jolai, F., & Shirazi, M. A. (2013). A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect. *International Journal of Production Economics*, 141(1), 99-111.

[34] HazıR, Ö., & Dolgui, A. (2013). Assembly line balancing under uncertainty: Robust optimization models and exact solution method. *Computers & Industrial Engineering*, 65(2), 261-267.

[35] Hermann, M., Pentek, T., & Otto, B. (2016, January). Design principles for industrie 4.0 scenarios. *In 2016 49th Hawaii international conference on system sciences (HICSS) (pp. 3928-3937)*, IEEE, Koloa, HI, USA.

[36] Hu, H., Ng, K. K. H., & Qin, Y. (2016). Robust parallel machine scheduling problem with uncertainties and sequence-dependent setup time. *Scientific Programming*, 2016, 1-13.

[37] Hu, Y., Li, H., Shi, P., Chai, Z., Wang, K., Xie, X., & Chen, Z. (2018). A prediction method for the real-time remaining useful life of wind turbine bearings based on the Wiener process. *Renewable Energy*, 127, 452-460.

[38] Huang, G. Q., Zhang, Y. F., Chen, X., & Newman, S. T. (2008). RFID-enabled real-time wireless manufacturing for adaptive assembly planning and control. *Journal of Intelligent Manufacturing*, 19(6), 701-713.

[39] Hui, P. L., Chan, K. C., Yeung, K. W., & Ng, F. F. (2002). Fuzzy operator allocation for balance control of assembly lines in apparel manufacturing. *IEEE Transactions on Engineering Management*, 49(2), 173-180.

[40] Huo, J., Wang, Z., Chan, F. T., Lee, C. K., & Strandhagen, J. O. (2018). Assembly Line Balancing Based on Beam Ant Colony Optimisation. *Mathematical Problems in Engineering*, 2018, Article ID 2481435, 1-17.

[41] Jazdi, N. (2014, May). Cyber physical systems in the context of Industry 4.0. *In 2014 IEEE international conference on automation, quality and testing, robotics (pp. 1-4)*. IEEE, Cluj-Napoca, Romania.

[42] Keung, K. L., Lee, C. K. M., Ng, K. K. H., & Yeung, C. K. (2018, December). Smart City Application and Analysis: Real-time Urban Drainage Monitoring by IoT Sensors: A Case Study of Hong Kong. *In 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) (pp. 521-525)*. IEEE, Bangkok, Thailand.

[43] Kilbridge, M., & Wester, L. (1961). The balance delay problem. *Management Science*, 8(1), 69-84.

[44] Kolisch, R., Sprecher, A., & Drexl, A. (1995). Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science*, 41(10), 1693-1703.

[45] Kong, M., Tian, P., & Kao, Y. (2008). A new ant colony optimization algorithm for the multidimensional knapsack problem. *Computers & Operations Research*, 35(8), 2672-2683.

[46] 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.

[47] Kucukkoc, I., & Zhang, D. Z. (2016). Mixed-model parallel two-sided assembly line balancing problem: A flexible agent-based ant colony optimization approach. *Computers & Industrial Engineering*, 97, 58-72.

[48] Lee, C. K. M., Zhang, S. Z., & Ng, K. K. H. (2017a). Development of an industrial Internet of things suite for smart factory towards re-industrialization. *Advances in Manufacturing*, 5(4), 335-343.

[49] Lee, C. K. M., Lv, Y., Ng, K. K. H., Ho, W., & Choy, K. L. (2018a). Design and application of Internet of things-based warehouse management system for smart logistics. *International Journal of Production Research*, 56(8), 2753-2768.

[50] Lee, C. K. M., Keung, K. L., Ng, K. K. H., & Lai, D. C. (2018b, December). Simulation-based Multiple Automated Guided Vehicles Considering Charging and Collision-free Requirements in Automatic Warehouse. *In 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) (pp. 1376-1380)*. IEEE, Bangkok, Thailand.

[51] Lee, C. K. M., Cao, Y., & Ng, K. H. (2017b). Big data analytics for predictive maintenance strategies. *In Supply Chain Management in the Big Data Era (pp. 50-74)*. IGI Global.

[52] Leu, Y. Y., Matheson, L. A., & Rees, L. P. (1994). Assembly line balancing using genetic algorithms with heuristic-generated initial populations and multiple evaluation criteria. *Decision Sciences*, 25(4), 581-605.

[53] Li, Y. (2017). The type-II assembly line rebalancing problem considering stochastic task learning. *International Journal of Production Research*, 55(24), 7334-7355.

