OPTIMIZATION OF THE PRODUCTION SCHEDULING PROBLEM WITH MAINTENANCE

FU XIAOYUE

PhD

The Hong Kong Polytechnic University

2020

The Hong Kong Polytechnic University

Department of Industrial and Systems Engineering

Optimization of the Production Scheduling Problem with Maintenance

Fu Xiaoyue

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

January 2020

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)

______Fu Xiaoyue__________(Name of student)

Abstract

To improve production efficiency and save cost, production scheduling is usually integrated with maintenance planning. However, in most of the existing research, only the maintenance on machine is considered. On the other hand, mould maintenance is also a vital element in the plastic industry and the breakdowns caused by the mould are even more than the breakdowns caused by the machine. In this connection, it is necessary to integrate mould maintenance into the production scheduling with maintenance problem. Traditionally, a joint scheduling approach is used to deal with the Production Scheduling with Mould Maintenance (PS-MM) problem, which means that the production scheduling, machine maintenance, and mould maintenance are decided jointly. In addition, some algorithms based on Genetic Algorithm (GA) are used to obtain good optimization solutions. Even so, there is still a mismatch among the production scheduling, machine maintenance and mould maintenance in the existing research. Some new strategies and new algorithms need to be explored to provide a satisfactory solution to this problem.

The objective of this research is to analyse and optimize the production scheduling problem with mould maintenance. After analysing the character of the original integrated problem deeply, a problem decomposition mechanism is proposed. Based on this mechanism, a new meta-heuristic algorithm is put forward. Particularly, a multi-maintenance strategy that is seldom included in the traditional integrated problem is considered in the new proposed problem. The maintenance strategy depends on the state of the resource and has different effects on the resource. A Three-Level Particle Swarm Optimization (TLPSO) algorithm is used to solve the problem. Furthermore, uncertainty is firstly considered in the integrated problem. The processing time and the maintenance time are represented by triangular fuzzy numbers instead of determinate numbers. Two objectives, i.e. the robustness and the fuzzy makespan, are optimized.

This research deepens the study of the production scheduling problem with mould maintenance and can provide a vital reference for future research on the PS-MM problem. Overall, three deliverables are generated in this research. Firstly, a Three-level Particle Swarm Optimization with Variable Neighbourhood Search Algorithm (TLPSO-VNS) is proposed to solve the addressed problem and is proven to outperform other existing algorithms. Secondly, based on the problem decomposition mechanism, a Three-Level PSO (TLPSO) algorithm is designed for the multi-state multi-maintenance integrated problem. Numerical experiments have shown the advantages of the mechanism and the algorithm. Thirdly, a Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm is proposed for the fuzzy production scheduling problem considering mould maintenance. The efficiency and effectiveness of the mechanism used in the MOPIO have been proved by the comparison algorithms including Non-dominated Sorting Genetic Algorithm II (adapted NSGA-II) and Multi-

Objective Particle Swarm Optimization (MOPSO) algorithm. The problem decomposition mechanism and the algorithms proposed in this research are instructive for other integrated problems. The future research will focus on proposing some efficient strategies to make some improvement in the optimization performance of algorithms proposed in this research.

Publication Arising from the Thesis

Journal paper

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S. H. and Qu, T., 2019. A multi-objective pigeon inspired optimization algorithm for fuzzy production scheduling problem considering mould maintenance. *SCIENCE CHINA Information Sciences. (IF:2.731)* 62(7): 070202. https://doi.org/10.1007/s11432-018-9693-2

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S.H. and Qu, T., 2019. A three-level particle swarm optimization with variable neighbourhood search algorithm for the production scheduling problem with mould maintenance. *Swarm and Evolutionary Computation (IF:6.330).* 50(2019):100572. https://doi.org/10.1016/j.swevo.2019.100572

Conference paper

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S.H., and Bi Y., 2017. Minimization of makespan through jointly scheduling strategy in production system with mould maintenance consideration. *2017 International Conference on Intelligent Computing*, August 7-10, 2017, Liverpool, UK.

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S.H., 2018a. A three-level particle swarm optimization algorithm approach for production scheduling with mould maintenance problem, *2018 ACENS Asian Conference on Engineering and Natural Sciences*, February 6-8, 2018, Osaka, Japan.

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S.H., 2018b. Minimization of makespan for production scheduling with multi-resources maintenance problem by a hybrid algorithm, *2018 TICEAS The International Conference on Engineering and Applied Sciences*, February 22-24, 2018, Bangkok, Thailand.

Fu, X.Y., Chan, F.T.S., Niu, B., Chung, S.H. and Qu, T., 2020. Scheduling multiple maintenances and production scheduling through a three-level particle swarm optimization algorithm. *BIT's 11th World Gene Convention-2020 (WGC-2020),* June 7-9, 2020, Osaka, Japan.

Acknowledgements

I would like to express my heartfelt gratitude and appreciation to my chief supervisor, Prof. Felix T.S. Chan, Professor of the Department of Industrial and Systems Engineering, the Hong Kong Polytechnic University. Thanks to his professionalism and experience in production scheduling, my research could go in the right direction under his invaluable advice. I am deeply appreciative of his kindness and encouragement. Furthermore, I would like to express my sincere gratitude to my co-supervisors Prof. Ben Niu, Professor of the College of Management, Shenzhen University and Prof. Ting Qu, Professor of School of Electrical and Information Engineering, Jinan University who always provide me with useful and inspiring suggestions. In addition, my thanks must go to Dr. Nick S.H. Chung, for his patient supports and help. Moreover, I would like to give my thanks to all the staff in the Department of Industrial and Systems Engineering, the Hong Kong Polytechnic University.

Particularly, I would like to give my thanks to my family, especially my beloved daughter Zhang Duanyi, who is the sunshine of my life. Without the support and understanding of my family, my Ph. D. study could not be finished. Furthermore, my sincere thanks go to all my friends in the Hong Kong Polytechnic University. The good memory with them is my lifelong wealth.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

GA	Genetic Algorithm
PSO	Particle Swarm Optimization
VNS	Variable Neighborhood Search
PIO	Pigeon Inspired Optimization
MOPIO	Multi-Objective Pigeon Inspired Optimization
MOPSO	Multi-Objective Particle Swarm Optimization
PS-MM	Production Scheduling with Mould Maintenance
FPSP-MM	Fuzzy Production Scheduling Problem with Mould Maintenance
TLPSO	Three-Level Particle Swarm Optimization
TLPSO-VNS	Three-Level Particle Swarm Optimization with Variable Neighbourhood Search
PM	Preventive Maintenance
CBM	Condition-Based Maintenance
MSS	Multi-State System
FJSSP	Fuzzy Job Shop Scheduling Problem
fJSSP	Flexible Job Shop Scheduling Problem
FfJSSP	Fuzzy Flexible Job Shop Scheduling Problem
DAC	Dual-Ants Colony
UGF	Universal Generating Function
NSGA-II	Non-dominated Sorting Genetic Algorithm II

TS	Tabu Search
BRE	Best Error Rate
MA	Memetic Algorithm
DE	Differential Evolution
HICATS	Hybrid Imperialist Competition Algorithm and Tabu Search
NP-hard	Non-deterministic Polynomial hard
DFfJSP	Dynamic Fuzzy Flexible Job Shop Scheduling Problem
PBM	Parallel Batch Processing Machine
SNS	Swarm-based Neighborhood Search
FJSSP-PM	Fuzzy Job Shop Scheduling Problem considering Preventive Maintenance
JSSP	Job Shop Scheduling Problem
PVNS	Population-based Variable Neighbourhood Search
GVNS	General Variable Neighbourhood Search
UAVs	Unmanned Aerial Vehicles
EPF	Edge Potential Function
SAPIO	Simulated Annealing Pigeon Inspired Optimization
PPPIO	Predator-Prey Pigeon-Inspired Optimization
UCAV	Uninhabited Combat Aerial Vehicle
ESN	Echo State Network
GPIO	Gaussian Pigeon Inspired Optimization

SA	Simulated Annealing
DPSO	Discrete Particle Swarm Optimization
SVNS	Stochastic Variable Neighbourhood Search
PHS	Permutation-based Harmony Search
EBVNS	Enhanced Basic Variable Neighbourhood Search
ANN	Artificial Neural Network
GAVNS	Genetic Algorithm with the Variable Neighbourhood Search
NBA	Neighborhood Best Archive
PBA	Personal Best Archive
GBA	Global Best Archive
CD	Crowding Distance
SCD	Special Crowding Distance
PF	Pareto Front
TFN	Triangular Fuzzy Number

Chapter 1. Introduction

This chapter begins with the research background, and it then points out the research motivation. To provide a clear direction for tracking the identified problem, this chapter moves on to the research scope and objectives, followed by the research significance and value. The outline of each chapter of this thesis is given at the end of this chapter.

1.1 Research Background

Because of the special merits of plastics such as high strength, low price, lightweight, and user-friendliness, plastics materials find extensive utilization in many industries and daily life (Shameem et al. 2017). Moreover, the plastics industry greatly contributes to the economy of many countries such as the United States through providing employment to a large number of people and is regarded as the third largest manufacturing industry in the country (Lokensgard 2017). The main process of converting plastics into products is by injection moulding (Figure 1-1), which needs both an injection machine (Figure 1-2) and an injection mould (Figure 1-3). Since consumer demand for plastic products is reflected by the overall growth of plastics sales (Lokensgard 2017), more methods are being explored to improve the resource efficiency of the production process under new visions for the future, for example, the "smart factory" (Dangel 2016). Furthermore, to realize production objectives and avoid production bottlenecks, a production schedule is planned in advance. A good production scheduling can ensure the efficient use of labour resources and guarantee a high machine utilization rate (Christopher 2006). Many good algorithms are proposed for the scheduling problem (Gao et al . (2016), Gao et al. (2017), Gao et al. (2018), Gao et al. (2019)).

In most research studies on production scheduling, it is postulated that machines are always available throughout the whole production planning stage in most research studies on production scheduling. However, in actual situations, this assumption may not be reasonable, because failure may occur at any time, making some machines unavailable for job processing (Rajkumar, Asokan, and Vamsikrishna 2010). So, machine maintenance planning is essential and can enhance the reliability of the system. To maximize the system productivity, maintenance planning and production scheduling must be considered together, and be given the same importance level (Berrichi et al. 2010). More and more scholars are paying attention to production scheduling with the machine maintenance problem, and different models that integrate the production scheduling with machine maintenance planning have been built to optimize productivity by harmonizing both activities.

However, only the maintenance of the machine is considered in traditional production scheduling with the resource maintenance problem and research on production scheduling with the mould maintenance problem is limited. The injection mould is also a significant component in the plastics industry and the breakdowns caused by factors related to mould are even more than breakdowns caused by factors related to machine (Wong, Chan, and Chung, 2012). The plastics manufacturer needs to consider the condition of the mould to guarantee good production efficiency. So the integrated problem with mould maintenance consideration should be given more attention. So far, only a limited number of studies have investigated the production problem with multi-resource maintenance and more efficient algorithms need to be explored to find better solutions for this integrated problem.

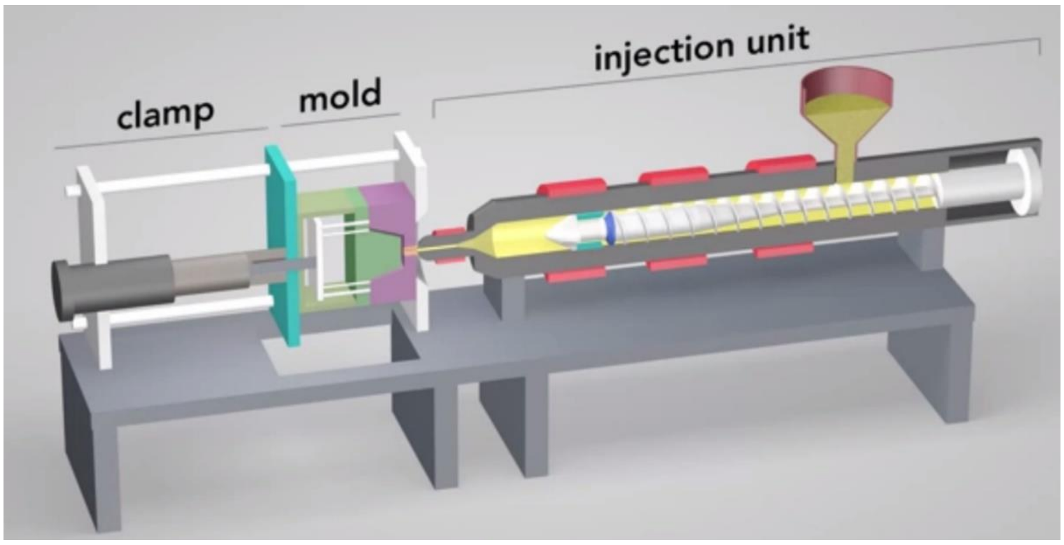


Figure 1-1 Injection moulding

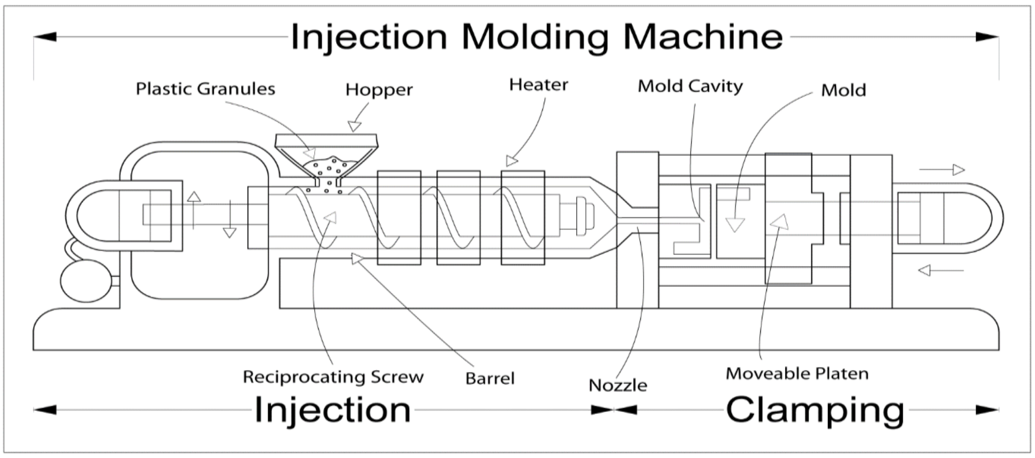


Figure 1-2 Injection moulding machine

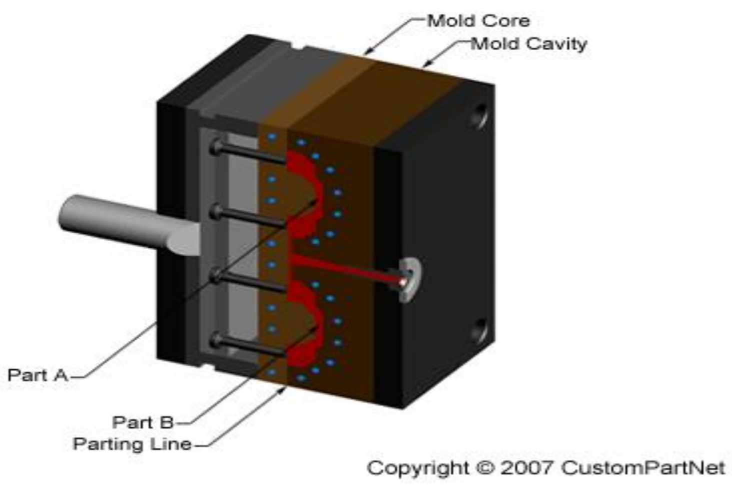


Figure1-3 Injection mould

1.2 Research Motivation

Recently, Wong, Chan, and Chung (2012) built a model to integrate production scheduling with mould maintenance. Time-dependent deteriorating maintenance schemes for the machine and mould were used and a joint scheduling strategy was proposed, which decided production scheduling, machine maintenance, and mould maintenance simultaneously to minimize the overall makespan. The Genetic Algorithm (GA) was used to solve this problem. However, the maintenance planning found may not be the most suitable one for the production scheduling when production scheduling and resource maintenance planning were decided concurrently since the local search ability of the genetic algorithm is limited. The mismatch between production scheduling and resource maintenance is underestimated, which may result in low production efficiency. More strategies need to be proposed to overcome the drawback of a joint scheduling strategy. In addition, more efficient algorithms need to be explored to improve the quality of the solutions to the integrated problem.

Furthermore, in most of the research related to the integrated problem, the maintenance is preventive maintenance and preventive maintenance is usually periodically conducted or related on time. In practice, the cost of the periodical machine maintenance is high and usually unnecessary. Moreover, most of this preventive maintenance is perfect maintenance which means that after the maintenance, the machine is as perfect as new. However, except for perfect maintenance, there are other kinds of maintenances, and these maintenances may bring a machine to a state between as good as new and as bad as old. The machine or the system may have many states and different maintenances may have different effects. The multiple maintenances need to be considered in the integrated problem.

In addition, in the original integrated model, all the parameters are determinate values, such as job information and maintenance parameters. However, practical manufacturing systems may have to deal with imprecise values. Uncertainty could be defined as the situation which involves vague information or unknown information. Uncertainty is the difference between the amount of information required to perform a task and the amount of information already possessed. There are many ways to handle uncertainty. One of the important ways of handling uncertainty is the probability approach. The probability approach can be described as mathematically and interpretatively. While fuzzy sets and the fuzzy logic concept is another common way of handling uncertainty. Since fuzzy logic tries to capture the essential property of vagueness, uncertainty can be better handled than the probability approach.

Therefore, the main motivation of this research is to fill the gaps mentioned above by proposing a new scheduling strategy and developing an efficient optimization algorithm to solve the production scheduling problem with multi-resource maintenance. Moreover, multi-maintenance is considered in the integrated model and these maintenances depend on the states of the resource and have different effects on the corresponding resources. Furthermore, since various factors involved in the integrated problems are often imprecise or uncertain in nature, which is especially true when human-made factors are incorporated into the problems, fuzzy processing times and fuzzy maintenance time will be considered in the integrated model.

1.3 Research Scope and Objectives

A traditional production scheduling problem is to distribute different jobs on different machines to achieve some goals such as minimizing the makespan. While, in practice, resources such as machines are not always available, preventive maintenance on machines is usually considered by scholars but the research on the mould maintenance is limited. The difference between maintenance problems of "mould maintenance" and "machine maintenance" in terms of production scheduling is small. However, when these two maintenance problems are considered simultaneously in the production scheduling problem, the integrated problem will be more complex. This study focuses on the production scheduling problem with multi-resource maintenance in the plastics injection production system. In the practical plastics injection production system, each job has only one operation and the workshop of the system contains more than one machine. The processing time for different machines is the same if the injection mould for the job could be installed properly on these machines. So, this research is to solve the single-operation scheduling problems with identical parallel machines and some new algorithms with excellent performance are developed to find good solutions for integrated problems.

To overcome the shortcoming of previous research and improve the overall production efficiency, this research carries out an in-depth analysis of the production scheduling problem with multi-resource maintenance. The main objective is to propose some innovative algorithms to solve the Production Scheduling with the Mould Maintenance (PS-MM) problems efficiently. Firstly, a hybrid algorithm named as Three-level Particle Swarm Optimization with Variable Neighbourhood Search Algorithm (TLPSO-VNS) will be designed to solve the Production Scheduling with the Mould Maintenance (PS-MM) problem which has already been addressed. Secondly, a Three-Level PSO (TLPSO) algorithm will be designed for the multi-state multi-maintenance integrated problem. Thirdly, a Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm will be proposed for the fuzzy production scheduling problem considering mould maintenance. The details of the sub-objectives are as follows:

(1) To analyse the integrated Production Scheduling with the Mould Maintenance (PS-MM) problem deeply and to propose a problem decomposition mechanism, different from the joint scheduling approach, which is traditionally used. The Production Scheduling with the Mould Maintenance (PS-MM) problem will be firstly solved by another heuristic algorithm—Particle Swarm Optimization (PSO) algorithm, different from Genetic Algorithm (GA) which is traditionally used to solve the production scheduling problem with multi-resource maintenance and a new encoding and decoding method of particles will be defined. Furthermore, an effective and efficient hybrid metaheuristic algorithm (Three-level Particle Swarm Optimization with Variable Neighbourhood Search Algorithm) building on the advantages of these metaheuristic components and overcoming the inherent limitations will be designed.

(2) To extend the original integrated problem into a production scheduling with the multi-maintenance problem. Particularly, the maintenance depends on the states of the resource and different maintenances have different influences on the states of the resource. An algorithm (Three-level Particle Swarm Optimization Algorithm) based on the problem decomposition mechanism is to be developed to

solve the production scheduling problem with the multi-maintenance problem.

(3) To build a new Production Scheduling with the Mould Maintenance (PS-MM) problem considering the fuzzy processing time and fuzzy maintenance, except the fuzzy makespan, another objective will be considered. Moreover, a new multi-objective algorithm (Multi-Objective Pigeon Inspired Optimization algorithm) to solve the fuzzy problem will be designed, some new mechanisms will be added into the existing algorithm making it more efficient.

1.4 Research Significance and Contribution

To improve production efficiency and save cost, production scheduling is usually integrated with maintenance planning. However, in most of the existing research, only the maintenance on machine is considered. On the other hand, mould maintenance is also a vital element in the plastic industry and the breakdowns caused by the mould are even more than the breakdowns caused by the machine. In this connection, it is necessary to integrate mould maintenance into the production scheduling with maintenance problem. Traditionally, a joint scheduling approach is used to deal with the Production Scheduling with Mould Maintenance (PS-MM) problem, which means that the production scheduling, machine maintenance, and mould maintenance are decided jointly. In addition, some algorithms based on Genetic Algorithm (GA) are used to obtain good optimization solutions. Even so, there is still a mismatch among the production scheduling, machine maintenance and mould maintenance in the existing research. Some new strategies and new algorithms need to be explored to provide a satisfactory solution to this problem.

This research analyses and optimizes the production scheduling problem with mould maintenance. After analysing the character of the integrated problem deeply, a problem decomposition mechanism is proposed. Based on this mechanism, a new meta-heuristic algorithm is put forward. Particularly, a multi-maintenance strategy that is seldom included in the traditional integrated problem is considered in the new proposed problem. The maintenance strategy depends on the state of the resource and has different effects on the resource. A Three-Level Particle Swarm Optimization (TLPSO) algorithm is used to solve the problem. Furthermore, uncertainty is firstly considered in the integrated problem. The processing time and the maintenance time are represented by triangular fuzzy numbers instead of determinate numbers. Two objectives, i.e. the robustness and the fuzzy makespan, are optimized.

This research deepens the study of the production scheduling problem with mould maintenance and can provide a vital reference for future research on the PS-MM problem. Overall, three deliverables are generated in this research and the contribution is given as follows:

(1) A Three-level Particle Swarm Optimization with Variable Neighbourhood Search Algorithm (TLPSO-VNS) is proposed to solve the addressed problem and is

proven to outperform other existing algorithms.

(2) Based on the problem decomposition mechanism, a Three-Level PSO (TLPSO) algorithm is designed for the multi-state multi-maintenance integrated problem. Numerical experiments have shown the advantages of the mechanism and the algorithm.

(3) A Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm is proposed for the fuzzy production scheduling problem considering mould maintenance. The efficiency and effectiveness of the mechanism used in the MOPIO have been proved by the comparison algorithms.

The problem decomposition mechanism and the algorithms proposed in this research are instructive for other integrated problems not only the production scheduling problem with maintenance.

1.5 Organization of the Research

After a brief introduction of the research background, motivation, scope, and objective, as well as research significance and contribution in Chapter 1, the rest of this thesis is organized as follows:

Chapter 2 is a review of the literature on this topic, which provides an understanding of the previous research in this area including the production scheduling problem with machine maintenance, the production scheduling problem with multi-resource maintenance, the production scheduling problem with uncertainty and the corresponding meta-heuristic algorithms including Genetic Algorithm (GA), Particle Swarm Optimization (PSO) algorithm, Variable Neighbourhood Search (VNS) algorithm and Pigeon Inspired Optimization (PIO) algorithm, etc, as well as providing a rationale for the choice of the topic in the present study.

Chapter 3 presents the hybrid algorithm for the Production Scheduling problem with Mould Maintenance (PS-MM). In this chapter, the basic integrated problem is described, followed by the primary work done on this integrated problem. Then, the hybrid algorithm for the integrated problem is given including the overall algorithm description, encoding and decoding of the particles in the Three-Level PSO, swarm initialization and swarm improvement through Three-Level PSO, and the intensification phase via VNS. Furthermore, numerical experiments are shown. Finally, this chapter ends with conclusions and summary.

Chapter 4 introduces the Three-Level PSO algorithm to solve the production scheduling problem with multiple maintenances. Firstly, the problem description is given; Secondly, the process of the algorithm design is shown. Thirdly, numerical experiments are conducted to illustrate the advantages of the proposed Three-level

Particle Swarm Optimization (PSO) algorithm. Finally, the conclusions and the summary are given.

Chapter 5 demonstrates the Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm for the Fuzzy Production Scheduling problem with Mould Maintenance (FPS-MM). In the problem description, the content related to the arithmetic operations on triangular fuzzy numbers, uncertain maintenance time, objective measure and Pareto domination relationship are given; Then, in the algorithm design part, the basic pigeon inspired optimization algorithm, encoding and decoding of pigeons and Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm are shown. Finally, numerical experiments are conducted to illustrate the advantages of the proposed Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm by comparing it with other algorithms. Furthermore, the results, discussion, and summary are given.

Chapter 6 draws the conclusions, shows the contributions of the research and then points out the limitations of the existing research. After analysing the problem deeply and conducting the literature review, the future direction is shown.

Chapter 2. Literature Review

Chapter 2 begins with the literature review on the production scheduling problem with the machine maintenance problem, which shows the advantages of integrating the production scheduling problem with the maintenance planning problem. The study on production scheduling problem with multi-resource maintenance will then be presented. Furthermore, existing research about production scheduling with uncertainty is analysed, followed by a discussion of meta-heuristic algorithms including Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Variable Neighbourhood Search (VNS) and Pigeon Inspired Optimization (PIO) algorithm, etc. Then, the research gap is identified, and this chapter ends with the summary. This chapter provides an overview of the past study in this field and gives the reason for the choice of the topic and the algorithm in the present study.

2.1 Production Scheduling Problem with Machine Maintenance

Manufacturers are forced to improve their efficiency because of the growing expectation of customers and fierce competition. As one of the most vital elements in many industries, maintenance planning has an explicit effect on the performance of production (Guner, Chinnam, and Murat 2016). Traditionally, decisions on production scheduling are independent with decisions on preventive maintenance planning although they are interdependent. Nowadays, more and more researchers try to consider the interdependency and explore more efficient methods to optimize different production targets. To minimize the total weighted tardiness, an integrated model was proposed (Cassady and Kutanoglu 2003) which determines preventive maintenance planning, as well as production scheduling simultaneously, and numerical experiments showed the effectiveness of this integration. In addition, another integrated model (Cassady and Kutanoglu 2005) was proposed which harmonizes single-machine scheduling decisions with decisions on maintenance planning aiming at minimizing the overall expected weighted completion time of jobs, and they pointed out that integrated production scheduling with the preventive maintenance planning problem is a meaningful research area. Furthermore, Seidgar et al. (2016) investigated the two-stage assembly flow shop problem with preventive maintenance activities and machine breakdowns aiming at minimizing the makespan, and they presented two meta-heuristic algorithms. Moreover, El Khoukhi et al. (2017) proposed two new models to formulate the flexible job shop scheduling problem with machine unavailability restrictions because of preventive maintenance, and a new efficient hybrid algorithm named "Dual-Ants Colony" (DAC) was developed to minimize the makespan. All these studies show that it is worthwhile to conduct research on the integrated production scheduling problem with maintenance planning.

However, in most of the research related to the integrated problem, the maintenance is preventive maintenance and preventive maintenance is usually periodically conducted or related on time. In practice, the cost of the periodical machine maintenance is high and usually unnecessary. Moreover, most of this preventive maintenance is perfect maintenance which means that after the maintenance, the machine is as perfect as new. However, except for perfect maintenance, there are other kinds of maintenance, and these maintenances may bring a machine to a state between as good as new and as bad as old. The machine or the system may have many states and different maintenances may have different effects. A lot of research has been done about the effect of maintenances. Duan et al. (2018) invested the selective maintenance model considering multiple maintenances with stochastic maintenance quality. Moreover, Nguyen, et al. (2017) used a model that considers infinite memory in the age arithmetic reduction to character the repair efficiency of maintenance and three methods including a dynamic a static, and a failure limit policy was put forward. Le and Tan (2013) studied the inspection-maintenance planning for the multi-state system and the maintenance was assumed to be imperfect maintenance and continuous-time. Markov process was used to describe the degradation. The strategy about the inspection and optimal maintenance was delivered. Liu et al. (2013) optimized the multi-state systems by proposing a joint redundancy and imperfect maintenance strategy. The replacement strategy was considered except the redundancy levels aiming at minimizing the average expenditure. Moreover, a regular imperfect repair model was proposed. Condition-Based Maintenance (CBM) was studied by Do et al. (2015). The maintenance includes perfect maintenance, as well as imperfect maintenance. The influence of imperfect maintenance was investigated. And the adaptive maintenance was proposed. Liu and Huang (2010) applied the stochastic process model to evaluate the states' distribution of the Multi-State System (MSS) and used a quasi-renewal process to simulate the stochastic states of the elements in the multi-state system after repair. An optimal replacement policy was proposed. For Multi-State Systems (MSS), a selective maintenance strategy was explored. A cost-maintenance quality relationship was established and the probability that the repaired MSS finishing the following mission was obtained with the help of the Universal Generating Function (UGF) method. Moreover, for complex multicomponent systems, the repairperson assignment problem was integrated with the joint selective maintenance by Diallo, et al. (2018). They investigated two nonlinear formulations and resolved two built binary integer programming models. Pham and Wang (1996) summarized the different optimal policies and treatment methods on imperfect maintenance and some significant outcomes were given. Furthermore, a mathematical model with a numerical algorithm was developed by Ben et al. (2016) to find suitable imperfect preventive maintenance to minimize the cost of leased equipment over a finite lease period. In addition, for a production process with a random breakdown, Liao and Sheu (2011) built a model that considers both economic production quantity and periodic preventive maintenance. Both imperfect Preventive Maintenance (PM) and perfect Preventive Maintenance (PM) were considered. Besides, Olorunniwo and Izuchukwu (1991) built mathematical models to find good schedule solutions for overhaul maintenance as well as preventive maintenance with maintenance improvement factors considered. Moreover, for one protection system, Berrade, Cavalcante, and Scarf (2012) discussed the replacement and imperfect inspection policy and two models were developed.

In the integrated problem, different maintenance strategies on machines and the effects on production scheduling are also explored by scholars. Ruiz et al. (2007) studied different machine preventive maintenance policies regarding flow shop problems aiming at maximizing the availability of machines. An easy criterion was proposed to decide the production sequence and different maintenance operations. The importance of integration was shown. Furthermore, Aghezzaf and Najid (2008) examined integrating the production and preventive maintenance problem in a system that was made up of production lines that are parallel and at risk of failure. The minimal repair is assumed to be conducted when a production line breaks down and perfect preventive maintenance is conducted periodically, and the period is decided by the decision-maker. Then, different models were built depending on different maintenance policies. To minimize the overall makespan, Chung et al. (2009) developed a genetic algorithm with a double tier approach to address the problem under production networks with the multi-factory environment, which schedules perfect and imperfect maintenance on machines concurrently to keep the system reliability at an acceptable level. In addition, Li et al. (2017) explored a parallel-machine scheduling problem where periodic maintenance cycles of the machines are machine-dependent but not the same.

Recently, random failure was also considered in the integrated problem. Wang and Liu (2016) studied an optimisation problem that integrated production scheduling with maintenance under a two-machine flow shop environment aiming to minimize the makespan. The Weibull probability distribution was used to describe the failure time of each machine and four Genetic Algorithms (GA) based heuristics were proposed. Furthermore, Abdelrahim and Vizvari (2017) considered the case in which random failures occurred on a single machine in the integrated problem. The machine failure probability is supposed to be positively related to the age as well as the time interval length, and they only considered perfect maintenance. The consideration of different maintenance strategies and random failures makes the integrated model more complicated but more in line with the actual situation.

2.2 Production Scheduling Problem with Multi-Resource Maintenance

However, in the plastics industry, the mould is also a crucial element that guarantees the normal production and needs to be considered. Some studies have been conducted into the production scheduling problem with multi-resource maintenance. Wong, Chan, and Chung (2012) built a model to integrate production scheduling with mould maintenance and proposed a joint scheduling strategy to minimize the overall makespan under the assumption that the maintenance schemes are time-dependent. Besides, They (Wong, Chan, and Chung 2013) considered a more complicated case in which there are multiple resources and maintenance tasks in the integrated problem and proved that the proposed jointly scheduling method can reduce the makespan significantly. Furthermore, Wong, Chan, and Chung (2014) studied a new integrated problem that each job contains multiple operations with multiple moulds. In addition, Wang and Liu (2015) investigated a multi-objective parallel machine scheduling problem with flexible preventive maintenance on the machine and mould and aimed to minimize the makespan and unavailability of machine and mould, and a multi-objective integrated optimization method with NSGA-II adaption was presented. Also, to show the influence of the mould on production scheduling, Shen et al. (2016) built an optimization model, considering setup and mould maintenance, aiming at minimizing the weighted tardiness and earliness. Different maintenance strategies were considered, and the optimal solution was found by a genetic algorithm. It is still a key issue for some industries to integrate production scheduling with mould maintenance, but research on this topic is still limited and more efficient algorithms for solving such an integrated problem need to be developed.

2.3 Production Scheduling Problem with Uncertainty

The processing time and the maintenance time are fixed values for the regular scheduling problems. However, the maintenance time and the processing time are just uncertain values in the majority of real industries, since practical problems are usually affected by many factors, especially the human-related factors are involved. This situation may make the completion time uncertain. In this connection, the concept of uncertainty needs to be introduced to integrated scheduling problems. Fuzzy sets and the fuzzy logic concept is a common way of handling uncertainty since 1965. Because fuzzy logic tries to capture the essential property of vagueness, uncertainty can be better handled than the probability approach. A lot of research on fuzzy production scheduling has been conducted.

2.3.1 Fuzzy Production Scheduling Problem

As one combinational optimization problem, Fuzzy Job Shop Scheduling Problem (FJSSP) is recognized to be non-deterministic and polynomial-hard. Recently, a lot of research has been conducted on the FJSSP and many algorithms are proposed to solve them. Kuroda and Wang (1996) applied a branch-and-bound algorithm to address both dynamic and static job shop scheduling problems considering fuzzy due dates and/or fuzzy operation times. The fuzzy theory was applied to the practical job shop scheduling problem. Then, Sakawa and Mori (1999) considered a fuzzy due date, as well as fuzzy processing time into the job-shop scheduling problems. Genetic Algorithm (GA) was proposed aiming at maximizing the minimum agreement index. After comparing it with simulated annealing, the effectiveness and feasibility of the presented method were shown. Furthermore, the job-shop scheduling problem considering fuzzy operation time was studied by Lu and Cheng (2010b). To minimize the makespan, the Genetic Algorithm (GA) and the DNA evolutionary algorithm are integrated into a new algorithm named as hybrid DNA evolutionary algorithm to obtain good scheduling solutions. Furthermore, Particle Swarm Optimization (PSO) is a hybrid with Tabu Search (TS) to address the same problem by Li, and Pan (2013). To prove the fact that quality initial population can affect meta-heuristics convergence speed, Shaheed, Shukor, and Nababan (2016) proposed a population quality and Best Error Rate (BRE) to gauge the performance of initialization methods for Fuzzy Job-Shop Scheduling Problem (FJSSP), namely, random-based and priority rules-based methods, which was used in a Memetic Algorithm (MA).

Since FJSSP is a hot topic, some researchers conducted some literature review on it. Abdullah and Abdolrazzagh-Nezhad (2014) reviewed the classification of FJSSPs, restrictions, and objectives considered in FJSSPs, and the approaches used to solve FJSSPs, and they gave some possible suggestions for future studies. Moreover, Palacios et al. (2016) did some survey on the existing benchmarks for the fuzzy job shop aiming at minimizing the makespan. They also analyzed the diverse of these instances and the improvement space. Moreover, some more tough benchmark problems were proposed. For each instance, the lower bounds of the makespan and the referral makespan produced by the memetic algorithm were provided.

Fuzzy Job Shop Scheduling Problem (FJSSP) with multiple objectives is also explored by researchers. Sakawa and Kubota (2000) proposed a fuzzy job shop scheduling problem aiming at maximizing the minimum agreement index and the average agreement index and minimizing the maximum fuzzy completion time. Besides, Lei (2008) solved the job shop scheduling problem considering the fuzzy due date, processing time and multiple objectives. A new representation method based on priority rule was proposed, and the problem was transferred into a continuous optimization problem. Moreover, Hu, Yin, and Li (2011) investigated a similar problem and they developed a new possibility and necessity measures based ranking method of fuzzy numbers. Several novel objective functions were proposed, and a Differential Evolution (DE) algorithm was modified to address these objective functions. Furthermore, Lei (2010c) resolved a job shop scheduling problem considering fuzzy trapezoid or doublet due date and fuzzy processing time. To maximize the minimum agreement index and to minimize the maximum fuzzy completion time, a Genetic Algorithm (GA) which was integrated with a random key approach was proposed and shown to be efficient. A hybrid algorithm named as Hybrid Imperialist Competition Algorithm and Tabu Search (HICATS) was developed by Wang et al. (2016), combining tabu search with discrete imperialist competition algorithm to solve FJSP considering fuzzy processing time and fuzzy due date. The minimum agreement index, which relies on the agreement index of the fuzzy completion time and fuzzy due date is the objective function to be maximized. In addition, Gao, Suganthan and Pan et al. (2016) put forward an improved artificial bee colony algorithm for FJSP considering fuzzy processing time aiming at minimizing the maximum fuzzy completion time and the maximum fuzzy machine workload, respectively. José Palacios et al. (2017) tackled a variant of the job shop scheduling problem that fuzzy numbers were used to model undetermined task durations aiming at maximizing its robustness and minimizing the fuzzy makespan. A novel dominance-based tabu search method was integrated into the multi-objective evolutionary algorithm.

Other different models on the processing time are proposed by scholars except for the fuzzy processing time. Interval processing time was firstly considered in the job shop scheduling problem by Lei (2011). And then a neighbourhood search based on population was proposed to optimize the interval makespan. Moreover, decreasing time-dependent job processing time was considered a flow shop scheduling problem by Wang, Wang, and Wang (2011). The processing time was assumed to be a decreasing function of its execution starting time. Besides, processing time variability was considered in the job-shop scheduling problem by Yang et al. (2014) and the processing time variability is represented by the interval number method.

In the last decades, the classical Job Shop Scheduling Problem (JSP) is extended into the Flexible Job Shop Scheduling Problem (fJSSP), which is more in line with the practical conditions and its application is wide. The parallel machine production environment is considered in the fJSSP. Many different operations could be conducted by the same machine. The routes of operations may be the difference as long as they are processed by the machines that are available. There are two problems in the fJSSP: routing and scheduling. The main objective of the routing problem is choosing one suitable machine from all the machines that are available for each operation. The aim of the scheduling problem is to give a suitable sequence for operations assigned on each machine. The target of these two problems is to get a good schedule that could realize some scheduling objectives. It is more difficult to deal with the fJSSP than classical JSSP because of its complexity. The fJSSP is also Non-deterministic Polynomial hard (NP-hard). To resolve the Flexible Job Shop Scheduling Problem (fJSSP), much research has been conducted. And some approaches including exact methods and hybrid techniques are presented. Chaudhry and Khan (2016) presented the evolution of fJSSP, and the different methodologies used to solve the problem were investigated since 1990. Based on their investigation, they drew the conclusions.

The flexible job-shop scheduling problem considering fuzzy processing time is a combination of fuzzy scheduling and flexible scheduling in the job shop environment, which describes the real-world condition better. Because the fuzzy processing time is considered, algorithms proposed for the original problem may be no longer satisfactory and some modifications could be made. To hand fuzzy processing time, models were built related to fuzzy flexible job shop in numerous studies and this research has drawn the attention of many researchers recently.

Different algorithms were proposed for the Fuzzy Flexible Job Shop Scheduling Problem (FfJSSP) aiming at minimizing the fuzzy makespan such as decomposition-integration genetic algorithm (Lei 2010b), swarm-based neighbourhood search algorithm (Lei and Guo 2012), co-evolutionary genetic algorithm (Lei 2012), hybrid artificial bee colony algorithm (Wang et al. 2013), discrete harmony search algorithm (Gao et al. 2015), hybrid algorithm which combined the biogeography-based optimization and some heuristics (Lin 2015), effective estimation of distribution algorithm (Wang et al. 2013), genetic algorithm hybridized with tabu search and heuristic seeding (Palacios et al. 2015), improved version of discrete particle swarm optimization (Huang et al. 2016), improved artificial bee colony algorithm (Gao et al. 2015), teaching-learning-based optimization algorithm (Xu et al. 2015). Furthermore, flexible flow shop problems using the fuzzy approach were addressed by an efficient algorithm proposed by Khademi Zare and Fakhrzad (2011) aiming at minimizing the total job tardiness. They gave an assumption that the operation times are different on parallel machines. And parameters such as "operation time" and "due date" follow a triangular fuzzy number. An attribute-deductive tool for example data mining was integrated into the genetic algorithm to obtain a near-optimal solution.

Because of the complex situations in the workshop, the requirements such as dynamicity, complexity, and instantaneity in problems related to job-shop scheduling could not be satisfied by the majority of the traditional scheduling algorithms. Dynamic scheduling algorithms were proposed, which are more suitable for the practical conditions when the static algorithms are used as comparison algorithms. Since fuzzy processing time and flexibility are normal in practice, Liu, Fan, and Liu

(2015) focused on the dynamic flexible job-shop scheduling problem with fuzzy processing time. They simplify the original problem into a traditional static fuzzy flexible job-shop problem through using some transforming procedures. A fast estimation of distribution algorithm was developed. Gao, et al. (2016) addressed a flexible job shop scheduling problem with two constraints, namely new job insertion and fuzzy processing time aiming at minimizing the maximum fuzzy completion time. A two-stage artificial bee colony algorithm with several improvements was presented to address the new job insertion and fuzzy processing time constraints.

Fuzzy flexible job shop scheduling problem with multi-objectives is also studied by researchers. Liu et al. (2011) proposed a multi-objective fuzzy flexible job shop scheduling model under the assumption that the due date is subject to fuzzy time window distribution. The objective is to maximize customer satisfaction and minimize completion time. Besides, the crossbar collaborative multi-group genetic algorithm was proposed. A multi-objective swarm-based neighbourhood search algorithm was proposed by Zheng, Li, and Lei (2012) to address the fuzzy flexible job scheduling problems aiming at minimizing makespan and maximizing machine workload. Wang et al. (2017) constructed an effective memetic algorithm for simultaneously optimizing fuzzy makespan, minimal agreement index, and average agreement index.

Except for the fuzzy processing time, other uncertainty also considered by researchers.

Controllable process times and machine flexibility were considered in the flexible job shop scheduling problem by Hadi and Mehrdad (2015). In their model, extra resources' consumptions could control the operations' processing times. Aiming at minimizing the total costs, the best balance between delay cost and processing cost was found. A scatter search method was developed. Moreover, controllable processing time was integrated into the multi-objective model for fJSP by Lu et al. (2017) and a new multi-objective discrete virus optimizer was presented to resolve this problem. Furthermore, the Parallel Batch Processing Machine (PBM) was considered in the flexible job-shop scheduling problem by Ham (2017). By assuming that processing time by the same machine for different jobs are the same, job-independent processing time is used to replace the job-dependent processing time.

Furthermore, Wang et al. (2015) studied the flexible job shop scheduling problem considering uncertain processing time in a multi-type and low-volume environment. A minimax regret based robust scheduling model was built aiming at minimizing the makespan. An original sequential search rule was put forward to reduce the calculation amount of the algorithm and a two-stage genetic algorithm was designed to figure out the redundant and optimal solutions. The results demonstrate that the performance of the proposed algorithm is better than the genetic algorithm on flexible job-shop scheduling problem under uncertain and dynamic environment. Bounded processing times and transportation constraints were considered in the model built for the flexible job shop scheduling problem by Zhang (2012) aiming at minimizing the storage of solutions and the makespan. Tabu search procedure was integrated into the genetic algorithm to resolve the assignments of resources as well as sequencing problems on every resource. To resolve the problems with fuzzy factors, the interval-valued trapezoidal fuzzy number was firstly considered in the fuzzy flexible job-shop scheduling model introduced by Lu and Cheng (2010a) and a hybrid PSO algorithm was proposed.

2.3.2 Fuzzy Production Scheduling Problem with Maintenance

Preventive Maintenance (PM) are intergrade into numerous scheduling problems, but, scheduling problem with PM under a fuzzy situation are rarely explored. The running machines also require maintenance that conducted preventively to guarantee a good situation under the fuzzy condition. Fuzzy due-date, non-resumable jobs, and periodic maintenance were considered in the fuzzy job shop scheduling problem considering the availability. Lei (2010a) studied the restrictions aiming at maximizing the minimum agreement index. A genetic algorithm with a random key approach was designed for the problem. Besides, the genetic algorithm with a random key approach was used to address the problem considering PM in the fixed time intervals and resumable jobs by Zheng, Li, and Lei (2010). In their research, an original random key representation approach, discrete crossover, and a novel decoding strategy with maintenance operation incorporated were used. The optimization performance of the genetic algorithm with the random key method in addressing fuzzy scheduling considering PM was shown by the computational results. Furthermore, an effective algorithm named as Swarm-based Neighborhood Search (SNS) was introduced by Lei (2011) to resolve FJSSP-PM that contains m machines and n jobs which are resumable. Some approaches from the existing literature are chosen as the comparison algorithms to show the efficiency of the presented SNS algorithm. Finally, the results illustrate that the developed SNS algorithm surpasses other algorithms when addressing the fuzzy scheduling problem considering PM. Moreover, a hybrid chemical-reaction optimization algorithm was presented by Li, and Pan (2013) to address the job-shop scheduling problem considering fuzzy processing time. In their research, both non-resumable and resumable situations are considered for flexible maintenance such that the integrated problem is more in line with the real world. The convergence capability of the proposed algorithm is increased by integrating the local search based on Tabu Search (TS).

2.4 Meta-heuristic Algorithms

Optimization problems, aiming at finding the effective assignment of finite resources to satisfy some desired purpose, are normal in different research fields. Traditional exact optimization approaches namely branch-and-bound, the simplex method, cutting plane methods, dynamic programming, or approximation algorithms are able to find exactly optimal solutions, but the time required to resolve the problems grows significantly when the scale of the problem increases, and these exactly algorithms may not suffice to solve very large instances. Therefore, the meta-heuristic algorithm shows its advantage to resolve problems with a large scale using a computational cost that is reasonable. Meta-heuristic algorithms, for example, Genetic Algorithms (GA), Variable Neighbourhood Search (VNS) algorithm, Particle Swarm Optimization (PSO) algorithm, etc. are extensively applied to optimize problems with enormous size which includes production scheduling problems.

2.4.1 Genetic Algorithm

Genetic Algorithm (GA) was firstly developed by Holland (1975) in the machine learning field. It is a stochastic search algorithm that imitates the mechanism of natural genetic evolution in biological systems. In most cases, a specific chromosome is required to be designed to represent the solution of the individual problem and the performance of the Genetic Algorithm (GA) will be affected by the genetic operator. As one of the most popular algorithms, GA is known as a suitable and effective method to deal with a scheduling problem. In recent decades, researchers have conducted a lot of research on the application of the Genetic Algorithm (GA) in addressing practical problems.

A Genetic Algorithm (GA) based heuristic approach was presented by Pang (2013) to resolve the no-wait flow shop scheduling problem with two machines and the setup time on the machines depends on the class aiming at minimizing the maximum lateness of the jobs. To test the performance of the presented algorithm, a large number of numerical experiments are done, and results show that for this kind of problem, the proposed methodology is efficient to produce a good scheduling solution in practical application. The green genetic algorithm was proposed to connect sustainability with multi-objective problems assessment by May et al. (2015) and this algorithm is proven to have the ability to obtain a semi-optimal makespan which is much the same as the solution acquired by some methodizes with good performance, while the total energy consumption of the proposed algorithm is significantly lower. Furthermore, some scholars try to add some strategies in the basic genetic algorithm to improve the search ability. An object-coding Genetic Algorithm (GA) is developed by Zhang and Wong (2015) for integrated process planning and scheduling problem. They presented an effective object-coding representation and its corresponding genetic operations, and real objects like machining operations are directly used to represent genes. Moreover, a Genetic Algorithm (GA) was presented by Amir-Mohammad et al. (2016) to address the scheduling problem of one single machine with pre-emption in jobs, aiming at minimizing various objectives, for example, earliness, work in process, and tardiness. Three methods were designed: competition sampling, random sampling, and roulette wheel sampling. The numerical results showed that when the scale of the problem is small, competition sampling could obtain optimal solutions. When the scale of the problem is large, the competition sampling could acquire near-optimal solutions of higher quality using comparatively lower computing costs when other sampling approaches are compared with it. An efficient genetic algorithm considering the new island model was presented by Kurdi (2016) to resolve the popular job shop scheduling problem aiming at minimizing the makespan. Many innovative strategies were proposed to enhance the algorithm. Moreover, a joint optimization model was presented by Xiao et al. (2016) aiming at minimizing the total cost which includes maintenance cost for random failures, preventive maintenance cost, tardiness cost, and production cost. Genetic Algorithm (GA) with random keys approach was designed to resolve the NP-hard problem. A multi-objective genetic algorithm was developed by Zhang and Chiong (2016) and two local search procedures that were problem-specific were incorporated into the algorithm to resolve the job shop scheduling problem with two objectives. Aiming at enhancing the quality of the solution, these local improvement strategies are based on some mathematical models and these mathematical models are formulated for two constricted subproblems that are obtained from the primary problem. The advanced evolution of a Genetic Algorithm (GA) was presented by Dao, Abhary, and Marian (2017) for the integrated production planning and scheduling problem. They also developed GA with new characters such as new algorithm structure, novel chromosome encoding, original crossover, mutation, and selection operator. Since the Genetic Algorithm (GA) was introduced, it has been very popular to solve the production scheduling problem. However, more and more algorithms are proposed and some of them can achieve solutions with better quality.

2.4.2 Particle Swarm Optimization Algorithm

As an intelligent optimization algorithm inspired by the social behaviour of animals, Particle Swarm Optimization (PSO) was firstly presented by Kennedy and Eberhart

(1995). The main characteristic of PSO is that the PSO algorithm does not utilize some filtering procedures (for example, crossover and mutation), and the members of the whole population are obtained by sharing information socially among individuals and directing the search in the direction of the leading location among the search space which makes it evidently different from other heuristic algorithms. Because of attractive features, such as easy execution, few parameters, and fast convergence, PSO has been applied in a wide variety of optimization problems, including the job shop scheduling problem.