[54] Liu, W., Lim, C. C., Shi, P., & Xu, S. (2016). Backstepping fuzzy adaptive control for a class of quantized nonlinear systems. *IEEE Transactions on Fuzzy Systems*, 25(5), 1090-1101.

[55] López-Ibáñez, M., Stützle, T., & Dorigo, M. (2016). Ant colony optimization: A component-wise overview. *Handbook of Heuristics*, 1-37.

[56] Lu, C., & Yang, Z. (2016). Integrated assembly sequence planning and assembly line balancing with ant colony optimization approach. *The International Journal of Advanced Manufacturing Technology*, 83(1-4), 243-256.

[57] Lu, S., Xu, C., Zhong, R. Y., & Wang, L. (2018). A passive RFID tag-based locating and navigating approach for automated guided vehicle. *Computers & Industrial Engineering*, 125, 628-636.

[58] Makssoud, F., Battaïa, O., Dolgui, A., Mpofu, K., & Olabanji, O. (2015). Re-balancing problem for assembly lines: new mathematical model and exact solution method. *Assembly Automation*, 35(1), 16-21.

[59] McMullen, P. R., & Tarasewich, P. (2006). Multi-objective assembly line balancing via a modified ant colony optimization technique. *International Journal of Production Research*, 44(1), 27-42.

[60] McMullen, P. R., & Tarasewich, P. (2003). Using ant techniques to solve the assembly line balancing problem. *IIE Transactions*, 35(7), 605-617.

[61] Merkle, D., & Middendorf, M. (2000, April). An ant algorithm with a new pheromone evaluation rule for total tardiness problems. *In Workshops on Real-World Applications of Evolutionary Computation (pp. 290-299)*. Springer, Berlin, Heidelberg.

[62] Morrison, D. R., Sewell, E. C., & Jacobson, S. H. (2014). An application of the branch, bound, and remember algorithm to a new simple assembly line balancing dataset. *European Journal of Operational Research*, 236(2), 403-409.

[63] Nakandala, D., Samaranayake, P., & Lau, H. C. (2013). A fuzzy-based decision support model for monitoring on-time delivery performance: A textile industry case study. *European Journal of Operational Research*, 225(3), 507-517.

[64] Otto, A., Otto, C., & Scholl, A. (2013). Systematic data generation and test design for solution algorithms on the example of SALBPGen for assembly line balancing. *European Journal of Operational Research*, 228(1), 33-45.

[65] Ozbakir, L., Baykasoglu, A., Gorkemli, B., & Gorkemli, L. (2011). Multiple-colony ant algorithm for parallel assembly line balancing problem. *Applied Soft Computing*, 11(3), 3186-3198.

[66] Özcan, U., & Toklu, B. (2009). Multiple-criteria decision-making in two-sided assembly line balancing: A goal programming and a fuzzy goal programming models. *Computers & Operations Research*, 36(6), 1955-1965.

[67] Peng, Y., Wang, Y., & Zi, Y. (2019). Switching state-space degradation model with recursive filter/smoother for prognostics of remaining useful life. *IEEE Transactions on Industrial Informatics*, 15(2), 822-832.

[68] Porter, M. E., & Heppelmann, J. E. (2014). How smart, connected products are transforming competition. *Harvard Business Review*, 92(11), 64-88.

[69] Rada-Vilela, J., Chica, M., Cordón, Ó., & Damas, S. (2013). A comparative study of multi-objective ant colony optimization algorithms for the time and space assembly line balancing problem. *Applied Soft Computing*, 13(11), 4370-4382.

[70] Rüßmann, M., Lorenz, M., Gerbert, P., Waldner, M., Justus, J., Engel, P., & Harnisch, M. (2015). Industry 4.0: The future of productivity and growth in manufacturing industries. *Boston Consulting Group*, 9(1), 54-89.

[71] Sabuncuoglu, I., & Bayiz, M. (1999). Job shop scheduling with beam search. *European Journal of Operational Research*, 118(2), 390-412.

[72] Salveson M. E. (1955). The assembly line balancing problem. *Journal of Industrial Engineering*, 6, 18-25.

[73] Sancı, E., & Azizoğlu, M. (2017). Rebalancing the assembly lines: exact solution approaches. *International Journal of Production Research*, 55(20), 5991-6010.

[74] 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(3), 666-693.

[75] Scholl, A. (1993). Data of Assembly Line Balancing Problems. Schriften zur Quantitativen Betriebswirtschaftslehre, 16/1993, TH Darmstadt.

[76] Scholl, A., Fliedner, M., & Boysen, N. (2010). Absalom: Balancing assembly lines with assignment restrictions. *European Journal of Operational Research*, 200(3), 688-701.