Recently, more and more researchers have modified PSO to improve its performance. Lian, Gu, and Jiao (2006) proposed a similar particle swarm optimization algorithm aiming at minimizing makespan for the flow-shop scheduling problem and the outcome shows that the developed algorithm is more efficient than standard GA. And then, they added some new valid algorithm operators into the developed similar particle swarm optimization algorithm and applied the new algorithm to solve the Job Shop Scheduling Problem (JSSP) aiming at minimizing the makespan. Later, they (Lian, Gu, and Jiao 2008) proposed a novel particle swarm optimization algorithm to resolve the permutation flow-shop scheduling problem aiming at minimizing makespan and showed the effectiveness of the new algorithm. Furthermore, aiming at minimizing the overall weighted completion time for the hybrid flow shop scheduling problem, Tang and Wang (2010) proposed an improved particle swarm optimization algorithm. One solution is represented by one job permutation. To obtain an entire hybrid flow shop schedule from one job permutation, a greedy method is used.

2.4.3 Variable Neighbourhood Search Algorithm

The Variable Neighbourhood Search (VNS) algorithm was proposed by Mladenović and Hansen (1997), which was an effective and simple metaheuristic algorithm based on the systematic change of neighbourhoods. Since it is proposed, it has been used for many production scheduling problems.

The scheduling problem considering identical parallel machines and two conflicting objectives: total tardiness and makespan was resolved by Liang and Tien (2011) through Variable Neighbourhood Search (VNS) algorithm. Moreover, a Variable Neighbourhood Search (VNS) algorithm based on an integrated approach was proposed by Bagheri and Zandieh (2011) to resolve the flexible job-shop scheduling problem that setup times depend on the sequence-aiming at minimizing mean tardiness and makespan. The stop condition of the algorithm is controlled by the external loop and the search process is executed by the internal loop. In addition, Bathrinath et al. (2015) used a Simulated Annealing algorithm (SA) and Variable Neighbourhood Search algorithm (VNS) to find near-optimal solutions for scheduling problems considering identical parallel machines aiming at minimizing makespan and the number of tardy jobs. The results showed that the VNS-based heuristics are better than the SA-based heuristics. Moreover, some new algorithms based on VNS were also developed. A mixed-integer nonlinear programming model was developed by Mokhtari, Mozdgir, and Abadi (2012) to describe the joint production and maintenance scheduling considering multiple preventive maintenances problems and a Population-based Variable Neighbourhood Search (PVNS) algorithm was proposed to resolve this problem. The result illustrated that the proposed PVNS was better than traditional algorithms, particularly for problems with large size. Lan et al. (2016) proposed an improved variable neighbourhood search algorithm to address the robust Job Shop Scheduling (JSS) problem to minimize the makespan. They applied discrete scenarios to describe uncertain processing and a new robust optimization model is built. Furthermore, Komaki and Malakooti (2017) addressed the distributed no-wait flow shop scheduling problem with several parallel identical factories. A new algorithm named General Variable Neighbourhood Search (GVNS) is proposed. So far, VNS is still one of the very popular and efficient algorithms in dealing with production scheduling problems.

2.4.4 Pigeon Inspired Optimization Algorithm

The Pigeon Inspired Optimization (PIO) algorithm was firstly proposed by Duan and Qiao (2014) and is a novel bio-inspired swarm intelligence optimizer mimicking the homing characteristics of pigeons. There are two main operators of the PIO algorithm: the map and compass operator, and the landmark operator. Since the PIO algorithm was first proposed, it has been used in many real-world applications and many new variants based on it have been put forward. To achieve the target detection task for

Unmanned Aerial Vehicles (UAVs) at low altitude, a hybrid model of Edge Potential Function (EPF) and the Simulated Annealing Pigeon Inspired Optimization (SAPIO) algorithm were proposed by Li and Duan (2014). The robustness and effectiveness of the SAPIO algorithm were shown by a number of comparative experiments with other algorithms. Moreover, a novel Predator-Prey Pigeon-Inspired Optimization (PPPIO) (Zhang and Duan 2017) was proposed to solve the Uninhabited Combat Aerial Vehicle (UCAV) three-dimension path planning problem in the dynamic environment. The comparative simulation results show that the proposed PPPIO algorithm is more efficient than other algorithms for solving the problem. In addition, an orthogonal PIO algorithm (Duan and Wang 2015) was suggested and employed in the training process of the Echo State Network (ESN) to obtain the desired parameters. The superiority of the orthogonal PIO algorithm is shown by comparing it with several existing bio-inspired optimization algorithms. Furthermore, an improved PIO algorithm (Deng and Duan 2016) was utilized by converting the parameter design problem for the automatic carrier landing system to an optimization problem. A series of experiments were conducted to demonstrate the feasibility and effectiveness of the proposed method. In addition, the Gaussian Pigeon Inspired Optimization (GPIO) algorithm (Zhang and Duan 2015) was proposed for solving the optimal formation reconfiguration problems of multiple orbital spacecraft. The feasibility and effectiveness of the proposed GPIO algorithm in solving orbital spacecraft formation reconfiguration problems were verified by the comparative experiments with the basic PIO and the Particle Swarm

Optimization (PSO).

The PIO is also extended to solve multi-objective problems. Qiu and Duan (2015) proposed the Multi-objective Pigeon Inspired Optimization (MPIO) to solve the multi-objective optimization problems in designing the parameters of brushless direct current motors. Moreover, the MPIO was modified based on the hierarchical learning behaviour in pigeon flocks and a UAV distributed flocking control algorithm based on the modified MPIO was proposed to coordinate UAVs to fly in a stable formation under complex environments (Qiu and Duan 2018).

2.4.5 Hybrid Meta-heuristic Algorithm

Hybrid metaheuristics, which combine the advantages of separate components, have attracted the attention of more and more scholars. Many investigations in using hybrid algorithms in production scheduling problems have been carried out in the last decade. Different hybrid algorithms were proposed, and some examples are summarized in Table 2-1.

Since these hybrid metaheuristic algorithms employ the key characteristics of individual algorithms, they are more efficient compared with a single algorithm. Thus, more and more researchers are using hybrid metaheuristic algorithms to address different scheduling problems.

Table 2-1 Examples of hybrid algorithms

Hybrid algorithm	Source
A hybrid of the Particle Swarm Optimization (PSO) with the Nawaz-Enscore-Ham (NEH) heuristic, as well as the Simulated Annealing (SA)	Liu, Wang, and Jin (2007)
A combination of the Tabu Search and the Variable Neighbourhood Search (VNS/TS)	Liao and Cheng (2007)
A mixture of the Differential Evolution-based algorithm (DE), the Variable Neighbourhood Search (VNS) method and the Genetic Algorithm (GA)	Zobolas, Tarantilis, and Ioannou (2009)
Collaboration in the Genetic Algorithm (GA) and the variable neighbourhood search (VNS)	Behnamian and Ghomi (2011)
A hybrid of the Particle Swarm Optimization (PSO) and the Simulated Annealing (SA) algorithm with the Variable Neighbourhood Search (VNS)	Huang, Tian, and Ji (2016)
A combination of the Discrete Particle Swarm Optimization (DPSO) and the Stochastic Variable Neighbourhood Search(SVNS)	Wang and Tang (2012)
A mixture of the Permutation-based Harmony Search (PHS) with the Enhanced Basic Variable Neighbourhood Search (EBVNS)	Liu, and Zhou (2013)
A cooperation of the enhanced Variable Neighbourhood Search (VNS) and the Artificial Neural Network (ANN)	Mokhtari (2014)
The hybrid Particle Swarm Optimization algorithm (PSO) based on the Variable Neighbourhood Search (VNS)	Gao et al. (2015)
A hybrid Genetic Algorithm with the Variable Neighbourhood Search (GAVNS)	Xia, Li, and Gao (2016)

2.5 Research Gap

After reviewing the literature related to the production scheduling with machine maintenance problem, the production scheduling with multi-resource maintenance problem, production scheduling with uncertainty and meta-heuristic algorithm, the following research gaps are identified:

(1) Deep investigation on production scheduling problem with multi-resource

maintenance. Given the extensive research studies on production scheduling problem with machine maintenance, we can know that the integration of production and maintenance can improve the production efficiency greatly. Furthermore, as one of the most crucial elements in the plastic industry, the mould maintenance problem also attracts the attention of researchers while the research on the integration of the production scheduling problem with machine and mould maintenance is limited. Most studies on these integrated problems usually use the joint scheduling strategy, they usually decide the production scheduling and maintenance problem simultaneously aiming at achieving a goal. However, we need to analysis the problem deeply and propose a new strategy from another aspect to get a better result of the optimization problem. Moreover, some efficient algorithms need to be developed to resolve the integrated problem. Most studies on the production scheduling problem with multi-resource maintenance use the Genetic Algorithm (GA), which is a very traditional meta-heuristic algorithm. However, the convergence rate of GA is slow since the mutation of GA is unguided. Moreover, the diversity mechanism determines the performance of GA. GA tends to converge into local optima prematurely if the diversity mechanism does not work properly. Except for GA, other algorithms such as Particle Swarm Optimization (PSO) and Variable Neighbourhood Search algorithm (VNS) are also popular for these integrated problems. The main advantage of PSO is that the convergence rate is rapid, but it is susceptible to premature convergence, especially when dimensions or decision variables are large. The performance of the

VNS depends on the initialization and definition of the neighbourhood. If these two factors are not determined properly, the computing cost will be large, and the quality of the solution will be influenced. Since these algorithms have limitations, more innovative algorithms need to be proposed to solve the integrated problem. A new encoding and decoding method can be proposed, and many mechanisms can be added to the new algorithm to improve the search ability.

(2) Considering multiple maintenances in the production scheduling problem with multi-resource maintenance. From the review of the literature, it can be seen that the effect of maintenance is still a hot research area, and the system with multi-states is getting more and more attention to research. however, the research on production scheduling with diverse maintenances problem based on different states is still limited. So far, few of the research considers multiple maintenances in the production scheduling problem with multi-resource maintenance. Furthermore, the application of PSO in the production scheduling problem with maintenance is scarce. Some more efficient strategies based on PSO need to be proposed for such an integrated problem.

(3) Modification of production scheduling problem with mould maintenance by adding fuzzy factors. Although there have been a few studies on the production scheduling problem considering mould maintenance, none of them considered the indeterminate processing time and maintenance time, and the uncertainty needs to be

considered in practical application. Furthermore, when the fuzziness is considered in the integrated problem, the robustness should also be considered because that some differences may exist between the fuzzy schedule and the actually executed schedule. Moreover, since the integrated production scheduling with the maintenance problem is NP-hard, meta-heuristics are usually used to solve these problems. So far, most of these integrated problems are solved by the Genetic Algorithm (GA) approach. As a new swarm intelligent algorithm, the Pigeon Inspired Optimization (PIO) algorithm has not yet been applied in the production scheduling problem, and more efficient variants of PIO need to be explored to solve real-world problems.

2.6 Summary

Production scheduling problem with multi-resource maintenance has received much attention in existing research studies. This chapter presents a review of the literature related to this integrated problem from different aspects and provides fundamental supports for developing new algorithms to solve the addressed production scheduling problem with multi-resource maintenance in this research and a direction to modify the original model and seek new algorithms to solve these new problems.

From the review of the literature, it is noted that the production scheduling problem with multi-resource maintenance is still a crucial research area, especially in the plastic industry. For this complex problem, some scholars have built models and proposed algorithms to solve them. However, there is still space to improve the result and many other efficient algorithms such as Particle Swarm Optimization (PSO) algorithm and Variable Neighbourhood Search (VNS) algorithm which are proven to has the ability to resolve production scheduling problem have potential to achieve a better result. Furthermore, from the literature review, a new hybrid method of algorithms which combines the advantages of both algorithms could be proposed to improve the search ability of the separate algorithm.

Furthermore, in the existing production scheduling problem with multi-resource maintenance, the maintenance is preventive maintenance and only perfect maintenance is considered, which is not practical. Actually, a resource suffers from degradation and different states could be used to represent the different levels of degradation. The state of resource can be estimated through condition monitoring. For different states, maintenance strategies are different. So far, there is no research about this problem. In this connection, a new problem that considers multi-maintenance needs to be considered.

Besides, the production scheduling problem with fuzzy processing time and maintenance is more practical and is still a hot topic. Therefore, the development of effective algorithms to tackle optimization problems is a significant step in the research study and the integrated model will be modified by considering the fuzzy factors to make it more practical.

Chapter 3. A Hybrid Algorithm for the Production Scheduling Problem with Mould Maintenance

This chapter is to introduce the hybrid algorithm of the PS-MM problem. In Section 3.1, the problem is explained, which restates the problem. Section 3.2 describes the primary work that has been done around the problem, followed by Section 3.3, the proposition of the hybrid algorithm, and Section 3.4 numerical experiments. This chapter ends with conclusions and a summary.

3.1 Problem Description

The Production Scheduling with Mould Maintenance (PS-MM) problem was firstly proposed by Wong, Chan, and Chung (2012). It is a single-operation scheduling problem with identical parallel machines. In the practical plastics injection production system, each job has only one operation and the workshop of the system contains not only one machine. Different machines' processing time is the same if the injection mould for the job could be installed properly on these machines. The details of the problem could be described as follows: P jobs are distributed on Q injection machines and R injection moulds. Each problem is denoted as $P \times Q \times R$. All jobs, machines, and moulds are available for processing at time zero. A job can only be allocated to a specific mould and a specific mould can be allocated to many different machines but not all the machines. A specific mould can perform different jobs. At a given time slot, each job could only be performed by one machine with one specific mould; each machine could only conduct one job with one mould; each mould could only carry out one job on one machine. The unit operation time needed for a job is decided by the specific mould being used but not the machine. The batch size of each job is given, and the total operation time of a job is the product of batch size and the unit operation time. Each job is operated according to its order quantity (batch), which cannot be split, and there is no once interruption during the operational process of a job. The cumulated operating time of a resource is defined as the resource age (not including idle time). In practical situations, maintenance time needed is less if the maintenance is conducted earlier, so the relationship between maintenance time and resource (machine or mould) age can be fitted by a piecewise linear function. Because the possibility of mould breakdown is higher than machine breakdown, the maximum age of the mould is shorter than the maximum age of the machine. Once a resource reaches its maximum age, maintenance must be conducted after completion of the current job. This model only considers perfect maintenance, which means that after the maintenance, the resource age is reset to zero and the condition of the resource is as good as new. We assume that preventive maintenance can prevent all the random breakdowns of the resources. Furthermore, we assume that the set-up times do not depend on the sequence and is contained in the processing period, and the quality issue is not considered.

The objective is to find a good production scheduling and machine maintenance planning, as well as mould maintenance planning aiming at minimizing the makespan.

3.2 Primary Work

In most of the literature, Production Scheduling with Mould Maintenance (PS-MM) problem is solved by a Genetic Algorithm (GA), which is suitable for this discrete problem. This section tries to address this problem with the Particle Swarm Optimization algorithm (PSO), which is usually applied to resolve continuous problems. To successfully apply Particle Swarm Optimization (PSO) algorithm into this problem, the most important process is the encoding and decoding of particles. A new encoding and decoding approach designed for the PS-MM problem is given. Furthermore, to show the benefits of Particle Swarm Optimization (PSO), a simple Genetic Algorithm (GA) is designed. This section lays the foundation for the proposition of the new algorithm.

3.2.1 Particle Swarm Optimization Algorithm

The flowchart of PSO and its application in the production scheduling problem with multi-resource maintenance can be seen from Figure 3-1. The details can be described as follows:

Step 1. Initialization. Initialize a population of particles with random positions and velocities on $4\times P$ (P is the number of jobs) dimensions in the searching space.

Step 2. Fitness. Measure the fitness of each particle in the population and find the local best solution for all particles and the global best solution.

Step 3. Update the velocity and position of each particle. The velocity and position are updated according to Equation 3-1, Equation 3-2, and Equation 3-3, where, $X_{ij}(r)$,

$X_{ij}(r+1)$ and $V_{ij}(r)$, $V_{ij}(r+1)$ are the positions and velocities of the j^{th} dimension of the particle i at the r^{th} and $(r+1)^{th}$ iteration. $P_{ij}(r)$ is the best position of the j^{th} dimension of the particle i at the r^{th} iteration. $P_{gj}(r)$ is the best position of the j^{th} dimension of all the particles at the r^{th} iteration, and W is the inertia factor. $W_{\max}$ is the maximum inertia factor and $W_{\min}$ is the minimum inertia factor. C_1 is the particle acceleration coefficient and C_2 is the population acceleration coefficient.

Step 4. Fitness. Update the optimal value of each particle. P_{ibest} is the best value for particle i during the iteration process, and P_i^r is the current fitness of particle i. If $P_i^r < P_{ibest}$ then set $P_{ibest} = P_i^r$; otherwise, P_{ibest} retains. Update the optimal value of the population. Define g_{best} as the best value of the particle population, and $g_{best} = min(P_{ibest}), (i = 1,2,\cdots P_{e_{num}})$. If $g_{best}^r < g_{best}$, then set $g_{best} = g_{best}^r$, otherwise, g_{best} retains.

Step 5. Termination. Set $r = r+1$, and check the condition that if $r \le r_{\max}$, then go to Step 3, start a new iteration.; otherwise, terminate the algorithm and calculate minimum makespan.

$$V_{ij}(r+1) = W \times V_{ij}(r) + C_1 \times R_1 \times (P_{ij}(r) - X_{ij}(r)) + C_2 \times R_2 \times (P_{gj}(r) - X_{ij}(r)) \quad (3\text{-}1)$$

$$X_{ij}(r+1) = X_{ij}(r) + V_{ij}(r+1) \quad (3\text{-}2)$$

$$W = W_{\max} - \frac{W_{\max} - W_{\min}}{r_{\max}} \times r \quad (3\text{-}3)$$

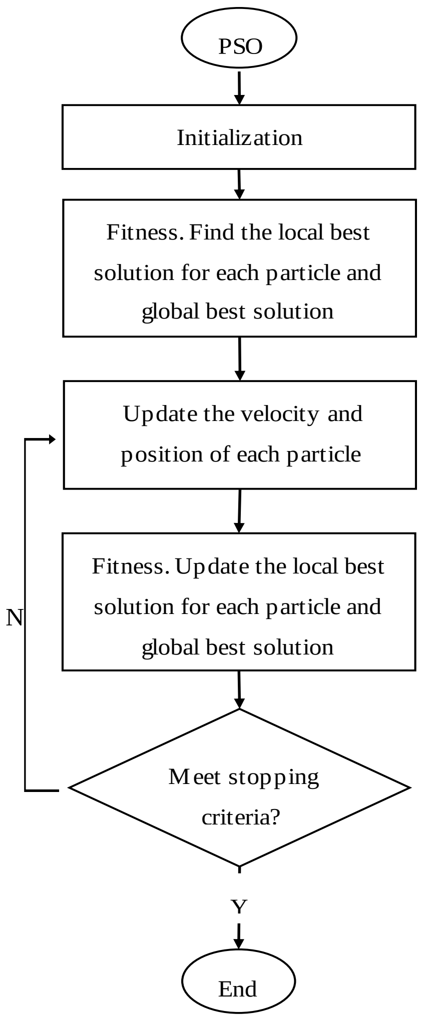
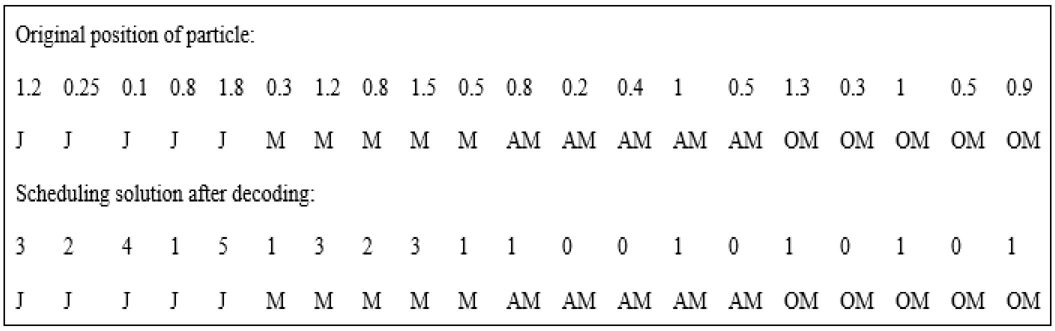

PSO
Initialization
Fitness. Find the local best solution for each particle and global best solution
Update the velocity and position of each particle
Fitness. Update the local best solution for each particle and global best solution
N
Meet stopping criteria?
Y
End

Figure 3-1 Flowchart of PSO

This algorithm uses the joint scheduling strategy, which means that the production scheduling, maintenance planning of machine and maintenance planning of mould are decided at the same time. Since each particle represents a candidate position (i.e. solution), every particle includes the information of job allocation and maintenance plan. For the problem containing P jobs, the dimension of the particle is $4 \times P$ which is divided into four groups, respectively representing the job sequence (J), the corresponding machine sequence (M), the machine maintenance (AM), the mould maintenance (OM). In the evolution process of PSO, the values of these particles' positions vary in the real number space. To decode the particles' position into a suitable scheduling solution for this problem, random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007) are used.

After decoding, the values of the J parameters are integers between 1 and P (P is the number of jobs); values of M parameters are integers between 1 and Q (Q is the number of machines); The AM parameter is the maintenance decision on the machine, with value 0 or 1; The OM parameter is the maintenance decision on the mould, with value 0 or 1; The corresponding resource is maintained after finishing the job if the relevant AM or OM is denoted as 1, otherwise they are denoted as 0.

Figure 3-2 shows examples of a particle before and after decoding. There are 5 jobs and 3 machines in this example. From Figure 3-2, we can see that the value of the job sequence (J) in the original position of the particle is (1.2 0.25 0.1 0.8 1.8), and it is transferred into (3 2 4 1 5) by random key representation (Bean 1994), and the smallest position value (SPV) rule (Tasgetiren et al. 2007). The value of the corresponding machine sequence (M) in the original position of the particle is (0.3 1.2 0.8 1.5 0.5) and we divide the interval [0.3 1.5] (0.3 is the minimum and 1.5 is the maximum among all the numbers) into 3 parts (there are 3 machines in this example). Since 0.3 is in the first part, after decoding, the value in the relevant position is 1, so the corresponding machine sequence (M) can be transferred into (1 3 2 3 1). A similar transfer method can be applied to machine maintenance (AM) and mould maintenance (OM). After decoding, we know that job 3 is distributed on machine 1 and machine 1 will be maintained after job 3 is finished, and the injection mould on machine 1 will also be maintained. Job 2 is allocated to machine 3, but machine 3 will not be maintained since the corresponding AM parameter is 0 and the injection mould on machine 3 will not be maintained either, because the corresponding OM parameter is 0.



Original position of particle:

1.2	0.25	0.1	0.8	1.8	0.3	1.2	0.8	1.5	0.5	0.8	0.2	0.4	1	0.5	1.3	0.3	1	0.5	0.9
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM

Scheduling solution after decoding:

3	2	4	1	5	1	3	2	3	1	1	0	0	1	0	1	0	1	0	1
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM

Figure 3-2 An example particle before and after decoding

3.2.2 Genetic Algorithm

The flowchart of GA and its application in the production scheduling problem with multi-resource maintenance can be seen from Figure 3-3. The details can be described as follows:

Step 1. Initialization. Initialize a pool of chromosomes with $4 \times P$ (P is the number of jobs) genes randomly in the chromosomes pool, each chromosome represents a candidate solution. The encoding of the chromosome is the same as the particle after decoding in PSO.

Step 2. Fitness. Calculate the makespan and fitness of each chromosome in the pool.

Step 3. Selection. Select chromosomes by Roulette Wheel Approach.

Step 4. Evolution. Perform Crossover or Mutation operation.

Step 5. Fitness. Calculate the makespan and fitness of each chromosome in the pool.

Step 6. Elitist Strategy. Record the best chromosome.

Step 7. Termination. Set $r = r + 1$, and check the condition that if $r < r_{\max}$, then go to Step 3; otherwise, terminate the algorithm and calculate minimum makespan.

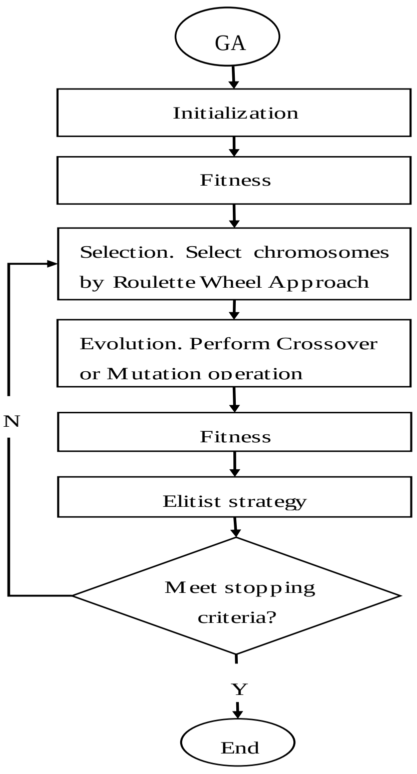

GA
Initialization
Fitness
Selection. Select chromosomes by Roulette Wheel Approach
Evolution. Perform Crossover or Mutation operation
Fitness
Elitist strategy
Meet stopping criteria?
N
Y
End

Figure 3-3 Flowchart of GA

Crossover Operator. To create new offspring chromosomes, some segments of two parent chromosomes are exchanged hoping to get better chromosomes through replacing the weaker segments with the stronger segments from others. In our algorithm, every two adjacent chromosomes are regarded as one pair of parent chromosomes that undergo a crossover operator. The crossover rate is 0.4. Figure 3-4 shows an example of the crossover operator. If there are 5 jobs in the problem which means 4×5 genes in the chromosomes and 2 genes in each group will undergo crossover operator. The first gene in every group is set as the initial position to begin the crossover operator. The gene in the job group is the conclusive gene. Once any gene in the job group (J) undergo the crossover operator, the corresponding gene in the group (M), (AM), (OM) undergo the same operator. The crossover operation should guarantee the validity of the chromosome corresponding gene in the group (M), (AM), (OM) undergo the same operator. The crossover operation should guarantee the validity of the chromosome.

Parent 1: **3 2** 5 4 1 1 3 2 3 1 1 0 0 1 0 1 0 1 0 1

Parent 2: **1 4** 5 3 2 2 3 1 2 3 0 1 1 0 0 0 1 0 1 1

Step 1:

Offspring 1: **1 4** 0 0 0 **2 3** 0 0 0 **0 1** 0 0 0 **0 1** 0 0 0

Offspring 2: **3 2** 0 0 0 **1 3** 0 0 0 **1 0** 0 0 0 **1 0** 0 0 0

Step 2:

Offspring 1: 1 4 3 2 5 2 3 1 3 2 0 1 1 0 0 0 1 1 0 1

Offspring 2: 3 2 1 4 5 1 3 2 3 1 1 0 1 1 0 1 0 1 0 1

Figure 3-4 An example of the crossover operator

Mutation Operator. The purpose of mutation is to increase the diversity of chromosome. To avoid repetition with the crossover operator, the only machine used by the job is changed. The gene in the M group is chosen stochastically. And the number of genes that undergo the mutation operator is decided by the mutation rate.

In this research, we set the mutation rate as 0.5, which means that 2 genes in the M gene group are changed. An example of a mutation can be seen in Figure 3-5.

Parent: 3 2 5 4 1 1 **3** 2 3 **1** 1 0 0 1 0 1 0 1 0 1

After mutation:

Offspring: 3 2 5 4 1 1 **2** 2 3 **3** 1 0 0 1 0 1 0 1 0 1

Figure 3-5 An example of the mutation operator

3.2.3 Numerical Experiments

In the numerical experiments, we assume that the relationship between maintenance time and resource age can be fitted by a piecewise linear function. The hypothetical maintenance scheme can be seen from Table 3-1, Figure 3-6 and Figure 3-7. Specifically, age is defined as the cumulated operating time of a resource. If a machine or mould age reaches the maximum age, maintenance has to be conducted for the particular machine after the completion of the current job. After each maintenance task, machine or mould age will be reset to 0, which means that the condition of the machine or mould is assumed to be as good as new after maintenance (perfect maintenance).

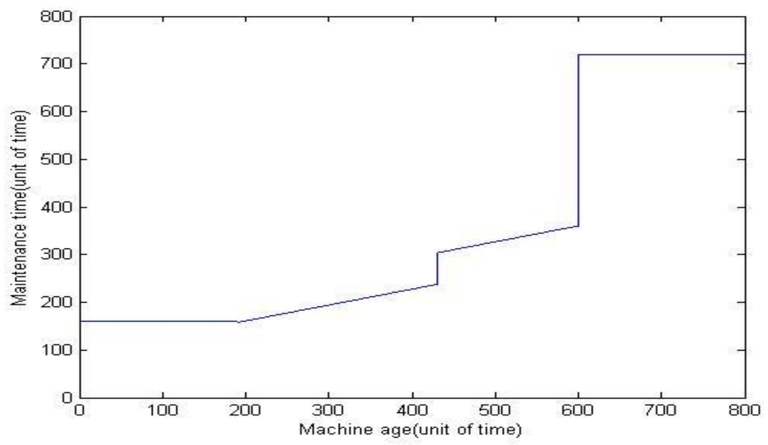


Figure 3-6 Maintenance time based on machine age

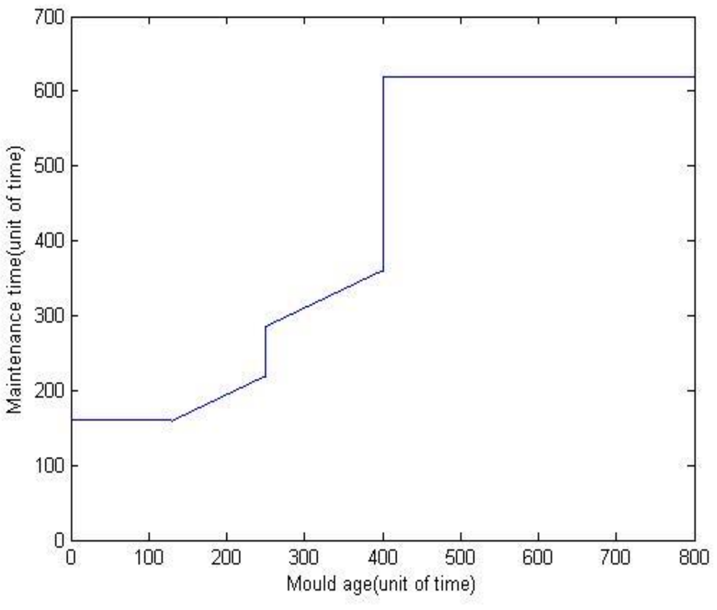


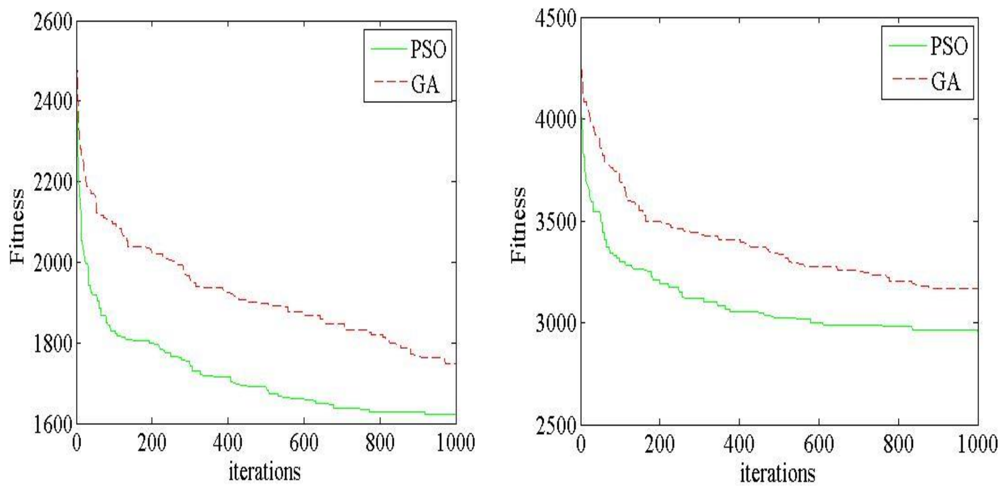
Figure 3-7 Maintenance time based on mould age

Table 3-1 Maintenance time based on machine/mould age

Machine age	Maintenance time	Mould age	Maintenance time
$0<A_1<=190$	160	$0<A_2<=130$	160
$190<A_1<=430$	$160+A_1/3+4-70$	$130<A_2<=250$	$160+A_2/2+4-70$
$430<A_1<=600$	$310+A_1/3-150$	$250<A_2<=400$	$310+A_2/2-150$
$600<A_1$	720	$400<A_2$	620

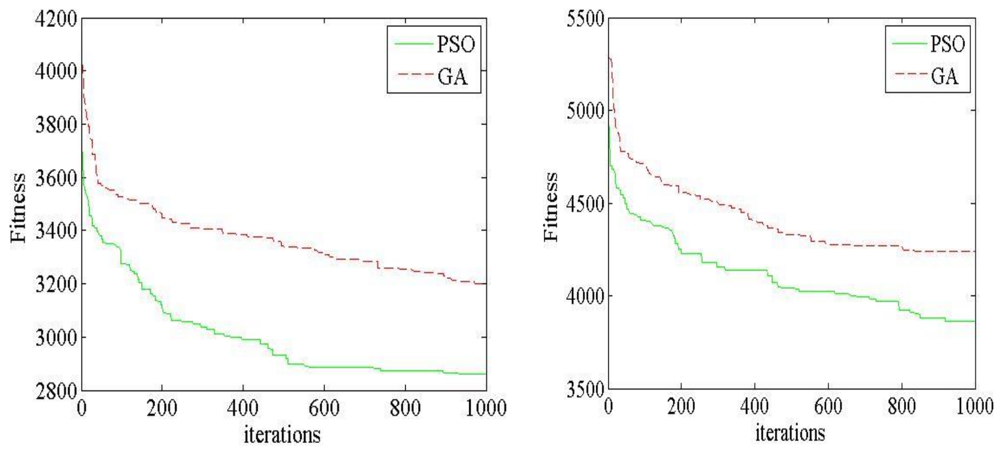
Numerical experiments are implemented in the Matlab environment on a personal computer with Intel(R) Core(TM) i7-6700 CPU 3.40GHz CPU. Based on the recommendation of Eberhart, and Shi (2000), the parameters of PSO are set as follows: $C_1 = C_2 = 2$, $W_{\max} = 0.9$, $W_{\min} = 0.4$. When W is initially set as 0.9, greater initial exploration is conducted and then the stepwise movement of each particle is reduced. When W is reduced linearly to 0.4, it can speed convergence to the global optimum. C_1 and C_2 determine the balance between the impact of the individual's experience and the group’s knowledge. This setting means that an individual’s knowledge and experience of the group have the same influence on the search direction. These settings are proven to be able to provide satisfactory results. For the parameters of GA, we set the mutation rate as 0.5 and the crossover rate as 0.4. To show the effectiveness of PSO, four data sets are tested. The parameters of the instances in Table 3-2 are set as follows: the operation time of each job is produced randomly between 30 and 55 units of time; the batch size of each job is produced randomly between 2 and 6 units. The initial size of the population is 50 and the number of iteration is 1000. Each algorithm will be run 10 times in order to measure the deviation of the solutions obtained. The results obtained are shown in Table 3-3. The convergence curves for each instance can be seen from Figure 3-8 and Figure 3-9.

From the results, we can see that for those four different cases, the convergence time of PSO and GA are similar, but the Average makespan, Min makespan and Max makespan obtained by PSO are all smaller than the corresponding results obtained by GA. Furthermore, PSO always converges faster than GA. From all the results, we can conclude that PSO is more suitable to solve these four cases in terms of the convergence rate and solution quality.



(a) Convergence curve of data set 1 (b) Convergence curve of data set 2

Figure 3-8 Convergence curve of data set 1 and 2



(a) Convergence curve of data Set 3 (b) Convergence curve of data set 4

Figure 3-9 Convergence curve of data set 3 and 4

Table 3-2 Parameters of instances

Data Set Number	Job Number	Machine Number	Mould Number
1	20	3	4
2	35	5	10
3	50	8	14
4	65	9	15

Table 3-3 Results comparison of PSO and GA

Data Set.	Algorithm	Average	Min	Max	St. Dev	CPU time
1	PSO	1622	1502	1714	72	120.75sec
1	GA	1749	1645	1884	73	115.70sec
2	PSO	2953	2736	3137	115	160.20sec
2	GA	3165	2865	3266	129	150.81sec
3	PSO	2862	2644	3091	138	210.54sec
3	GA	3201	3000	3354	111	196.62sec
4	PSO	3861	3758	4097	104	289.94sec
4	GA	4238	3962	4411	120	268.14sec

3.2.4 Conclusions

In this subsection, the production scheduling with multi-resource maintenance problem is studied, which integrates mould maintenance into the traditional job shop scheduling problem. Maintenance duration and interval are subject to the age of resources. To minimize the overall makespan, a jointly scheduling strategy is used. Particle Swarm Optimization Algorithm (PSO) is designed to solve this optimization problem and the Genetic Algorithm (GA) is the comparison algorithm. From simulation results of four instances with different sizes, we can know that PSO is more efficient to deal with the integrated problem than the proposed GA in terms of convergence rate and solution quality. More efficient algorithms based on PSO will be explored to solve the problem.

3.3 A Hybrid Algorithm for the PS-MM Problem

The optimization algorithm, named the TLPSO-VNS algorithm, is introduced in this subsection. The overall algorithm structure is introduced firstly and includes two stages: the stage of swarm initialization and swarm improvement by TLPSO and the stage of swarm intensification via VNS. Since encoding and decoding of the particles are critical for the successful application of TLPSO, encoding and decoding of the particles are introduced before the details of the algorithm.

3.3.1 Overall Algorithm Description

For this integrated scheduling problem, we divide it into three subproblems: the production scheduling problem, the machine maintenance problem and the mould maintenance problem. There are many potential production scheduling solutions and for every specific production scheduling solution, there are many potential machine maintenance solutions and mould maintenance solutions. The First level PSO contributes to finding some good production scheduling solutions. Then the Second level PSO can help every specific production scheduling solution to find a good machine maintenance solution. Once the production scheduling and machine maintenance solutions are confirmed, the Third level PSO helps to find a good mould maintenance solution for every specific production scheduling solution with specific machine maintenance. Compared with the joint scheduling strategy, this decomposition mechanism improves the precision of the search. Once some good scheduling solutions are obtained from TLPSO, VNS is used to enhance these solutions. For these solutions, seven kinds of neighbours are designed to conduct the local search. The best solution is reserved when the VNS ends. Figure 3-10. The framework of the overall algorithm shows the process of the problem analysis and the design of the overall algorithm.

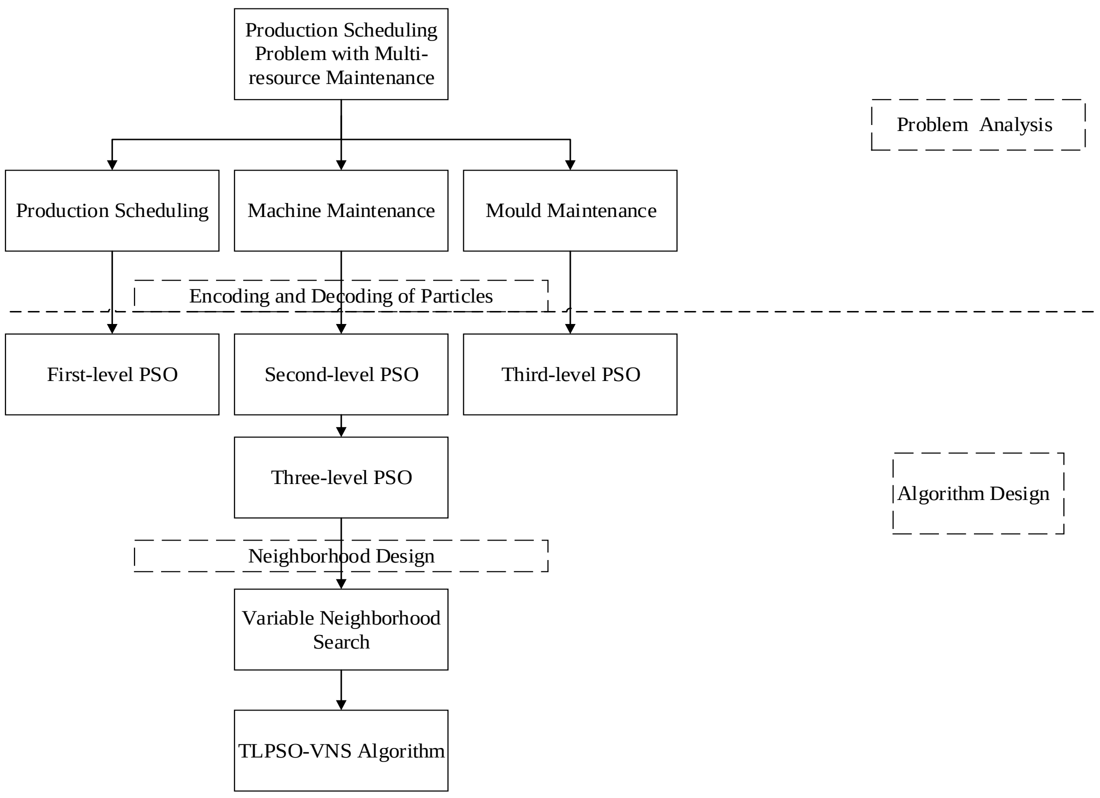


Figure 3-10 The framework of the overall algorithm

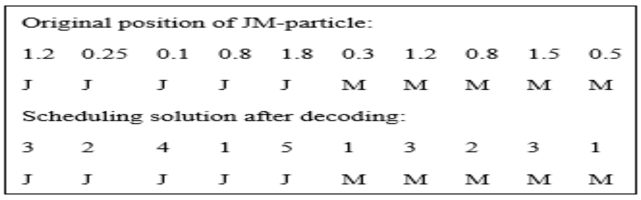
3.3.2 Encoding and Decoding of the Particles in the Three-Level PSO (TLPSO)

In the Three-Level PSO (TLPSO), the first level PSO only considers the problem of production scheduling. So, the particle in the first level PSO only includes information on the job sequence (J) and the corresponding machine sequence (M). The particle in the first level PSO is named as the JM-Particle with dimension $2 \times P$ (P is the number of jobs). The second level PSO focuses on the problem of machine maintenance (AM), so the particle in the second level PSO also contains information on machine maintenance (AM) apart from information on the job sequence (J) and corresponding machine sequence (M). The particle in the second level PSO is named as the JMAM-Particle, with dimension $3 \times P$. The third level PSO focuses on the problem of mould maintenance (OM), so the particle in the third level PSO also includes information on the mould maintenance (OM) besides information on the job sequence (J), the corresponding machine sequence (M) and machine maintenance (AM). The particle in the third level PSO is named as the JMAMOM-Particle with dimension $4 \times P$. In the evolution process of TLPSO, the values of these particles' positions vary in the real number space. To decode the particles' position into a suitable scheduling solution for this problem, random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007) are used. After decoding, the values of the J parameters are integers between 1 and P (P is the number of jobs); values of M parameters are integers between 1 and Q (Q is the number of machines); The AM parameter is the maintenance decision on the machine, with value 0 or 1; The OM parameter is the maintenance decision on the mould, with value 0 or 1; The corresponding resource is maintained after finishing the job if the relevant AM or OM is denoted as 1, otherwise they are denoted as 0.

Figure 3-11. Encoding and decoding of a series of particles in TLPSO shows a series of JM-Particle, JMAM-Particle, JMAMOM-Particle examples before and after decoding. There are 5 jobs, 3 machines and 2 moulds in this example. Mould 1 can be used to produce Jobs 1, 3, 5 and Mould 2 can be used to produce Jobs 2, 4. From Figure 2c, we can see that the value of the job sequence (J) in the original position of the JMAM-particle is (1.2 0.25 0.1 0.8 1.8), and it is transferred into (3 2 4 1 5) by

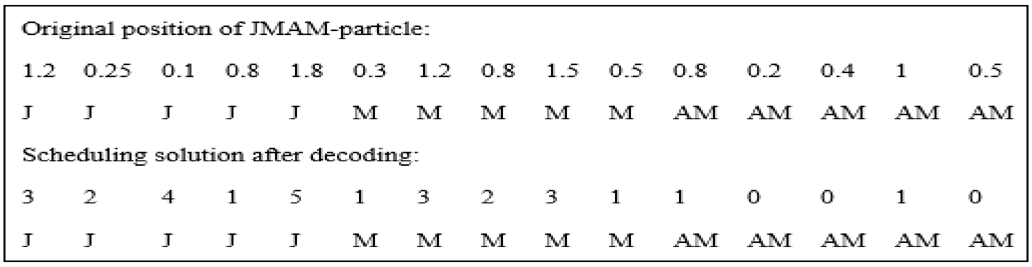
random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007). We rank the sequence (1.2 0.25 0.1 0.8 1.8) according to the ascending order and obtain (0.1 0.25 0.8 1.2 1.8). Since the number 0.1 is in the third position of the original sequence, we decode it into 3. Since the number 0.25 is in the second position of the original sequence, we decode it into 2. By this rule, the original positions can be decoded into a suitable scheduling solution (3 2 4 1 5). The value of the corresponding machine sequence (M) in the original position of the JMAM-particle is (0.3 1.2 0.8 1.5 0.5). We divide the interval [0.3 1.5] (0.3 is the minimum and 1.5 is the maximum among all the numbers) into 3 intervals, [0.3 0.7), [0.7 1.1) and [1.1 1.5] (there are 3 machines in this example). Since 0.3 and 0.5 are in the first interval, after decoding, the value in the relevant position is 1. Since 0.8 is in the second interval after decoding, the value in the relevant position is 2. Since 1.2 and 1.5 are in the third interval after decoding, the value in the relevant position is 3. So, the corresponding machine sequence (M) can be transferred into (1 3 2 3 1). The value of the corresponding machine maintenance sequence (AM) in the original position of the JMAM-particle is (0.8 0.2 0.4 1 0.5) and we divide the interval [0.2 1] (0.2 is the minimum and 1 is the maximum among all the numbers) into 2 intervals, [0.2 0.6) and [0.6 1]. Since 0.8 and 1 are in the interval [0.6 1], after decoding, the value in the relevant position is 1. Since 0.2, 0.4 and 0.5 are in the interval [0.2 0.6), after decoding, the value in the relevant position is 0. So, the corresponding machine maintenance sequence (AM) can be transferred into (1 0 0 1 0). A similar transfer method can be applied to mould maintenance (OM). After decoding, it can be known that job 3 is distributed on machine 1 and machine 1 will be maintained after job 3 is finished, and the injection mould on machine 1 will also be maintained. Job 2 is allocated to machine 3, but machine 3 will not be maintained since the corresponding AM parameter is 0 and the injection mould on machine 3 will not be maintained because the corresponding OM parameter is 0.

To illustrate the influence of the preventive maintenance on the objective function's value, the Gantt charts of the example in Figure 3-11a (without resource maintenance consideration) and the Gantt charts of the example in Figure 3-11c (with resource maintenance consideration) are demonstrated in Figure 3-12 and Figure 3-13. In the Gantt charts, MT means maintenance. The numbers in the brackets are the processing time needed by each job and the maintenance time needed by the corresponding resource. Since the beginning time of a job is decided by the available time of the machine and the mould that the job uses, we can see that the job 1 is delayed because of the maintenance on mould 1, and the job 5 is delayed because of the maintenance on machine 1. However, the maintenance on machine 3 after job 1, the maintenance on mould 1 after job 5 and the maintenance on mould 2 after job 4 have no influence on the objective because all the jobs are finished. Also, we know that the makespan is changed from 125 units of time to 180 units of time because of machine maintenance and mould maintenance.