[77] Scholl, A., Boysen, N., & Fliedner, M. (2013). The assembly line balancing and scheduling problem with sequence-dependent setup times: problem extension, model formulation and efficient heuristics. *OR Spectrum*, 35(1), 1-30.

[78] Simaria, A. S., & Vilarinho, P. M. (2009). 2-ANTBAL: An ant colony optimisation algorithm for balancing two-sided assembly lines. *Computers & Industrial Engineering*, 56(2), 489-506.

[79] Simona, D. I. N. U. (2015). Multi-objective Assembly Line Balancing Using Fuzzy Inertia-adaptive Particle Swarm Algorithm. *Studies in Informatics and Control*, 24(3), 283-292.

[80] Stork, A. (2015). Visual computing challenges of advanced manufacturing and industrie 4.0 [guest editors' introduction]. *IEEE Computer Graphics and Applications*, 35(2), 21-25.

[81] Tamani, K., Boukezzoula, R., & Habchi, G. (2011). Application of a continuous supervisory fuzzy control on a discrete scheduling of manufacturing systems. *Engineering Applications of Artificial Intelligence*, 24(7), 1162-1173.

[82] Thames, L., & Schaefer, D. (2016). Software-defined cloud manufacturing for Industry 4.0. *Procedia CIRP*, 52, 12-17.

[83] Tsourveloudis, N. C., Dretoulakis, E., & Ioannidis, S. (2000). Fuzzy work-in-process inventory control of unreliable manufacturing systems. *Information Sciences*, 127(1-2), 69-83.

[84] Wan, J., Tang, S., Shu, Z., Li, D., Wang, S., Imran, M., & Vasilakos, A. V. (2016). Software-defined industrial internet of things in the context of industry 4.0. *IEEE Sensors Journal*, 16(20), 7373-7380.

[85] Wang, L. X. (1993). Stable adaptive fuzzy control of nonlinear systems. *IEEE Transactions on Fuzzy Systems*, 1(2), 146-155.

[86] Wang, C., Bi, Z., & Da Xu, L. (2014). IoT and cloud computing in automation of assembly modeling systems. *IEEE Transactions on Industrial Informatics*, 10(2), 1426-1434.

[87] Wee, T. S., & Magazine, M. J. (1982). Assembly line balancing as generalized bin packing. *Operations Research Letters*, 1(2), 56-58.

[88] Yan, H., Wan, J., Zhang, C., Tang, S., Hua, Q., & Wang, Z. (2018). Industrial big data analytics for prediction of remaining useful life based on deep learning. *IEEE Access*, 6, 17190-17197.

[89] Yagmahan, B. (2011). Mixed-model assembly line balancing using a multi-objective ant colony optimization approach. *Expert Systems with Applications*, 38(10), 12453-12461.

[90] Yang, C., Gao, J., & Sun, L. (2013). A multi-objective genetic algorithm for mixed-model assembly line rebalancing. *Computers & Industrial Engineering*, 65(1), 109-116.

[91] Zacharia, P. T., & Nearchou, A. C. (2012). Multi-objective fuzzy assembly line balancing using genetic algorithms. *Journal of Intelligent Manufacturing*, 23(3), 615-627.

[92] Zha, J., & Yu, J. J. (2014). A hybrid ant colony algorithm for U-line balancing and rebalancing in just-in-time production environment. *Journal of Manufacturing Systems*, 33(1), 93-102.

[93] Zheng, P., Xu, X., & Chen, C. H. (2018a). A data-driven cyber-physical approach for personalised smart, connected product co-development in a cloud-based environment. *Journal of Intelligent Manufacturing*, https://doi.org/10.1007/s10845-018-1430-y.

[94] Zheng, P., Lin, Y., Chen, C. H., & Xu, X. (2019). Smart, connected open architecture product: an IT-driven co-creation paradigm with lifecycle personalization concerns. *International Journal of Production Research*, 57(8), 2571-2584.

[95] Zheng, P., Sang, Z., Zhong, R. Y., Liu, Y., Liu, C., Mubarok, K., Yu, S., & Xu, X. (2018b). Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives. *Frontiers of Mechanical Engineering*, 13(2), 137-150.

[96] Zhong, Y. G., & Ai, B. (2017). A modified ant colony optimization algorithm for multi-objective assembly line balancing. *Soft Computing*, 21(22), 6881-6894.

[97] Zhong, R. Y., Xu, C., Chen, C., & Huang, G. Q. (2017). Big data analytics for physical internet-based intelligent manufacturing shop floors. *International Journal of Production Research*, 55(9), 2610-2621.