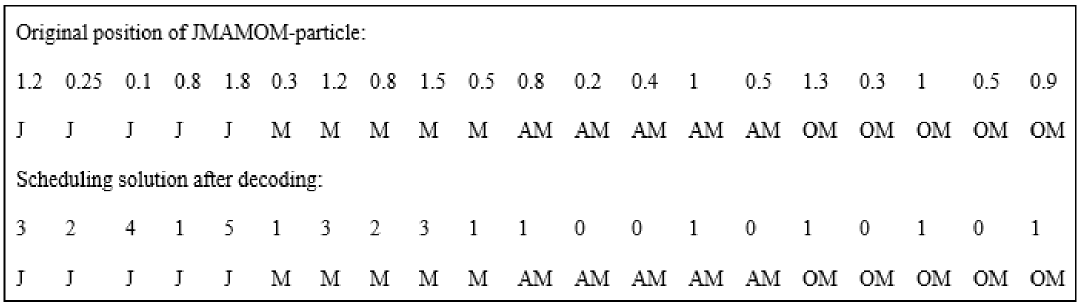
Original position of JM-particle:									
1.2	0.25	0.1	0.8	1.8	0.3	1.2	0.8	1.5	0.5
J	J	J	J	J	M	M	M	M	M
Scheduling solution after decoding:									
3	2	4	1	5	1	3	2	3	1
J	J	J	J	J	M	M	M	M	M

Figure 3-11a. Encoding and decoding of JM- particle



Original position of JMAM-particle:														
1.2	0.25	0.1	0.8	1.8	0.3	1.2	0.8	1.5	0.5	0.8	0.2	0.4	1	0.5
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM
Scheduling solution after decoding:														
3	2	4	1	5	1	3	2	3	1	1	0	0	1	0
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM

Figure 3-11b Encoding and decoding of JMAM-particle



Original position of JMAMOM-particle:																			
1.2	0.25	0.1	0.8	1.8	0.3	1.2	0.8	1.5	0.5	0.8	0.2	0.4	1	0.5	1.3	0.3	1	0.5	0.9
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM
Scheduling solution after decoding:																			
3	2	4	1	5	1	3	2	3	1	1	0	0	1	0	1	0	1	0	1
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM

Figure 3-11c Encoding and decoding of JMAMOM-particle

Figure 3-11 Encoding and decoding of a series of particles in TLPSO

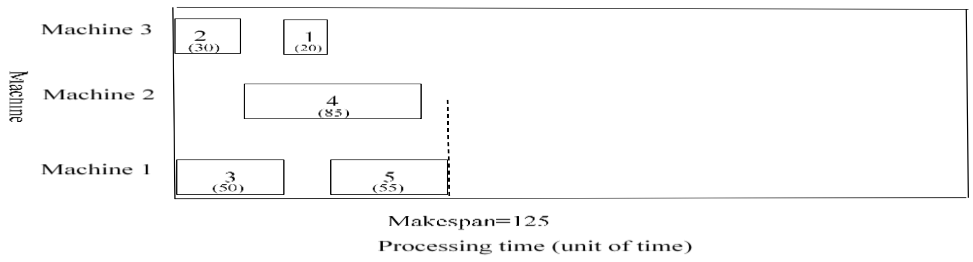


Figure 3-12a Machine schedule of the example in Figure 3-11a

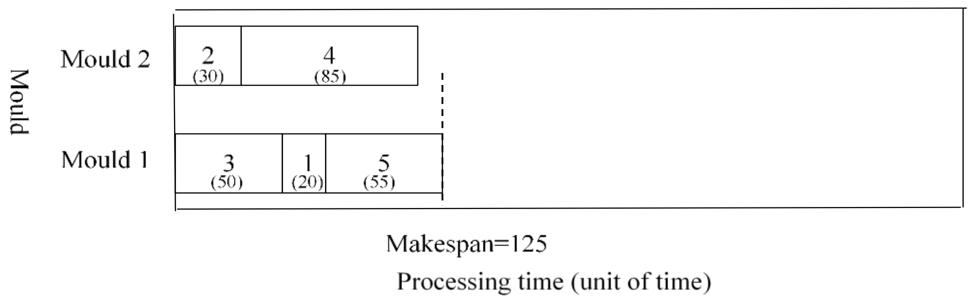


Figure 3-12b Mould schedule of the example in Figure 3-11a
Figure 3-12 Gantt charts of the example in Figure 3-11a

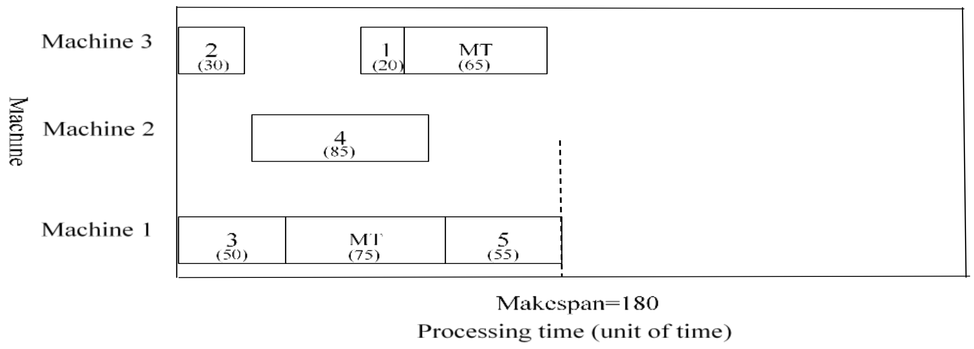


Figure 3-13a Machine schedule of the example in Figure 3-11c

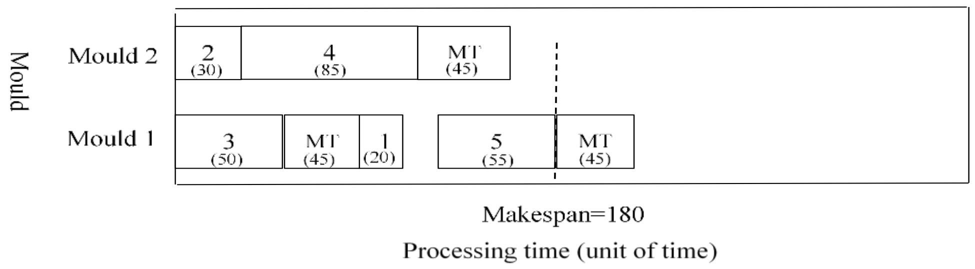


Figure 3-13b Mould schedule of the example in Figure 3-11c
Figure 3-13 Gantt charts of the example in Figure 3-11c

In our assumption, the maintenance must be conducted after completion of the current job once a resource reaches its maximum age. There is the possibility that the scheduling solution does not show the corresponding compulsory maintenance on the resource. Under this situation, the compulsory maintenance time (the maximum maintenance time need by a resource) should be added when the objective function (makespan) is calculated. To explain this situation better, an example is given, and the according Gantt charts are shown in Figure 3-14. Gantt charts without compulsory maintenance and Figure 3-15. Gantt charts with compulsory maintenance. There are 8 jobs, 3 machines and 2 moulds (job 1, 3, 5 are allocated to mould 1 and job 2, 4, 6, 7, 8 are allocated to the mould 2) in this example. The maximum age of the machine is 100 units of time and the maximum age of the mould is 90 units of time. Once the maximum age for a resource is reached, compulsory maintenance must be conducted when the current job is completed. A scheduling solution after decoding is (5 7 2 1 8 3 6 4 3 2 1 2 3 3 1 2 1 0 0 0 0 1 1 0 0 0 0 1 0 1 0 1) and the according Gantt chart is demonstrated in Figure 3-14. From Figure 3-14, we can know that the age of mould 2 is over 90 units of time after the job 8. So, compulsory maintenance (50 units of time) must be conducted after the job 8, and the Gantt charts in Figure 3-14 need to be modified as the Gantt charts in Figure 3-15. The compulsory preventive maintenance is added after the job 8 and the overall makespan is improved from 175 units of time to 225 units of time because of the compulsory preventive maintenance.

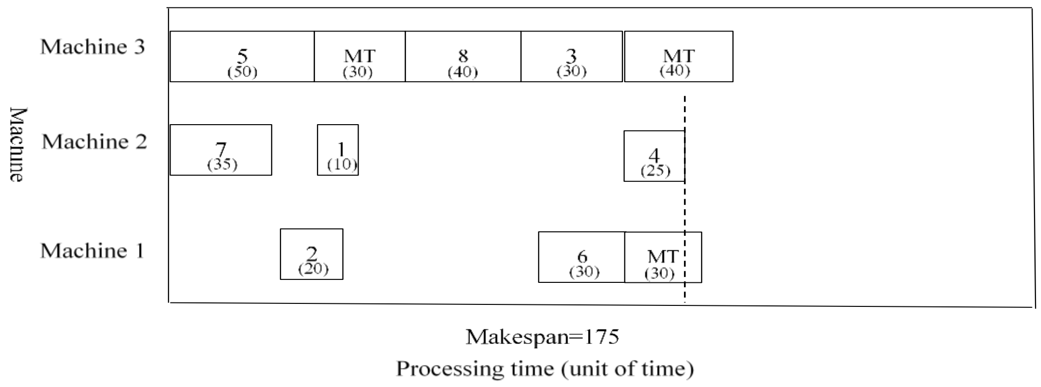


Figure 3-14a Machine schedule without compulsory maintenance

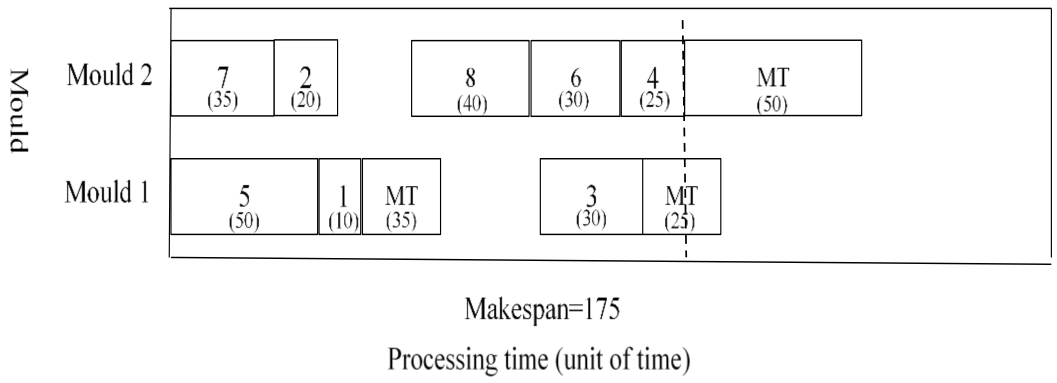


Figure 3-14b Mould schedule without compulsory maintenance

Figure 3-14 Gantt charts without compulsory maintenance

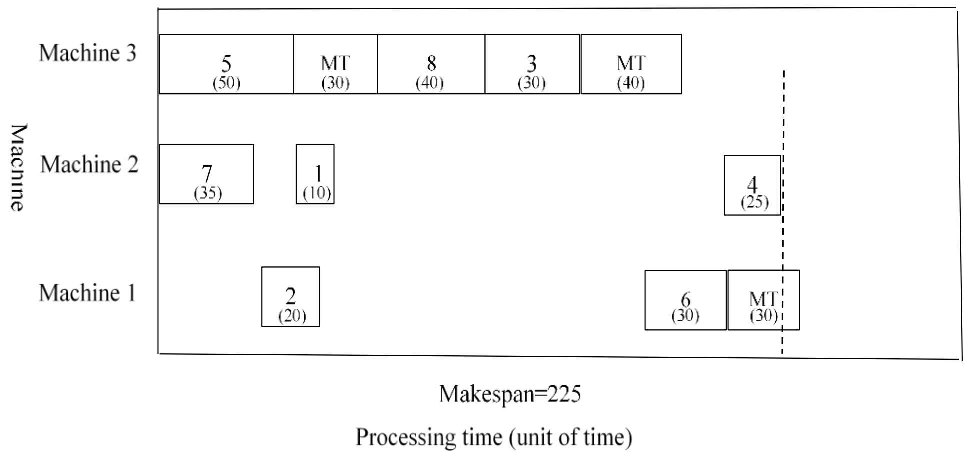


Figure 3-15a Machine schedule with compulsory maintenance

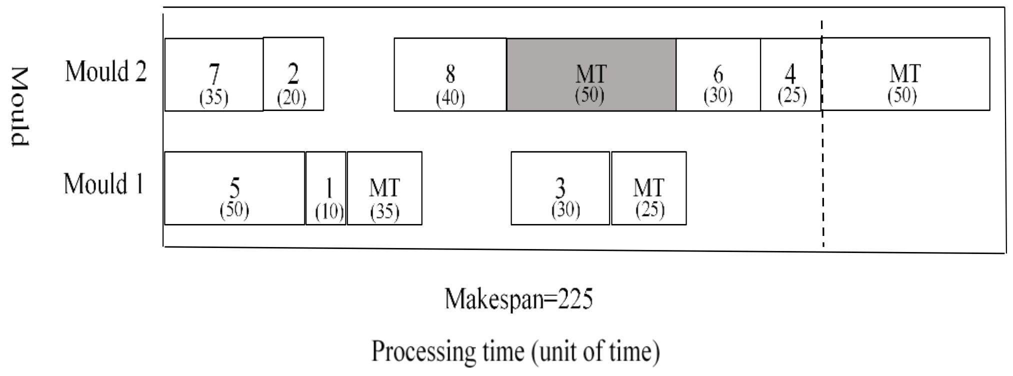


Figure 3-15b Mould schedule with compulsory maintenance

Figure 3-15 Gantt charts with compulsory maintenance

3.3.3 Swarm Initialization and Swarm Improvement through Three-Level PSO

The TLPSO algorithm consists of three levels, referred to as the first level PSO, the second level PSO and the third level PSO. The first level PSO solves the production scheduling problem, which is the master problem. The JM-particle in the first level PSO stores the information on the job sequence and the corresponding machine sequence. Firstly, the JM-particle, acting as input data, will be passed to the second level PSO individually. In the second level PSO, each JM-particle will get a series of JMAM-particles which have the same information on job sequence and corresponding machine sequence but different information on machine maintenance. Secondly, each JMAM-particle will be passed into the third level PSO to obtain a series of JMAMOM-particles which contain the same information on job sequence, corresponding machine sequence, and machine maintenance, but different information on mould maintenance. Thirdly, the best JMAMOM-particle for each JMAM-particle obtained from the third level PSO will be sent back to the second level PSO. After the evolution process of the second level PSO, the best JMAM-particle corresponding to the best JMAMOM-particle obtained from the third level PSO will be delivered back to the first level PSO. The best JMAM-particle from the second level PSO with its corresponding best JMAMOM-particle from the third level PSO for each JM-particle will be recorded at each iteration of the first level PSO. When the iteration process of the first level PSO finishes, all the JM-particles positions, and the scheduling solution of corresponding best JMAMOM-particles are recorded. The TLPSO algorithm ends. The details can be described as follows:

Start: the batch size of each job, the corresponding mould of each job, the available machines for each job and the unit operation time of each job are the input data for TLPSO.

1st level PSO

Step 1: Initialization. Initialize a population of JM-Particles with random positions and velocities in [0, 1] and the dimension is $2 \times P$ (P is the number of jobs).

Step 2: Pass each JM-Particle one by one to the second level PSO. In the second level PSO, **Step 2a- Step 2g** will be conducted until the stopping condition is reached, and the best JMAM- Particle with its corresponding best JMAMOM-Particle will be recorded.

2nd level PSO

Step2a: Initialization. For every JM-Particle, initialize a population of JMAM-Particles with $3 \times P$ dimensions. The first $2 \times P$ dimensions of JMAM-Particles are all the same as its related JM-particle. The last P dimensions of the JMAM-Particles are produced randomly in [0, 1]. A population of velocities is produced randomly, and the first $2 \times P$ dimensions of these velocities are zero, and the last P dimensions of the velocities are produced randomly in [0, 1].

Step2b: Pass every JMAM-Particle one by one to the third level PSO. In the third level PSO, **Step2b(i)-Step2b(v)** will be conducted until the stopping condition is reached, and the best JMAMOM-Particle will be recorded.

<u>3rd level PSO</u>

Step2b(i): Initialization. For each JMAM-Particle, initialize a population of JMAMOM-Particles with $4 \times P$ dimensions. The first $3 \times P$ dimensions of the JMAMOM-Particles are all the same as its related JMAM-Particle. The last P dimensions of the JMAM-Particles are produced randomly in [0, 1]. A population of velocities is produced, and the first $3 \times P$ dimensions of these velocities are zero, and the last P dimensions of velocities are produced randomly in [0, 1].

Step2b(ii): Fitness. Use random key representation and the smallest position value (SPV) rule to transfer the continuous position vector of the JMAMOM-Particle into a suitable scheduling solution, and then measure the fitness (makespan) of each JMAMOM-Particle in the population and find the local best solution for all JMAMOM-Particles and the global best solution.

Step2b(iii): Update the velocity and position of each JMAMOM-particle. The velocity is updated according to Equation 3-4 and then the position is updated according to Equation 3-5. Finally, the inertia factor W is updated according to Equation 3-6, where, $X_{ij}(r_3)$, $X_{ij}(r_3+1)$ and $V_{ij}(r_3)$, $V_{ij}(r_3+1)$ are the positions and velocities of the j^{th} dimension of the particle i at the r_3^{th} and $(r_3+1)^{th}$ iteration. Without loss of generality, we restrict the velocity in [-1, 1]. $P_{ij}(r_3)$ is the best position of the j^{th} dimension of the particle i at r_3^{th} iteration. $P_{gj}(r_3)$ is the best position of the j^{th} dimension of all the particles at the r_3^{th} iteration, and W is the inertia weight. $W_{\max}$ is the maximum inertia weight and $W_{\min}$ is the minimum inertia weight. C_1 is the particle acceleration coefficient and C_2 is the population acceleration coefficient. R_1 and R_2 are random numbers between 0 and 1.

$$\begin{aligned} V_{ij}(r_3+1) = W \times V_{ij}(r_3) + C_1 \times R_1 \times (P_{ij}(r_3) - X_{ij}(r_3)) \\ + C_2 \times R_2 \times (P_{gj}(r_3) - X_{ij}(r_3)) \end{aligned} \tag{3-4}$$

$$X_{ij}(r_3+1) = X_{ij}(r_3) + V_{ij}(r_3+1) \tag{3-5}$$

$$W = W_{\max} - \frac{W_{\max} - W_{\min}}{r_{3\max}} \times r_3 \tag{3-6}$$

Step2b(iv): Fitness. Again, use random key representation and the smallest position value (SPV) rule to transfer the continuous position vector of the particles into a suitable scheduling solution and measure the fitness (makespan) of each

JMAMOM-Particle and update the optimal value of each JMAMOM-particle. $P1_{ibest}$ is the best value for the JMAMOM-particle i during the iteration process, and $P1_i^{r_3}$ is the current fitness of particle i. If $P1_i^{r_3} < P1_{ibest}$, then set $P1_{ibest} = P1_i^{r_3}$, update $P_{ij}(r_3)$; otherwise, retain $P1_{ibest}$ and $P_{ij}(r_3)$. Update the optimal value of the population. Define $g1_{best}$ as the best value of the particle population, and $g1_{best} = min(P1_{ibest}), (i = 1,2,\cdots P1_{enum3})$ ($enum3$ is the number of the popular size in the third level PSO). If $g1_{best}^{r_3} < g1_{best}$, set $g1_{best} = g1_{best}^{r_3}$, update $P_{gj}(r_3)$; otherwise, $g1_{best}$ and $P_{gj}(r_3)$ retain.

Step2b(v): Termination. Set $r_3 = r_3 + 1$, and check the condition that if $r_3 \le r_{3\max}$, then go to Step**2b(iii)**, start a new iteration; otherwise, terminate the third level PSO and calculate the minimum fitness for the JMAMOM-particles. The minimum fitness, corresponding to the best JMAMOM-particle and corresponding best scheduling solution are recorded.

Step 2c: The best JMAMOM-particle and minimum fitness for each JMAM-particle obtained from the third level PSO are delivered to the second level PSO.

Step 2d: Fitness. The fitness of each JMAM-particle is the fitness of the corresponding best JMAMOM-particle. Then, find the local best solution for all JMAM- particles and the global best solution.

Step 2e: Update the velocity and position of each JMAM-particle. The velocity is updated according to Equation 3-3 and then the position is updated according to Equation 3-4. Finally, the inertia factor W is updated according to Equation 3-5. The parameters have the same meaning and only need to change r_3 into r_2 to represent the second level iteration.

Step 2f: Fitness. To get the fitness of each new JMAM-particle, again pass every JMAM-Particle one by one to the third level PSO, repeat the process from Step2b-Step2c. Then, update the optimal value of each JMAM-particle. $P2_{ibest}$ is the best value for JMAM-particle i during the iteration process and $P2_i^{r_2}$ is the current fitness of JMAM-particle i. If $P2_i^{r_2} < P2_{ibest}$, then set $P2_{ibest} = P2_i^{r_2}$, update $P_{ij}(r_2)$; otherwise, $P2_{ibest}$ and $P_{ij}(r_2)$ retain. Update the optimal value of the population. Define $g2_{best}$ as the best value of the particle population, and $g2_{best} = min(P2_{ibest}), (i = 1,2,\cdots P2_{enum2})$ ($enum2$ is the number of the popular size in the second level PSO). If $g2_{best}^{r_2} < g2_{best}$, set $g2_{best} = g2_{best}^{r_2}$, update $P_{gj}(r_2)$; otherwise, retain $g2_{best}$ and $P_{gj}(r_2)$.

Step 2g: Termination. Set $r_2 = r_2 + 1$, and check the condition that if $r_2 \le r_{2\max}$, then go to Step 2e, and start a new iteration.; otherwise, terminate the second level PSO and calculate the minimum fitness for the JMAM-particles, which is the fitness of each JM-particle.

Step 3: The best JMAM-particle (with its corresponding best JMAMOM-particle obtained from the third level PSO) and the minimum fitness for each JM-particle obtained from the second level PSO are delivered to the first level PSO.

Step 4: Fitness. Measure the fitness of each JM-particle in the population and find the local best solution for all JM-particles and the global best solution.

Step 5: Update the velocity and position of each JM-particle. The velocity is updated according to Equation 3-3, and then the position is updated according to Equation 3-4. Finally, the inertia factor W is updated according to Equation 3-5. The parameters have the same meanings and only need to change r_3 into r_1 to represent the first level iteration.

Step 6: Fitness. To get the fitness of each new JM-particle, again pass each JM-Particle one by one to the second level PSO, repeat the process from **Step 2-Step 3**. Then, update the optimal value of each JM-particle. P_{ibest} is the best value for JM-particle i during the iteration process and $P_i^{r_1}$ is the current fitness of JM-particle i. If $P_i^{r_1} < P_{ibest}$, then set $P_{ibest} = P_i^{r_1}$, update $P_{ij}(r_1)$; otherwise, retain P_{ibest} and $P_{ij}(r_1)$. Update the optimal value of the population. Define g_{best} as the best value of the particle population, and $g_{best} = min(P_{ibest}), (i = 1,2, \cdots P_{enum1})$ ($enum1$ is the number of the popular size in the first level PSO). If $g_{best}^{r_1} < g_{best}$, set $g_{best} = g_{best}^{r_1}$, update $P_{gj}(r_1)$; otherwise, retain g_{best} and $P_{gj}(r_1)$.

Step 7: Termination. Set $r_1 = r_1 + 1$ and check the condition that if $r_1 \le r_{1\max}$, then go to **Step 5**, and start a new iteration.; otherwise, terminate the first level PSO and calculate the best value for each JM-particle i and the best scheduling solution of JMAMOM- particle for each JM-particle i is recorded.

End: the best scheduling solution (JMAMOM-particle) for each JM-Particle is recorded.

Figure 3-16. shows the flowchart of TLPSO. In this flowchart, an example containing 5 jobs and 3 machines is given to illustrate the process of TLPSO. The swarm size is assumed to be 3 for all these three level PSOs. Firstly, in the first level PSO, the positions of 3 JM-particles are randomly generated, such as (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6); (0.25 0.3 0.6 0.8 0.7 0.5 0.9 0.1 0.4 0.5); (0.9 0.54 0.1 0.4 0.6 0.8 0.1 0.5 0.7 0.1) and the velocities are also produced randomly. The dimension of these positions and velocities is 10. Then, each of these JM-particles will be passed to the second level PSO to determine the machine maintenance. The first JM-particle is taken as an example. The positions of 3 JMAM-particles with dimensions of 15 are randomly generated, such as (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 <u>0.5 0.3 0.35 0.6 0.8</u>); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 <u>0.4 0.6 0.1 0.2 0.9</u>); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 <u>0.6 0.7 0.2 0.7 0.1</u>). The first 10 dimensions of these three JMAM-particles are the same as its related JM-particle. Three velocities are generated. Since in the second level PSO, we only focus on the machine maintenance and the production scheduling is not changed, the first 10 dimensions of these velocities are zero. The remaining 5 dimensions of velocities are generated randomly. These JMAM-particles are passed into the third level PSO one by one. The first JMAM-particle is taken as an example. In the third level PSO, the positions of three JMAMOM-particles are generated randomly, such as (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.5 0.3 0.35 0.6 0.8 0.4 0.3 0.6 0.1 0.9); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.5 0.3 0.35 0.6 0.8 0.7 0.2 0.4 0.6 0.5); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.5 0.3 0.35 0.6 0.8 0.5 0.6 0.2 0.1 0.8). The first 15 dimensions of these three JMAMOM-particles are the same as its related JMAM- particle. Three velocities are produced randomly. The first 15 dimensions of these velocities are zero and the remaining 5 dimensions of velocities are generated randomly. With random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007) described in Section 4.2, the positions of the JMAMOM-particles are decoded into suitable scheduling solutions. The fitness value (makespan) of each JMAMOM-particle could be computed. The progress of simple PSO is conducted until the stopping condition is met and the best JMAMOM-particle (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.5 0.3 0.35 0.6 0.8 0.4 0.2 0.5 0.7 0.1) for the first JMAM-particle can be found. Then, the second and the third JMAM-particles will be passed into the third level PSO, and the simple PSO will be carried out again. Through the third level PSO, the best JMAMOM-particle and the fitness value (makespan) for each JMAM-particle are obtained and returned to the second level PSO. The second level PSO is then conducted until the best JMAM-particle (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.7 0.3 0.9 0.6 0.2) and the related best JMAMOM-particle (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.7 0.3 0.9 0.6 0.2 0.4 0.6 0.7 0.3 0.1) are found for the first JM-particle. Using the same process, we can find the best JMAM-particle and the best JMAMOM-particle for the other two JM-particles. Then the basic progress of the first level PSO is conducted before the stopping criterion is satisfied. Finally, three best scheduling solutions for the JM-particles are recorded, namely, (5 2 3 1 4 3 1 3 1 2 1 1 0 1 0 0 1

0 1 1); (3 4 5 2 1 3 2 1 3 1 1 0 1 0 0 1 0 1 0 0); (4 1 3 2 5 2 3 1 3 1 1 0 1 0 1 1 0 1 1 0).

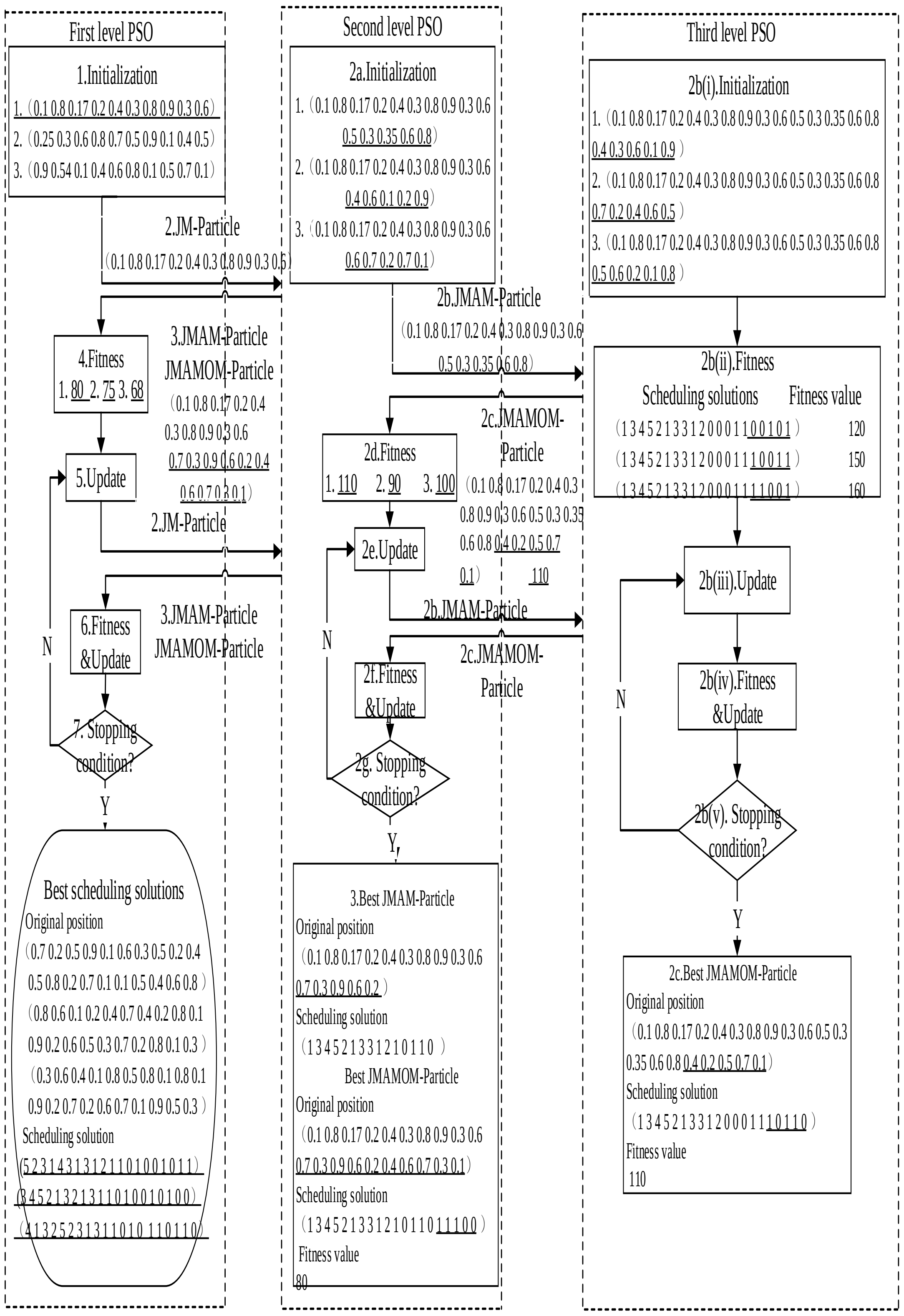


Figure 3-16 Flowchart of TLPSO

3.3.4 Intensification Phase via VNS

For a good algorithm, the balance of the generic search and local search is important. This hybrid algorithm adopts Variable Neighbourhood Search (VNS) to enhance its local search ability. Through TLPSO, some good scheduling solutions can be obtained, and VNS is used to enhance these solutions. Seven kinds of neighbourhoods are designed to make it more suitable for this integrated problem. Specifically, m and n are random integers in [1, P] (P represents the number of jobs) and m is smaller than n. U is a scheduling solution obtained from TLPSO. The definitions of these seven neighbourhoods are given as follows:

1. The first neighbourhood search is to change the $(m+3\times P)^{th}$ dimension of the scheduling solution from 0 to 1 or from 1 to 0 (change the mould maintenance of job m), which is abbreviated as $Change(U, m+3\times P)$. Assuming $m=2, n=4$, an original scheduling solution (5 2 3 1 4 3 1 3 1 2 1 1 0 1 0 0 ***1*** 0 1 1) is changed into (5 2 3 1 4 3 1 3 1 2 1 1 0 1 0 0 ***0*** 0 1 1) with the first neighbourhood search.

2. The second neighbourhood search is to insert the $(m+3\times P)^{th}$ dimension of the scheduling solution to the $(n+3\times P)^{th}$ dimension (change the mould maintenance from job m to job n), which is abbreviated as $Insert(U, m+3\times P, n+3\times P)$. Assuming $m=2, n=4$, an original scheduling solution (5 2 3 1 4 3 1 3 1 2 1 1 0 1 0 0 ***1*** 0 ***1*** 1) is changed into (5 2 3 1 4 3 1 3 1 2 1 1 0 1 0 0 ***0*** 1 ***1*** 1) with the second neighbourhood search.

3. The third neighbourhood search is to change the $(m+2\times P)^{th}$ dimension of the scheduling solution from 0 to 1 or from 1 to 0 (change the machine maintenance of job m), which is abbreviated as $Change(U,m+2\times P)$. Assuming $m=2,n=4$, an original scheduling solution (5 2 3 1 4 3 1 3 1 2 1 ***1*** 0 1 0 0 1 0 1 1) is changed into (5 2 3 1 4 3 1 3 1 2 1 ***0*** 0 1 0 0 1 0 1 1) with the third neighbourhood search.

4. The fourth neighbourhood search is to insert the $(m+2\times P)^{th}$ dimension of the scheduling solution to the $(n+2\times P)^{th}$ dimension (change the machine maintenance from job m to job n), which is abbreviated as $Insert(U,m+2\times P,n+2\times P)$. Assuming $m=2,n=4$, an original scheduling solution (5 2 3 1 4 3 1 3 1 2 1 ***1*** 0 ***1*** 0 0 1 0 1 1) is changed into (5 2 3 1 4 3 1 3 1 2 1 ***0*** 1 ***1*** 0 0 1 0 1 1) with the fourth neighbourhood search.

5. The fifth neighbourhood search is to exchange the m^{th} dimension of the scheduling solution with the n^{th} dimension (change the job sequence of job m and job n), which is abbreviated as $Exchange(U,m,n)$. Assuming $m=2,n=4$, an original scheduling solution (5 ***2*** 3 ***1*** 4 3 1 3 1 2 1 1 0 1 0 0 1 0 1 1) is changed into (5 ***1*** 3 ***2*** 4 3 1 3 1 2 1 1 0 1 0 0 1 0 1 1) with the fifth neighbourhood search.

6. The sixth neighbourhood search is to insert the m^{th} dimension of the scheduling solution to the n^{th} dimension (change the job sequence from job m to job n), which is abbreviated as $Insert(U,m,n)$. Assuming $m=2,n=4$, an original scheduling solution (5 ***2*** 3 ***1*** 4 3 1 3 1 2 1 1 0 1 0 0 1 0 1 1) is changed into (5 ***3*** 1

2 4 3 1 3 1 2 1 1 0 1 0 0 1 0 1 1) with the sixth neighbourhood search.

7. The seventh neighbourhood search is to change the $(m+P)^{th}$ dimension to an available number (change machine of the job m to another available machine), which is abbreviated as $Changemachine(U,m)$. Assuming $m=2, n=4$, an original scheduling solution (5 2 3 1 4 3 ***1*** 3 1 2 1 1 0 1 0 0 1 0 1 1) is changed into (5 2 3 1 4 3 ***2*** 3 1 2 1 1 0 1 0 0 1 0 1 1) with the seventh neighbourhood search.

The process of the VNS is given as follows:

Step 1. Set $Loop = 0$ and get the solution U.

Step 2. Shaking. Generate a solution S randomly from the first neighbourhood of the solution U.

Step 3. Set $Count = 0$, when $Count \leq 6$, conduct Step 3a.

Step 3a. Local Search. Apply the according local search method with the solution S and get a new solution S_1.

Step 3b. Move or not. If the new solution S_1 is better than the solution S, then use S_1 to replace S and the first neighbourhood is still used as the search space, otherwise, set $Count = Count + 1$ and conduct step 3a.

Step 4. Set $Loop = Loop + 1$. when $Loop \leq P \times (P-1)$ conduct step 3. If the performance of the solution U is worse than the performance of the solution S, replace U with S.

The pseudo-code of the local search is demonstrated in Figure 3-17. After all the solutions are finished in the processing of VNS, the solution with the best fitness value is chosen as the final solution.

```
$S_0$=U, Scheduling solution of particle obtained from TLPSO
m=rand (1, P), n=rand (1, P), m<n
S= Change (U, m+3*P)
Loop=0;
Do {
      Count=0
      Max=6
      Do
      {
      If(count==0) then {$S_1$=Change (S, m+3*P)}
      If(count==1) then {$S_1$=Insert (S, m+3*P, n+3*P)}
      If(count==2) then {$S_1$= Change (S, m+2*P)}
      If(count==3) then {$S_1$=Insert (S, m+2*P, n+2*P)}
      If(count==4) then {$S_1$=Exchange (S, m, n)}
      If(count==5) then {$S_1$=Insert (S, m, n)}
      If(count==6) then {$S_1$=Changemachine (S, m)
      If (f($S_1$) <f(S)) then {
      Count=0;
      S=$S_1$
      }
      Else{count++}
      } while (count<Max)
Loop++
} while loop<=P*(P-1)
If f(S)<=f(U) then {
U=S
}
```

Figure 3-17 Pseudo-code of VNS

3.4 Numerical Experiments

The main objective of the numerical experiments is to test the optimization performance of the proposed TLPSO-VNS algorithm. For a fair comparison, the maintenance scheme of the resources and three datasets generated by Wong, Chan, and Chung (2012) are adopted. The sizes of these three problems are ($30\times3\times5$), ($40\times6\times10$) and ($60\times9\times15$). The quality of the solutions produced by the proposed TLPSO-VNS algorithm will be verified by comparing the results obtained by GADG (Strategy 4) (Wong, Chan, and Chung 2012) and results generated by adapted NSGA-

II (Wang and Liu 2015). Furthermore, some recent VNS variants (Hybrid Variable Neighbourhood Search (HVNS) algorithm (Zhang et al. 2018), the Nested General Variable Neighbourhood (NGVNS) algorithm (Todosijević, 2016) and PSO variants (Particle Swarm Optimization and Artificial Bee Colony Hybrid algorithm (ABCSPSO) (El-Abd 2013 2013), self-adaptive heterogeneous Particle Swarm Optimization (fk-PSO) (Nepomuceno and Engelbrecht 2011), Standard Particle Swarm Optimization 2011 (SPSO) (Zambrano-Bigiarini et al. 2013) proposed in the CEC'2013 competitions are used as the comparison algorithms to illustrate the performance of the TLPSO-VNS algorithm deeply. Six random instances with different sizes are used as benchmarks. To analysis the significant difference between the results, the Wilcoxon test is applied.

Numerical experiments are implemented in the Matlab environment and the statistical tests are conducted by the IBM SPSS Software on a personal computer with Intel (R) Core (TM) i7-6700 CPU 3.40GHz CPU.

3.4.1 Parameters Tuning

There are two kinds of parameters. One kind is the key algorithm parameters, namely the maximum number of generations and the swarm size. The other kind is the parameters related to simple PSO, including inertia weight W, particle acceleration coefficient C_1 and population acceleration coefficient C_2. At first, the parameters related to PSO are fixed. We use the parameters combination recommended by

Eberhart and Shi (2000), which are also the most widely used parameters combination related to PSO in the literature. The maximum number of generation and the swarm size are decided. Then, the key algorithm parameters are fixed, and we try different parameter combinations related to PSO and find the best parameters combination.

The production scheduling problem is the master problem and the machine maintenance problem, as well as mould maintenance problem, has a similar influence on the makespan. Without loss of generality, we set the Max Iteration of the second level PSO as equivalent to the Max Iteration of the third level PSO and the Max Iteration of the first level PSO is the double value of Max Iteration of the second level PSO. According to Mladenović and Hansen (1997), the Max Iteration of VNS is set as $P \times (P-1)$ (the number of jobs is represented by P), which is sufficient to acquire stable solutions for the problem with the largest size. The swarm size of the three PSOs in the different levels are the same. W is initially set as 0.9 and reduced linearly to 0.4. The values of C_1 and C_2 are equal to 2. This parameter combination related to PSO is the most often used in the literature. In the preliminary testing on key algorithm parameters, we test different combinations of parameters with swarm size={5,10,15} and Max Iteration of first level PSO={500,1000,1500} for medium-sized instance (40 $\times$ 6 $\times$ 10) generated by Wong, Chan, and Chung (2012) 30 times. There are 9 combinations in total. Table 3-4 shows the different parameter combinations, the corresponding average results and the time needed for each run. From Table 3-4, we can see that the swarm size has a great influence on the time needed for each run. When the swarm size increases 2 times (from 5 to 10), the time needed for each run increases by more than 5 times. When the swarm size increases 3 times (from 5 to 15), the time needed for each run increases by more than 10 times. The Max Iteration of the first level PSO has little influence on the time and the average result. A bigger Max Iteration does not always mean better results. To keep a balance between the time needed for each run and the quality of results, we choose the parameter combination{10, 1000}.

Table 3-4 Different key algorithm parameter combinations
and their influence on results.

No	Swarm size	Max Iteration of first level PSO	Average Result	Time(s)
1	5	500	2554	345
2	5	1000	2526	388
3	5	1500	2515	427
4	10	500	2546.3	1590
5	**10**	**1000**	**2492.6**	**1704**
6	10	1500	2526.7	2254
7	15	500	2526	4010
8	15	1000	2427.8	4988
9	15	1500	2546.1	5708

To find a good combination for the parameters related to PSO, we set the swarm size as 10, the Max Iteration of the first level PSO as 1000, the Max Iteration of the second level PSO as 500 and the Max Iteration of the third level PSO as 500, based on the preliminary testing. We compare the most widely used parameters combination (Eberhart and Shi, 2000) with the other four PSO models, namely social only model (there is no cognitive component, $C_1 = 0$) (Kennedy and Eberhart 1995); cognition only model (there is no social acceleration, $C_2 = 0$) (Kennedy and Eberhart 1995); time-varying acceleration coefficient model (C_1 starts with a high value than C_2 and decreases while the social acceleration starts with a lower value and linearly increases) (Ratnaweera et al. 2004); Clerc's constriction method (Eberhart and Shi 2000). The maintenance scheme of resources and the medium-sized instance ($40 \times 6 \times 10$) generated by Wong, Chan, and Chung (2012) is used, and we test each parameter combination 30 times. The results are demonstrated in Table 3-5.

Table 3-5 The influence of parameters related to PSO on results.

No.	Parameters in the PSO model	Swarm size	Max Iteration of first level PSO	average	Std
1	W_{max}=0.9, W_{min}=0.4, C_1=2, C_2=2	10	1000	**2492.6**	58.7
2	W_{max}=0.9, W_{min}=0.4, C_1=0, C_2=2	10	1000	2497.5	64
3	W_{max}=0.9, W_{min}=0.4, C_1=2, C_2=0	10	1000	2583.3	77
4	W_{max}=0.9, W_{min}=0.4, C_{1max}=2.5, C_{1min}=0.5, C_{2max}=2.5, C_{2min}=0.5	10	1000	2538.6	12
5	W=0.729, C_1=1.49445, C_2=1.49445	10	1000	2525.7	39

In Table 3-5, it can be seen that the difference between these parameter combinations is little, which means that the PSO related parameters have little influence on the results. Finally, we use the most widely used parameter combination recommended by

Eberhart and Shi (2000) to solve this problem. In our random test, the average value and the standard deviation are the minima compared with other parameter combinations. However, other parameters combination can also produce good results for this problem.

3.4.2 Comparison with Results in the Literature

The maintenance scheme of resources and three datasets with different sizes generated by Wong, Chan, and Chung (2012) are adopted. The comparison results are shown in Table 3- 6.

Table 3-6 Comparison with results in the literature.

	$30\times3\times5$			$40\times6\times10$			$60\times9\times15$		
	Min	Avg	SD	Min	Avg	SD	Min	Avg	SD
Results by GADG	2250	2475	107	2576	2757	144	2417	2721	183
Results by NSGA-II	**2160.7**	2255.9	78.6	2490	2591.7	68.6	2508	2586.3	60.1
Results by TLPSO-VNS	**2177.3**	**2251.6**	**45.5**	**2422**	**2492.6**	**58.7**	**2378.5**	**2431.5**	**27.3**
Result improvement (compared with GADG)	3.2%	9%		6%	9.6%		1.6%	10.6%	
Result improvement (compared with NSGA-II)	-0.7%	0.19%		2.7%	3.8%		5.2%	6%	

According to Table 3-6, the average makespans for the three instances obtained by TLPSO-VNS are 2251.6, 2492.6, 2431.5 units of time. Compared with the results generated by GADG, the improvement is between 9% and 10.6%. Compared with the results of NSGA-II, the improvement is between 0.19% and 6%. Moreover, the improvement increases when the scale of the problem grows. It can be concluded that TLPSO-VNS can provide better solutions than GADG and NSGA-II when the scale of the problem grows. The minimum solutions of these three instances obtained by the TLPSO-VNS algorithm are 2177.3, 2422, 2378.5 units of time. Compared with results by GADG, the improvement is between 1.6% and 6%. Compared with results by GADG, the improvement is between -0.7% and 5.2%. The TLPSO-VNS could obtain better minimum solutions compared with GADG and NSGA-II, except for the minimum result for the instance $30 \times 3 \times 5$. But the gap between the minimum results obtained by TLPSO-VNS and NSGA-II is quite small, only 0.7%. Furthermore, the TLPSO-VNS algorithm is more robust than GADG and NSGA-II because the standard deviations of the TLPSO-VNS algorithm are smaller for all these three instances of different sizes. The production scheduling for the optimal results of the first instance is shown in Figure 3-19. Gantt chart of instance 1 ($30 \times 3 \times 5$). Specifically, MT means maintenance and the numbers in the boxes are the sequences of jobs.

3.4.3 Comparison with Recent VNS Variants and PSO Variants

In this subsection, the presented TLPSO-VNS is compared with recent VNS variants (HVNS (Zhang et al. 2018) and NGVNS (Todosijević, et al. 2016)) and PSO variants in CEC’2013 ((ABCSPSO (El-Abd, 2013), fk-PSO (Nepomuceno and Engelbrecht 2013), and SPSO2011(Zambrano-Bigiarini et al. 2013)).

Six random instances with different sizes are used to explore the performance of the developed TLPSO-VNS algorithm. Based on the recommendation of Wong, Chan, and Chung (2012), the parameters of the six random instances are chosen as follows: the processing time of the jobs is produced between 30 and 55 units of time randomly; the batch size of the jobs is produced between 2 and 6 units randomly. 30 times is carried out for each algorithm to calculate the deviation of the acquired solutions. The maintenance scheme of the resources generated by Wong, Chan, and Chung (2012) is adopted. For a fair comparison, algorithm parameters of the VNS variants including neighbourhoods definition and the maximum number of iteration are the same as the proposed TLPSO-VNS and algorithm parameters of the PSO variants, including the parameters related to PSO, the maximum number of generations and the population size of these three algorithms are the same as the proposed TLPSO-VNS. The comparison results are demonstrated in Table 3- 7.

From the Table 3-7, we can know that when the iteration and the parameters are the same, the proposed TLPSO-VNS algorithm has better performance than both VNS variants and PSO variants with regard to the minimum, maximum, average results and the standard deviation for all the six instances. Overall, the performance of VNS variants surpasses the performance of PSO variants. However, the proposed TLPSO-VNS algorithm also needs more time than all the comparison algorithms. Figure 3-18 shows the time increase trends of these six algorithms for these six instances.

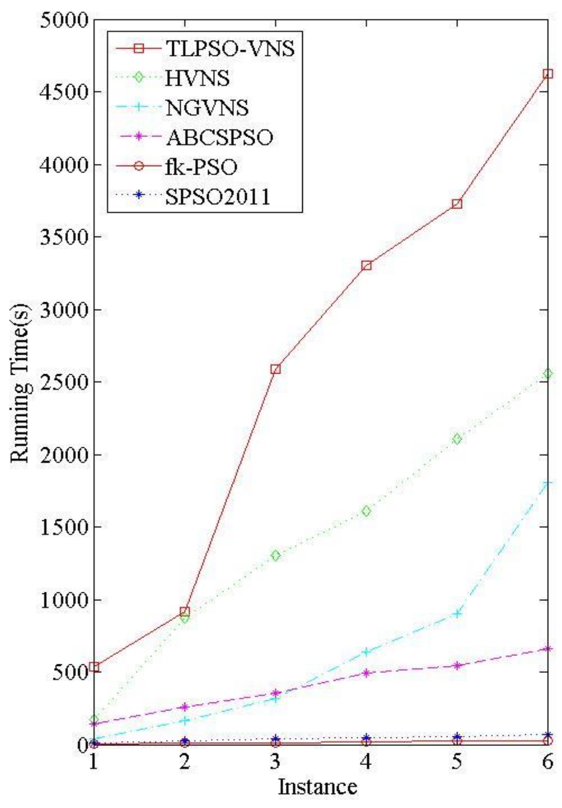


Figure 3-18 Time increase trends for the six algorithms

From Figure 3-19, we can know that as the scale of the problem grows, the CPU time needs for the proposed TLPSO-VNS algorithm increases a lot, but the amount of the increase for the other five comparison algorithms is not so obvious. The main reason is that as the scale of the problem increases, the TLPSO-VNS algorithm needs more time to conduct the deep search in the Three-Level PSO as well as VNS to guarantee the quality of the solutions compared with other algorithms.

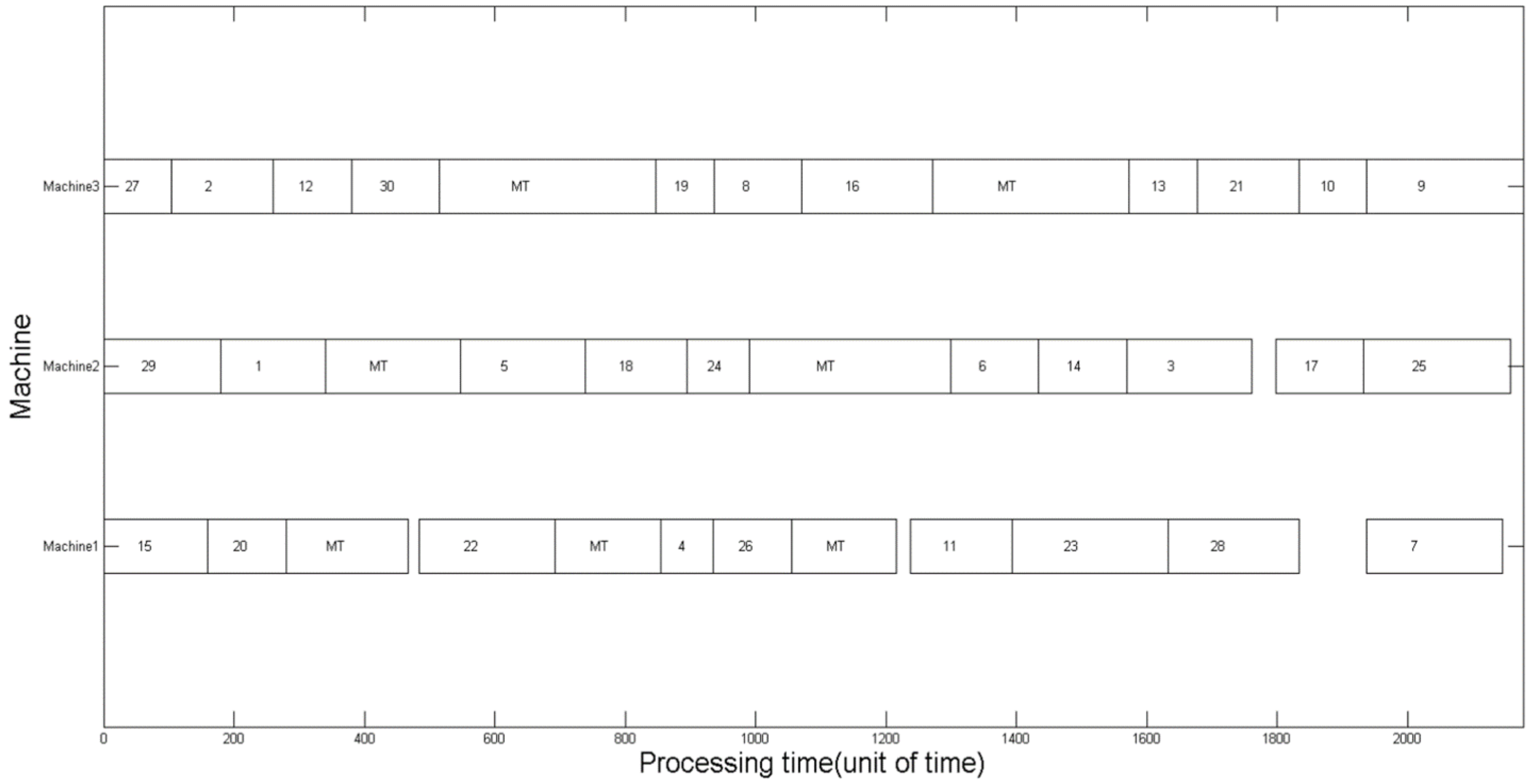


Figure 3-19a Machine schedule for instance ($30 \times 3 \times 5$)

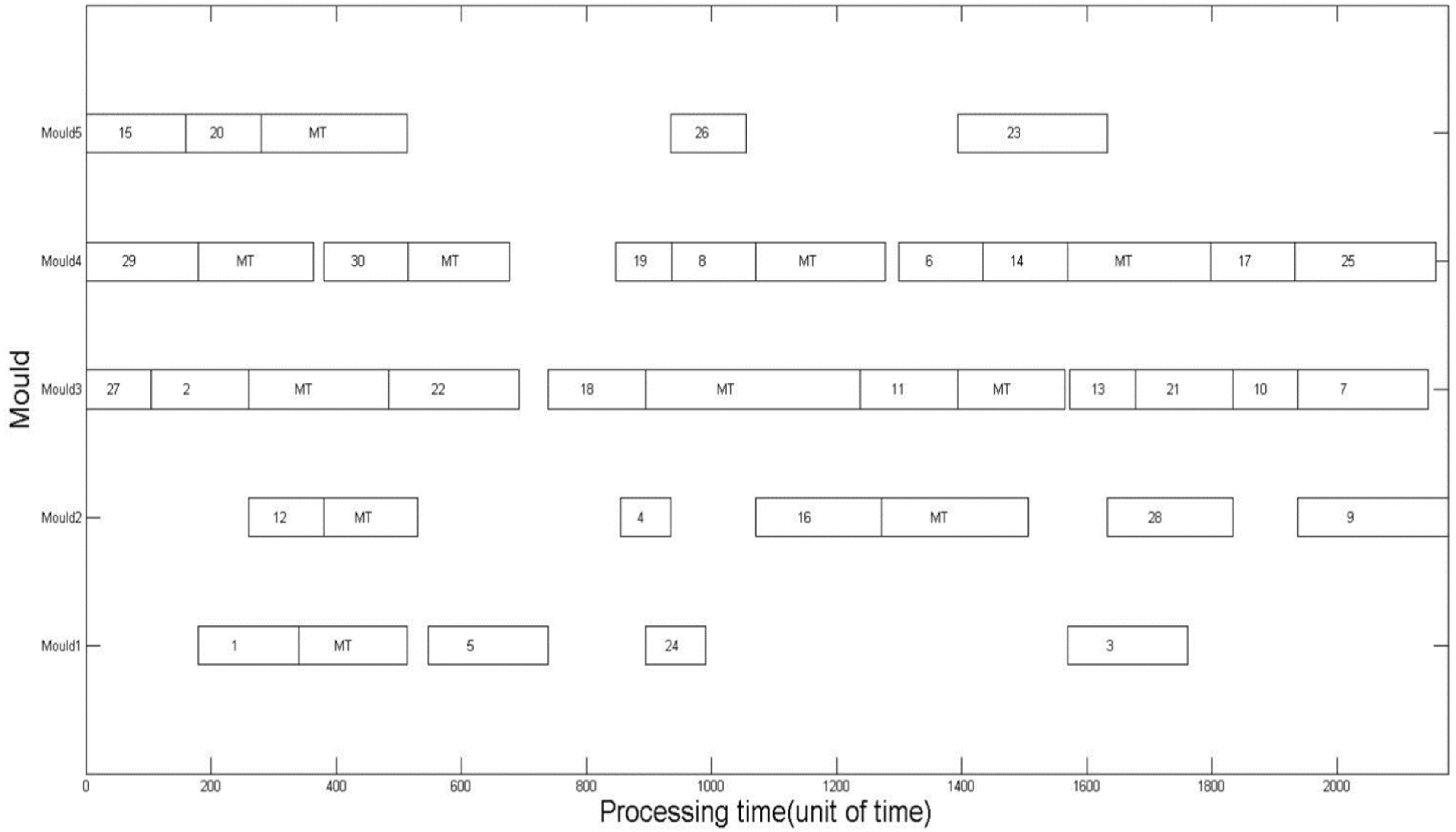


Figure 3-19b Mould schedule for instance ($30 \times 3 \times 5$)

Figure 3-19 Gantt charts of instance ($30 \times 3 \times 5$)

Furthermore, since the time need for the proposed TLPSO-VNS algorithm is longer than other comparison algorithms, to better illustrate the performance of the proposed algorithm, the comparison algorithms are carried out under the same computational time that the proposed TLPSO-VNS algorithm needs for all the instances. Each algorithm is run 30 times and the results are given in Table 3-8.

From Table 3-8, we can see that TLPSO-VNS surpasses VNS Variants and PSO Variants for these six instances with regard to the average value, the minimum value and the maximum value under the same computational cost. To deeply analyse the difference between the TLPSO-VNS and the other comparison algorithms, the Wilcoxon Signed Rank Test for these results is conducted. The Null hypothesis is that there is no median difference between the two algorithms. The significant level is 0.05. If the P value is smaller than 0.05, it means that the Null hypothesis is rejected. Otherwise, the Null hypothesis is retained. The results of the Wilcoxon Signed Rank Test are shown in Table 3-9. From Table 3-9, we can know that there are no median differences between TLPSO-VNS and PSO variants as well as HVNS because the corresponding P values are smaller than the significant level of 0.05. For most of the instances, there is no median difference between the TLPSO-VNS and NGVNS. However, for instance $35 \times 4 \times 6$ and instance $50 \times 7 \times 10$, the medians of TLPSO-VNS and NGVNS are the same with a 95% confidence level. Since the average value and the standard deviation of TLPSO-VNS are better than NGVNS, we can still consider that TLPSO-VNS is better than NGVNS. Since the TLPSO-VNS consists of three interrelated PSO, the local search ability is greatly improved but it needs more time to conduct deep searching. VNS, as a good search tool, can produce relatively good solutions, however, VNS needs more time as the size of the problem grows. After hybridizing TLPSO with VNS, VNS enhances the local search ability of TLPSO and the solutions produced by TLPSO act as the initial solutions of VNS. The search ability of TLPSO-VNS is greatly improved, but the time needed is also longer.

Table 3-7 Comparison results with other algorithms under the same iteration.

No	Size	Algorithm	Min	Max	Avg	SD	Time (s)
1	20×2×4	TLPSO-VNS	2328	2424	2365	35	535
		HVNS	2391	2691	2471.3	86	168
		NGVNS	2453	2535	2492.7	27	41
		ABCSPSO	2504	2974	2704	147	139
		fk-PSO	2763	3236	2933.7	142	5
		SPSO2011	2517	2806	2652.6	92	14.2
2	35×4×6	TLPSO-VNS	2423	2690	2539	82	916
		HVNS	2702	3017	2802	106	869
		NGVNS	2555	2876	2702	89	162
		ABCSPSO	3029	3827	3545	251	261
		fk-PSO	3874	4707	4234.6	316	9
		SPSO2011	3273	4216	3872.6	276	26
3	50×7×10	TLPSO-VNS	2189	2425	2281.7	90	2585
		HVNS	2204	2634	2413.8	144	1301
		NGVNS	2203	2351	2293	43	314
		ABCSPSO	2829	3391	3036.6	169	352
		fk-PSO	3695	4157	3893	138	14
		SPSO2011	3329	3717	3483.4	121	38
4	70×9×12	TLPSO-VNS	3672	3816	3700.6	45	3302
		HVNS	3772	4477	4095.3	189	1610
		NGVNS	3844	4115	3936.4	94	639
		ABCSPSO	3905	4336	4128	149	492
		fk-PSO	4422	4423	4892	229	16
		SPSO2011	4615	5073	4830	143	48
5	80×10×16	TLPSO-VNS	3200	3355	3279	46	3728
		HVNS	3360	4452	3856	301	2103
		NGVNS	3593	4361	3924	210	904
		ABCSPSO	3634	4028	3866.0	143	542
		fk-PSO	4563	5351	5048	229	23
		SPSO2011	4617	5022	4787	142	56
6	100×12×20	TLPSO-VNS	3487	3809	3623.5	94	4626
		HVNS	3815	4398	4026.6	197	2560
		NGVNS	3744	4428	3906	200	1808
		ABCSPSO	3942.0	4762.3	4368.0	250	659
		fk-PSO	5523.7	6101.5	6004.7	269	22
		SPSO2011	5224.8	6433.5	5729.5	264	67.2

Table 3-8 Comparison results with other algorithms under the same computational cost.

No	Size	Algorithm	Min	Max	Avg	SD
1	$20\times2\times4$	TLPSO-VNS	2328	2424	2365	35
		HVNS	2341	2513	2429.2	50
		NGVNS	2324	2434	2396.1	39
		ABCSPSO	2395	2812	2607.7	131
		fk-PSO	2630	3081	2866.9	151
		SPSO2011	2383	2718	2571.7	103
2	$35\times4\times6$	TLPSO-VNS	2423	2690	2539	82
		HVNS	2526	2749	2661	60
		NGVNS	2387	2850	2611	126
		ABCSPSO	3112	3651	3380.5	208
		fk-PSO	3914	4654	4407.5	233
		SPSO2011	3373	4073	3636.8	204
3	$50\times7\times10$	TLPSO-VNS	2189	2425	2281.7	90
		HVNS	2264	2450	2385	64
		NGVNS	2205	2547	2383	126
		ABCSPSO	2600	2942	2807	106
		fk-PSO	3554	4101	3854.7	193
		SPSO2011	3086	3322	3219	76
4	$70\times9\times12$	TLPSO-VNS	3672	3816	3700.6	45
		HVNS	3672	4138	3863.6	162
		NGVNS	3672	4016	3847.6	104
		ABCSPSO	3726	4368	4050	238
		fk-PSO	4798	5418	5053.5	183
		SPSO2011	4241	5063	4531	257
5	$80\times10\times16$	TLPSO-VNS	3200	3355	3279	46
		HVNS	3469	3784	3606	116
		NGVNS	3225	3933	3586	229
		ABCSPSO	3549	4398	3784.8	244
		fk-PSO	4723	5427	5174	252
		SPSO2011	4347	4883	4639.5	167
6	$100\times12\times20$	TLPSO-VNS	3487	3809	3623.5	94
		HVNS	3626	4118	3857.9	135
		NGVNS	3400	4014	3775	181
		ABCSPSO	3838	4904	4290.7	296
		fk-PSO	5198	6852	6339	346
		SPSO2011	5886	4914	5421.6	281

Table 3-9 Wilcoxon signed rank test between TLPSO-VNS and comprasion algorithms

No	Size	Algorithm Comparison	P value
1	20×2×4	TLPSO-VNS VS HVNS	0.022
		TLPSO-VNS VS NGVNS	0.008
		TLPSO-VNS VS ABCSPSO	0.007
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.007
2	35×4×6	TLPSO-VNS VS HVNS	0.022
		TLPSO-VNS VS NGVNS	0.139
		TLPSO-VNS VS ABCSPSO	0.005
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.007
3	50×7×10	TLPSO-VNS VS HVNS	0.032
		TLPSO-VNS VS NGVNS	0.075
		TLPSO-VNS VS ABCSPSO	0.005
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.005
4	70×9×12	TLPSO-VNS VS HVNS	0.007
		TLPSO-VNS VS NGVNS	0.015
		TLPSO-VNS VS ABCSPSO	0.005
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.005
5	80×10×16	TLPSO-VNS VS HVNS	0.005
		TLPSO-VNS VS NGVNS	0.005
		TLPSO-VNS VS ABCSPSO	0.005
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.005
6	100×12×20	TLPSO-VNS VS HVNS	0.007
		TLPSO-VNS VS NGVNS	0.047
		TLPSO-VNS VS ABCSPSO	0.005
		TLPSO-VNS VS fk-PSO	0.005
		TLPSO-VNS VS SPSO2011	0.005

3.5 Conclusions

This research proposes a new hybrid algorithm named the TLPSO-VNS algorithm for Production Scheduling with the Mould Maintenance (PS-MM) problem, which combines the Three-Level Particle Swarm Optimization (TLPSO) algorithm and Variable Neighbourhood Search (VNS). Differing from the joint scheduling approach, this integrated problem is divided into three sub-problems: production scheduling problem, machine maintenance problem and mould maintenance problem. Three interrelated PSOs are used in the solution and is named as Three-Level Particle Swarm Optimization (TLPSO). After obtaining good solutions from TLPSO, VNS is conducted on all these solutions to enhance the local search ability. The best solution is chosen from all the solutions enhanced by VNS. This is the first algorithm to hybrid TLPSO with VNS to solve the production scheduling with mould maintenance (PS-MM) problem, aiming at minimizing the overall makespan. The optimization reliability of TLPSO-VNS is tested in numerical experiments based on data from the literature and other data generated randomly. The results show that the proposed TLPSO-VNS algorithm is effective in generating solutions with good quality in acceptable computation time. It is also shown that the problem decomposition mechanism employed in TLPSO works well in this integrated problem, and VNS is efficient in enhancing the local search ability of this hybrid algorithm. Furthermore, the proposed TLPSO-VNS algorithm is shown to surpass VNS variants and PSO variants in the CEC'2013 competition when solving this integrated problem.

This proposed TLPSO-VNS algorithm is mainly designed for the scheduling problem integrating production scheduling and resource maintenance scheduling simultaneously. The application over plastic manufacturing is only a special case of this kind of problem. For other discrete integrated problems, the problem decomposition mechanism and redesigns the structure of the basic algorithm are also applicable. Moreover, the algorithm hybridization method which combines TLPSO and VNS that are used in the proposed algorithm is also instructive.

However, since this algorithm contains many steps, the time needed to produce good solutions increases a lot when the scale of the problem increases. More strategies will be added to the algorithm to reduce the search repetition, thus improving the efficiency of this hybrid algorithm. In addition, the algorithm will be modified to be appropriate for production scheduling with mould maintenance problems with multi-objectives in the future.

3.6 Summary

This chapter introduces the Production Scheduling problem with Mould Maintenance problem (PS-MM Problem) and the methodology to solve the PS-MM Problem. In the first subsection, the original Production Scheduling problem with Mould Maintenance problem (PS-MM Problem) is restated. In the second subsection, some basic exploration is given about the methodology. Another meta-heuristic algorithm-PSO is applied to resolve the integrated problem. To apply PSO successfully into this problem, random key representation and the smallest position value (SPV) rule are used to transfer the continuous value of the particle position into discrete solutions. Numerical experiments are conducted to illustrate the superiority of PSO, compared with GA. The third subsection presents the proposed hybrid algorithm. Different from the joint scheduling strategy used in the traditional production scheduling problem with maintenance, this research divides the PS-MM problem into three sub-problems, and these three problems are solved by three interrelated PSOs named as TLPSO. A new encoding and decoding method is proposed for the particles in these three PSOs. Furthermore, VNS is used to enhance the solutions obtained by TLPSO. The proposed hybrid algorithm is tested by the three benchmarks proposed by other researchers. The numerical examples illustrate that the hybrid algorithm can produce better results than the existing optimization algorithm. In addition, to analysing the hybrid algorithm deeply, more instances are given, and five more algorithms are compared with the hybrid algorithm. The proposed algorithm is shown to has a superior performance by the computing results.

Chapter 4. Scheduling Multiple Maintenances and Production Scheduling through a Three-Level Particle Swarm Optimization Algorithm

After the basic production scheduling problem with mould maintenance is resolved by a hybrid algorithm, a new research question that is seldom explored is proposed in this chapter, and the mechanism and the methodology proposed in chapter 3 are applied to resolve the new problem. In the first subsection of this chapter, the problem description is presented, and the methodology employed to solve it is introduced in subsection 2, which includes the encoding and decoding method and the details of this algorithm. The numerical experiment and the conclusions are given in subsection 4.3 and subsection 4.4. Finally, this chapter ends with the summary.

4.1 Problem Description

The production scheduling with mould maintenance (PS-MM) problem was firstly proposed by Wong, Chan, and Chung (2012). The model proposed by them contains P jobs, Q injection machines and R injection moulds (we use $P \times Q \times R$ to represent each problem). Both jobs and resources are well prepared at the beginning. For each job, the mould it uses is fixed but the machine it uses needs to be determined. At a given time slot, each mould and machine could perform only one job. Jobs could only be conducted by their batch size and the batch size could not be split. The unit operation time of a job is based on the mould it uses. The total operation time of a job is the product of batch size and the unit operation time. Once a job begins, it could not be interrupted. In the original model, the relationship of the maintenance time and the age of the machine is fitted by a piecewise linear function and only the perfect maintenance is considered, which means that after the maintenance, the machine is as good as new. All the random breakdowns are assumed to be prevented. Moreover, the set-up times and the quality issue are not considered.

However, in the original model, they assume that the less maintenance time is needed if the earlier the maintenance is conducted. This preventive maintenance strategy may result in some unnecessary maintenances, which is a waste of resources. In practice, a resource suffers from degradation and various levels of degradation could be represented by different states. The state of resource can be estimated through condition monitoring. For different states, maintenance strategies are different.

In the plastic industry, the states of the resource (machine and mould) may be decided by many factors such as the temperature, the pressure and so on (Borkowski and Stasiak-Betlejewska 2015). In the manufacturing process, these factors may be varying and have a great effect on the production quality and production rate (Ribeiro 2005). Based on the sensor readings of these parameters, the production rate can be decided, and the states of the resource can be classified. According to Peng and Dong (2011), we assume that the machine and the mould have four health states: normal state, decline stage 1, decline stage 2 and failure. The different stages correspond to different production rates. The expected values of the duration time in each state could be seen from Figure 4-1 and Figure 4-2. In these two Figures, the four different states can be represented by four numbers: the normal state is represented by 1, the decline stage 1 is represented by 2, the decline stage 2 is represented by 3 and the failure is represented by 4.

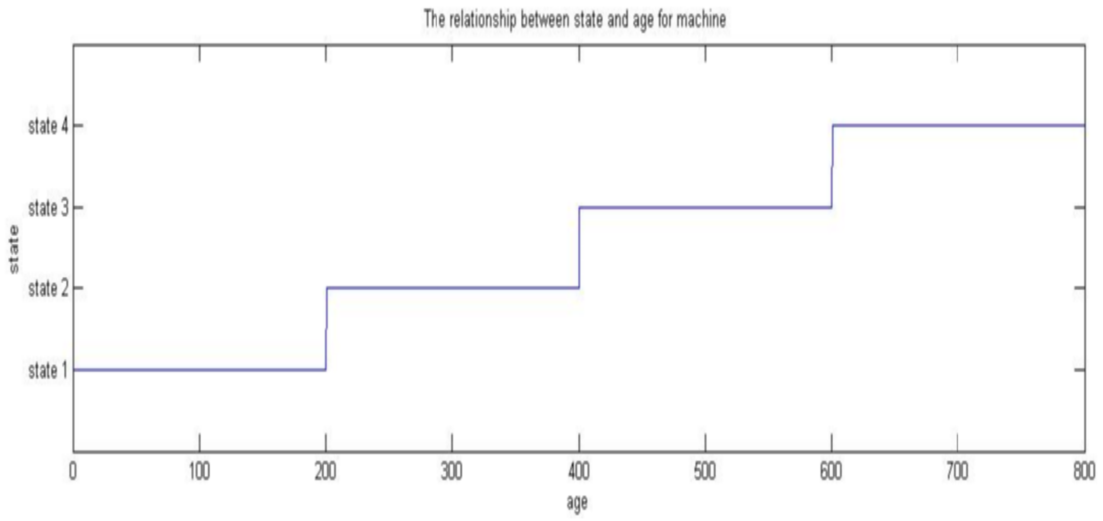


Figure 4-1 The relationship between state and age for machine

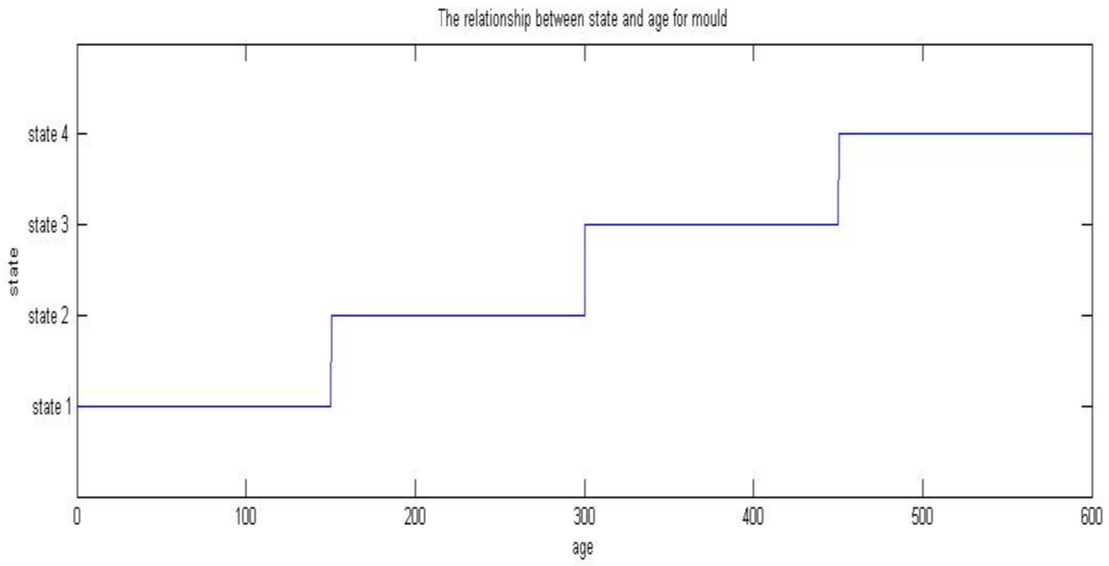


Figure 4-2 The relationship between state and age for mould

In the original model, the maintenance is integrated, which means that various kinds of maintenances are conducted together. While in practice, there are various kinds of maintenances, these maintenances may recover the machine to a state that as good as new, a state as bad as old or just a state between as bad as old and as good as new. According to Duan et al. (2018), there are three kinds of maintenances: the first is inspection such as tightening the loose parts, adjusting/calibrating, lubricating, adding supplements (waters, oil, etc.), and cleaning dust, etc. Such kind of maintenance will not change the condition of the resource and the state of the resource will not change. The second kind of maintenance is intermediate maintenance level, such as component disassembly, reassembly, and calibrations, especially the surface treatments. This kind of maintenance is imperfect maintenance and can make the resource to a better state. The third one is the overhaul or replacement. The machine will be overhauled or replaced by a new one. Usually, this kind of maintenance has to be conducted when the state of the machine is going to fail. This kind of maintenance is conducted to keep away from possible serious damages. For various kinds of maintenance, the maintenance time is different, and the details about the maintenance time for different strategies are shown in Table 4-1.

The objective of this research is to find a good production scheduling and suitable machine maintenance planning, as well as mould maintenance planning to minimize the makespan.

Table 4-1 The maintenance time for different maintenance strategies

The number of maintenance	The type of maintenance	Maintenance time for machine	Maintenance time for mould
1	Inspection (The state is not changed)	40	30
2	Intermediate maintenance (From state 3 to state 2 or from state 2 to state 1)	80	70
3	Intermediate maintenance (From state 3 to state 1)	120	100
4	Replacement or overhaul (From state 4 to state 1)	150	130

4.2 Algorithm Design

In this research, a Three-Level PSO is developed to resolve this integrated optimization problem. In the Three-Level PSO algorithm, there are three PSOs and these three PSOs are interrelated. The first level PSO focus on the production scheduling problem and the particle in the first level PSO contains the information about the job sequence (J) and the corresponding machine sequence (M). We name it as JM-particle with dimension $2 \times P$ (P is the number of jobs). The JM-particle is passed into the second level PSO and the particle in the second level PSO is named as JMAM-Particle with dimension $3 \times P$, which contains the information about the maintenance on the machine (AM) except the job sequence (J) and the machine sequence (M). The JMAM-Particle is sent into the third level PSO and the particle in the third level PSO is named as JMAMOM-Particle with dimension $4 \times P$. The JMAMOM-Particle contains information about the maintenance on the mould (OM) except the information on the job sequence (J), the corresponding machine sequence

(M) and machine maintenance (AM). Through the third level PSO, an optimized JMAMOM-Particle is found for each JMAM-Particle and the optimized JMAMOM-Particle is passed back into the second level PSO. In the second level PSO, an optimized JMAM-Particle along with its corresponding optimized JMAMOM-Particle is obtained and then they are passed back into the first level PSO. In the first level PSO, an optimized JM-Particle is obtained along with its corresponding optimized JMAM-Particle and JMAMOM-Particle. Finally, the solution corresponding to the optimized JMAMOM-Particle is the final optimized solution. The Three-Level PSO algorithm has two parts: the encoding and decoding of particles and the details of the algorithm. The details of the Three-Level PSO are given in subsection 4.2.1 and subsection 4.2.2.

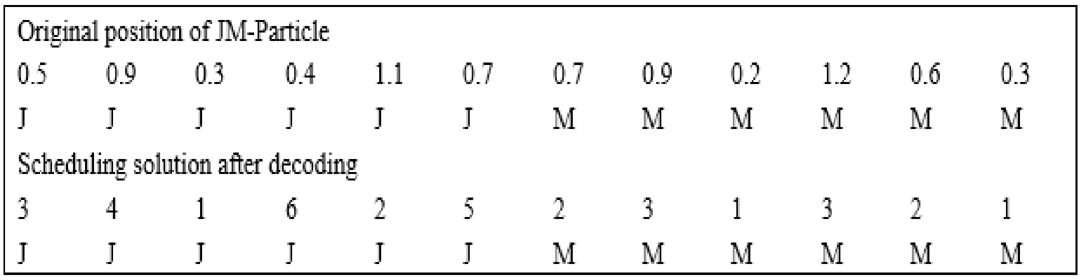
4.2.1 Encoding and Decoding of Particles

There are three particles in this Three-Level PSO (TLPSO) algorithm: the JM-Particle (the dimension is $2 \times P$), the JMAM-Particle (the dimension is $3 \times P$) and the JMAMOM-Particle (the dimension is $4 \times P$). To decode the original particles' position into a suitable scheduling solution for this problem, random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007) are used. After decoding, the values of the J parameters of all the three kinds of particles are integers between 1 and P (P is the number of jobs); values of M parameters of all the three kinds of particles are integers between 1 and Q (Q is the number of machines); The AM parameter is the maintenance decision on the machine.

Since the machine has four different states, the maintenance strategies may be different for different states. The values of the AM parameters are between 0 and 4. The value 0 means that there is no maintenance operation on the machine. 1 means that the basic inspection is conducted. 2 means that the intermediate maintenance can change the machine from state 2 to state 1 or from state 3 to state 2. 3 means that the intermediate maintenance can change the machine from state 3 to state 1. 4 means that the replaced maintenance is conducted to the machine. The OM parameter is the maintenance decision on the mould. The values of the OM parameters are between 0 and 4. The value 0 means that there is no maintenance operation on the mould. 1 means that the basic inspection is conducted and the state of the mould is not changed. 2 means that the intermediate maintenance can modify the mould from state 2 to state 1 or from state 3 to state 2. 3 means that the intermediate maintenance can improve the mould from state 3 to state 1. 4 means that the replaced maintenance is conducted to the mould. Different maintenance strategies have different influences on the makespan.

A series of JM-Particle, JMAM-Particle, JMAMOM-Particle examples before and after decoding are shown in Figure 4-3. This example contains 6 jobs, 3 machines and 2 moulds ($6 \times 3 \times 2$). Jobs 1, 3, 5 could only be processed by Mould 1. Jobs 2, 4, 6 could only be processed by Mould 2. From Figure 4-3c, it can be seen that the value of the job sequence (J) in the original position of the JMAM-particle is (0.5 0.9 0.3 0.4 1.1 0.7), and it is transferred into (3 4 1 6 2 5) by random key representation (Bean

1994) and the smallest position value (SPV) rule (Tasgetiren et al. 2007). The sequence (0.5 0.9 0.3 0.4 1.1 0.7) is ranked based on the ascending order and a new sequence (0.3 0.4 0.5 0.7 0.9 1.1) is obtained. Since the number 0.3 is in the third position of the original sequence, the first position is decoded into 3. Since the number 0.4 is in the fourth position of the original sequence, the second position is decoded into 4. By this rule, a suitable scheduling solution (3 4 1 6 2 5) is obtained. The value of the corresponding machine sequence (M) in the original position of the JMAM-particle is (0.7 0.9 0.2 1.2 0.6 0.3). The interval [0.2 1.2] (0.2 is the minimum and 1.2 is the maximum among all the numbers) into 3 intervals, [0.2 0.53), [0.53 0.87) and [0.87 1.2] (there are 3 machines in this example). Since 0.2 and 0.3 are in the first interval, after decoding, the value in the relevant position is 1. Since 0.6 and 0.7 are in the second interval, after decoding, the value in the relevant position is 2. Since 0.9 and 1.2 are in the third interval, after decoding, the value in the relevant position is 3. So, the corresponding machine sequence (M) can be transferred into (2 3 1 3 2 1). Then, we can know that the machine 1 is used to process job 1 and job 5; The machine 2 is used to process job 3 and job 2; The machine 3 is used to process job 4 and job 6. For different states of machine and mould, the maintenance strategies are different. Assuming that the machine is in state 3 after processing the job 3, there are four possible maintenance strategies (no maintenance, maintain to state 3, maintain to state 2, maintain to state 1). We divide the section [0.1 1] (0.1 is the minimum number in AM parameter section, 1 is the maximum number in AM parameter section) into 4 sections [0.1 0.33), [0.33 0.56), [0.56 0.78), [0.78 1]. Since 0.8 is in the fourth section, the number 0.8 is decoded into 3. So, the maintenance that can make the machine recover into state 1 is conducted. Assuming that the machine 4 is in state 2 after processing job 4, so there are three possible maintenance strategies (no maintenance, maintenance to state 2, maintenance to state 1). The section [0.1 1] is divided into 3 sections [0.1 0.4), [0.4 0.7), [0.7 1]. Since 0.9 is in the third section, the number is decoded into 2, which means that the maintenance makes the machine into state 1 is conducted. With the same method, the value 0.1 is decoded into 0, which means that there is no maintenance. The according AM parameters can be decoded into (3 2 0 2 1 0). For the OM parameter, after job 3, the mould 1 is in state 4, so the according parameter is 4, which means that after the job 3, the replace maintenance has to be conducted. After the job 3, the mould 2 is in state 3, the section [0.1 0.9] is decoded in 4 subsections [0.1 0.3), [0.3 0.5), [0.5 0.7), [0.7 0.9]. Since 0.3 is in the second section, it is decoded into 1. With the same method, the OM parameters can be decoded into (4 1 0 3 0 2). The Gantt charts of the example in Figure 4-3c are shown in Figure 4-4. In Figure 4-4, the job number is represented by the number in the box; the processing time for each job is represented by the number in the bracket; the maintenance is represented by the letter M.



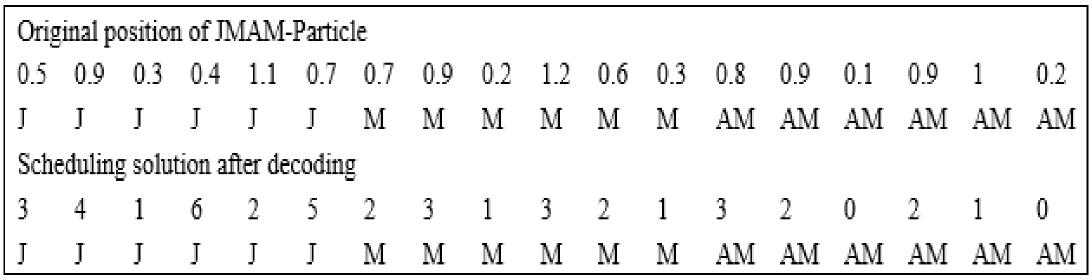
Original position of JM-Particle

0.5	0.9	0.3	0.4	1.1	0.7	0.7	0.9	0.2	1.2	0.6	0.3
J	J	J	J	J	J	M	M	M	M	M	M

Scheduling solution after decoding

3	4	1	6	2	5	2	3	1	3	2	1
J	J	J	J	J	J	M	M	M	M	M	M

Figure 4-3a The encoding and decoding of the JM-Particle



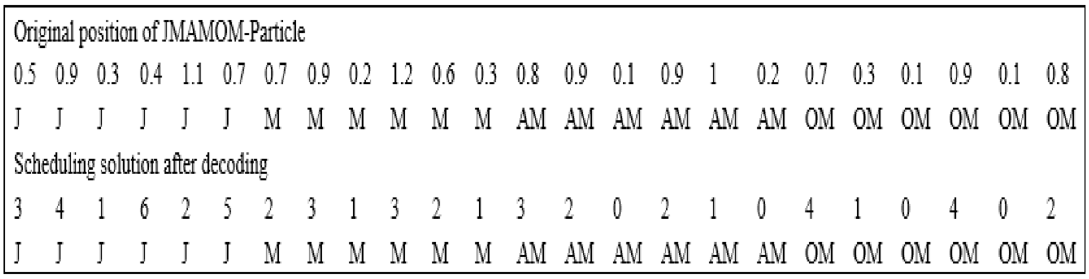
Original position of JMAM-Particle

0.5	0.9	0.3	0.4	1.1	0.7	0.7	0.9	0.2	1.2	0.6	0.3	0.8	0.9	0.1	0.9	1	0.2
J	J	J	J	J	J	M	M	M	M	M	M	AM	AM	AM	AM	AM	AM

Scheduling solution after decoding

3	4	1	6	2	5	2	3	1	3	2	1	3	2	0	2	1	0
J	J	J	J	J	J	M	M	M	M	M	M	AM	AM	AM	AM	AM	AM

Figure 4-3b The encoding and decoding of the JMAM-Particle



Original position of JMAMOM-Particle

0.5	0.9	0.3	0.4	1.1	0.7	0.7	0.9	0.2	1.2	0.6	0.3	0.8	0.9	0.1	0.9	1	0.2	0.7	0.3	0.1	0.9	0.1	0.8
J	J	J	J	J	J	M	M	M	M	M	M	AM	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM	OM

Scheduling solution after decoding

3	4	1	6	2	5	2	3	1	3	2	1	3	2	0	2	1	0	4	1	0	4	0	2
J	J	J	J	J	J	M	M	M	M	M	M	AM	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM	OM

Figure 4-3c The encoding and decoding of the JMAMOM-Particle
Figure4-3 The encoding and decoding of the particles

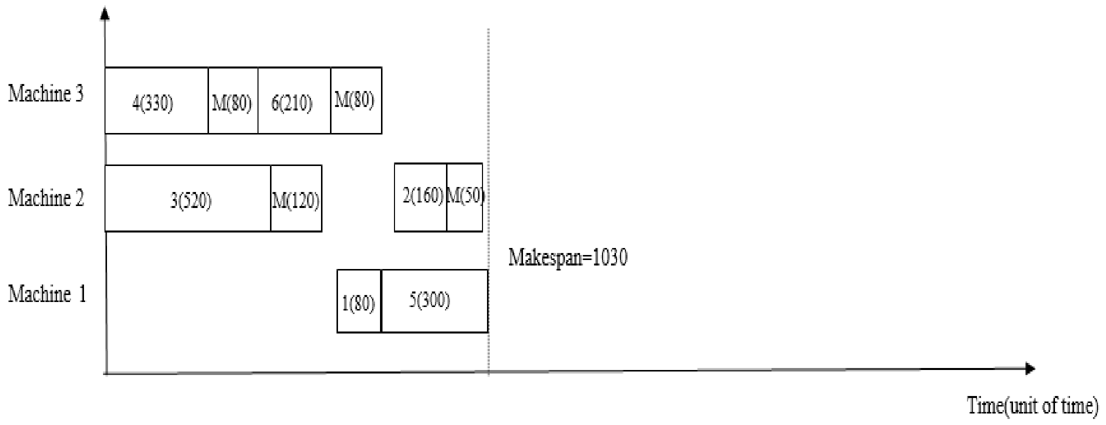


Figure 4-4a The Gantt Chart of the machine for the example in Figure 4-3c

(unit of time)

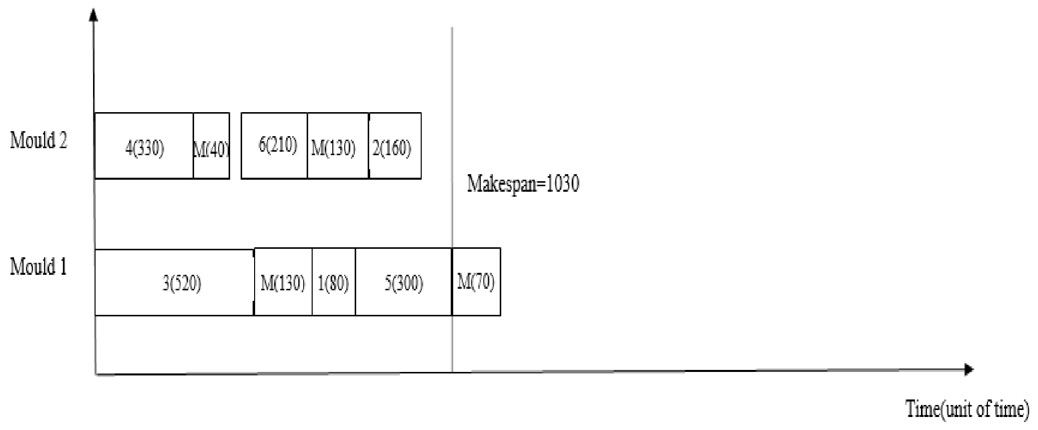


Figure 4-4b The Gantt Chart of the mould for the example in Figure 4-3c

Figure 4-4 The Gantt Chart of the JMAMOM-particle

4.2.2 The Three-Level PSO Algorithm

The Three-Level PSO contains three interrelated PSO algorithms. The flowchart of the three-level PSO is given in Figure 4-5.

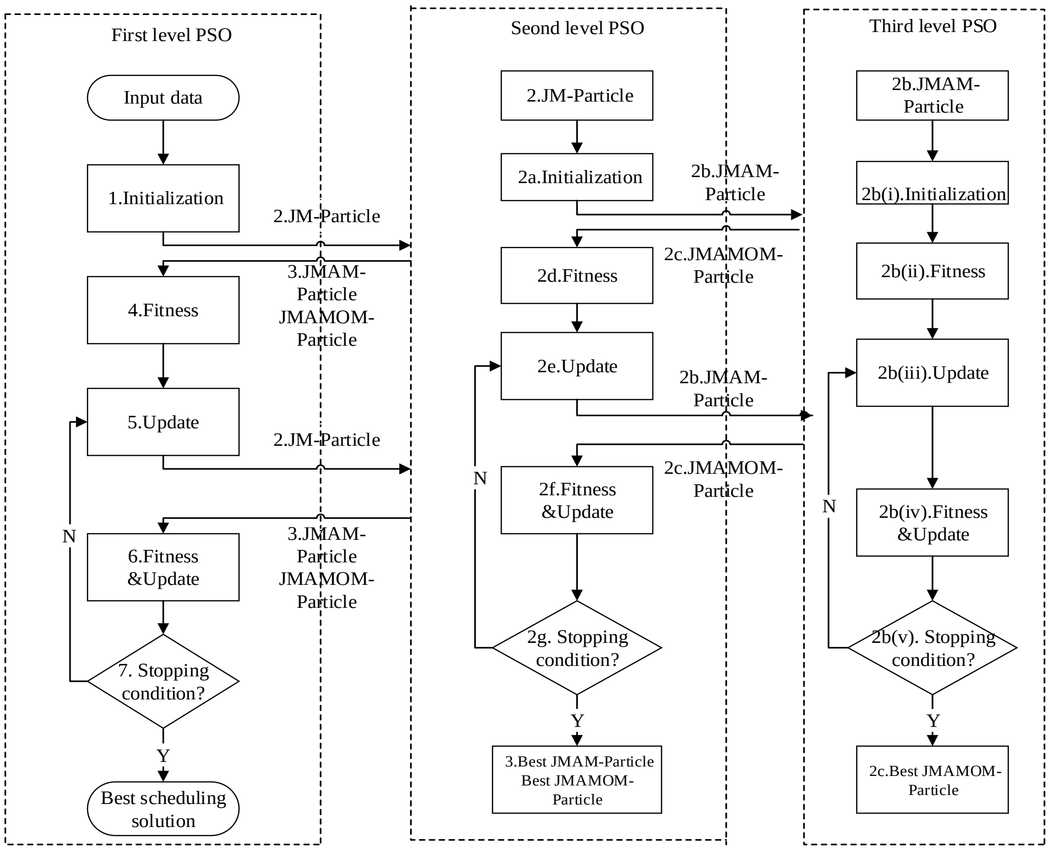


Figure 4-5 The flowchart of the Three-Level PSO

The details can be described as follows:

Start: the batch size of each job, the corresponding mould of each job, the available machines for each job and the unit operation time of each job are the input data for TLPSO.

1st level PSO

Step 1: Initialization. Initialize random positions and velocities in [0, 1] for a population of JM-Particles with dimension $2 \times P$ (P is the number of jobs).

Step 2: Pass each JM-Particle one by one to the second level PSO. In the second level PSO, **Step 2a- Step 2g** are conducted until the stopping condition is reached, and the best JMAM- Particle with its corresponding best JMAMOM-Particle are recorded.

2nd level PSO

Step2a: Initialization. For every JM-Particle, a population of JMAM-Particles with $3 \times P$ dimensions is initialized. The first $2 \times P$ dimensions of JMAM-Particles are the same as its related JM-particle. The last P dimensions of the JMAM-Particles are produced randomly in [0, 1]. A population of velocities is produced randomly, and the first $2 \times P$ dimensions of these velocities are zero, and the last P dimensions of the velocities are produced randomly in [0,1].

Step2b: Pass every JMAM-Particle one by one to the third level PSO. In the third level PSO, **Step 2b(i)-Step 2b(v)** are conducted until the stopping condition is reached, and the best JMAMOM-Particle is recorded.

3rd level PSO

Step2b(i): Initialization. For each JMAM-Particle, a population of JMAMOM-Particles with $4 \times P$ dimensions is initialized. The first $3 \times P$ dimensions of the JMAMOM-Particles are the same as its related JMAM-Particle. The last P dimensions of the JMAM-Particles are produced randomly in [0, 1]. A population of velocities is produced, and the first $3 \times P$ dimensions of these velocities are zero, and the last P dimensions of velocities are produced randomly in [0, 1].

Step2b(ii): Fitness. Using the encoding and decoding method in subsection 4.2.1 to transfer the continuous position vector of the JMAMOM-Particle into a suitable scheduling solution, and then measure the fitness (makespan) of each JMAMOM-Particle in the population and find the local best solution for all JMAMOM-Particles and the global best solution.

Step2b(iii): Update the velocity and position of each JMAMOM-particle. The velocity is updated based on Equation 4-1 and then the position is updated according to Equation 4-2. Finally, the inertia factor W is updated according to Equation 4-3, where $X_{ij}(r_3)$, $X_{ij}(r_3+1)$ and $V_{ij}(r_3)$, $V_{ij}(r_3+1)$ are the positions and velocities of the j^{th} dimension of the particle i at the r_3^{th} and $(r_3+1)^{th}$ iteration. Without loss of generality, we restrict the velocity in [-1 1]. $P_{ij}(r_3)$ is the best position of the j^{th} dimension of the particle i at r_3^{th} iteration. $P_{gj}(r_3)$ is the best position of the j^{th} dimension of all the particles at the r_3^{th} iteration, and W is the inertia weight. $W_{\max}$ is the maximum inertia weight and $W_{\min}$ is the minimum inertia weight. C_1 is the particle acceleration coefficient and C_2 is the population acceleration coefficient. R_1 and R_2 are random numbers between 0 and 1.

$$V_{ij}(r_3+1)=W\times V_{ij}(r_3)+C_1\times R_1\times(P_{ij}(r_3)-X_{ij}(r_3))+C_2\times R_2\times(P_{gj}(r_3)-X_{ij}(r_3)) \quad (4\text{-}1)$$

$$X_{ij}(r_3+1)=X_{ij}(r_3)+V_{ij}(r_3+1) \quad (4\text{-}2)$$

$$W=W_{\max}-\frac{W_{\max}-W_{\min}}{r_{3\max}}\times r_3 \quad (4\text{-}3)$$

Step2b(iv): Fitness. Again, use random key representation and the smallest position value (SPV) rule to transfer the continuous position vector of the particles into a suitable scheduling solution and measure the fitness (makespan) of each JMAMOM-Particle and update the optimal value of each JMAMOM-particle. $P1_{ibest}$ is the best value for the JMAMOM-particle i during the iteration process, and $P1_i^{r_3}$

is the current fitness of particle i. If $P1_i^{r_3} < P1_{ibest}$, then set $P1_{ibest} = P1_i^{r_3}$, update $P_{ij}(r_3)$; otherwise, retain $P1_{ibest}$ and $P_{ij}(r_3)$. Update the optimal value of the population. Define $g1_{best}$ as the best value of the particle population, and$g1_{best} = min(P1_{best}), (i = 1,2,\cdots P1_{enum3})$ (enum3 is the number of the popular size in the third level PSO). If $g1_{best}^{r_3} < g1_{best}$, set $g1_{best} = g1_{best}^{r_3}$, update $P_{gj}(r_3)$; otherwise, $g1_{best}$ and $P_{gj}(r_3)$ retain.

Step2b(v): Termination. Set $r_3 = r_3 + 1$, and check the condition that if $r_3 \le r_{3\max}$, then go to Step**2b(iii)**, start a new iteration; otherwise, terminate the third level PSO and calculate the minimum fitness for the JMAMOM-particles. The minimum fitness, corresponding to the best JMAMOM-particle and corresponding best scheduling solution is recorded.

Step 2c: The best JMAMOM-particle and minimum fitness for each JMAM-particle obtained from the third level PSO are delivered to the second level PSO.

Step 2d: Fitness. The fitness of each JMAM-particle is the fitness of the corresponding best JMAMOM-particle. Then, find the local best solution for all JMAM- particles and the global best solution.

Step 2e: Update the velocity and position of each JMAM-particle. The velocity is updated according to Equation 4-1 and then the position is updated according to Equation 4-2. Finally, the inertia factor W is updated according to Equation 4-3. The parameters have the same meaning and only need to change r_3 into r_2 to represent the second level iteration.

Step 2f: Fitness. To get the fitness of each new JMAM-particle, again pass every JMAM-Particle one by one to the third level PSO, repeat the process from Step2b-Step2c. Then, update the optimal value of each JMAM-particle. $P2_{ibest}$ is the best value for JMAM-particle i during the iteration process and $P2_i^{r_2}$ is the current fitness of particle i. If $P2_i^{r_2} < P2_{ibest}$, then set $P2_{ibest} = P2_i^{r_2}$, update $P_{ij}(r_2)$; otherwise, $P2_{ibest}$ and $P_{ij}(r_2)$ retain. Update the optimal value of the population. Define $g2_{best}$ as the best value of the particle population, and $g2_{best} = min(P2_{ibest}), (i = 1,2,\cdots, P2_{enum2})$(enum2 is the number of the popular size in the second level PSO). If $g2_{best}^{r_2} < g2_{best}$, set $g2_{best} = g2_{best}^{r_2}$, update $P_{gj}(r_2)$; otherwise, retain $g2_{best}$ and $P_{gj}(r_2)$.

Step 2g: Termination. Set $r_2 = r_2 + 1$, and check the condition that if $r_2 \le r_{2\max}$, , then go to Step 2e, and start a new iteration.; otherwise, terminate the second level PSO and calculate the minimum fitness for the JMAM-particles, which is the fitness of each JM-particle.

Step 3: The best JMAM-particle (with its corresponding best JMAMOM-particle obtained from the third level PSO) and the minimum fitness for each JM-particle obtained from the second level PSO is delivered to the first level PSO.

Step 4: Fitness. Measure the fitness of each JM-particle in the population and find the local best solution for all JM-particles and the global best solution.

Step 5: Update the velocity and position of each JM-particle. The velocity is updated according to Equation 4-1, and then the position is updated according to Equation 4-2. Finally, the inertia factor W is updated according to Equation 4-3. The parameters have the same meanings and only need to change r_3 into r_1 to represent the first level iteration.

Step 6: Fitness. To get the fitness of each new JM-particle, again pass each JM-Particle one by one to the second level PSO, repeat the process from **Step2 -Step3**. Then, update the optimal value of each JM-particle. P_{ibest} is the best value for JM-particle i during the iteration process and $P_i^{r_1}$ is the current fitness of JM-particle i. If $P_i^{r_1} < P_{ibest}$, then set $P_{ibest} = P_i^{r_1}$, update $P_{ij}(r_1)$; otherwise, retain P_{ibest} and $P_{ij}(r_1)$. Update the optimal value of the population. Define g_{best} as the best value of the particle population, and $g_{best} = min(P_{ibest}), (i = 1,2,\cdots, P_{enum1})$ (enum1 is the number of the popular size in the first level PSO). If $g_{best}^{r_1} < g_{best}$, set $g_{best} = g_{best}^{r_1}$, update $P_{gj}(r_1)$; otherwise, retain g_{best} and $P_{gj}(r_1)$.

Step 7: Termination. Set $r_1 = r_1 + 1$ and check the condition that if $r_1 \le r_{1\max}$, then go to **Step 5**, and start a new iteration.; otherwise, terminate the first level PSO and calculate the best value for each JM-particle i and the best scheduling solution of JMAMOM- particle for each JM-particle i is recorded.

End: the best scheduling solution (JMAMOM-Particle) for each JM-Particle is recorded.

4.3 Numerical Experiments

The main objective of the numerical experiments is to test the optimization performance of the strategy used in this research and the proposed TLPSO algorithm. Since the best algorithm parameters are different for different problems, the parameter tuning process is given for this proposed TLPSO firstly. And then, some other algorithms including the basic PSO, and other PSO variants (ABCSPSO, fkPSO, SPSO2011) proposed in the CEC'2013 competitions are chosen as the comparison algorithms to be compared with the proposed TLPSO algorithm. Numerical experiments are implemented in the Matlab environment on a personal computer with Intel (R) Core(TM) i7-6700 CPU 3.40GHz CPU.

4.3.1 Parameters Tuning

The Three-Level PSO contains two kinds of parameters: the key algorithm parameters (the maximum number of generations and the swarm size) and the parameters related to simple PSO (inertia weight W, particle acceleration coefficient C_1 and population acceleration coefficient C_2). At first, the parameters related to PSO are fixed. the parameters combination widely used and recommended by Eberhart and Shi (2000) is used. The maximum number of generations and the swarm size are decided. Then, the key algorithm parameters are fixed, and different parameters combinations related to PSO are tried and the best parameters combination is found.

In this integrated problem, the production scheduling problem is the master problem and the machine maintenance problem, as well as mould maintenance problem, has a similar influence on the makespan. Without loss of generality, we set the Max Iteration of the second level PSO as equivalent to the Max Iteration of the third level PSO and the Max Iteration of the first level PSO is the double value of Max Iteration of the second level PSO. The swarm size of the three PSOs in the different levels are the same. W is initially set as 0.9 and reduced linearly to 0.4. The values of C_1 and C_2 are equal to 2. This parameters combination related to PSO is the most often used in the literature. In the preliminary testing on key algorithm parameters, we test different combinations of parameters with swarm size={10, 15, 20} and Max Iteration of first level PSO={300, 800, 1200} for medium-sized instance ($40\times6\times10$) 10 times. There are 9 combinations in total. Table 4-2 shows the different parameters combinations, the corresponding average results and the time needed for each run. From Table 4-2, we can see that the result is better when the swarm size increase but better Max iteration does not always mean better on the result is not obvious. But as the swarm size and Max iteration increase, the running time increases. So the parameter combination{20, 800} is chosen as the final key algorithm parameter combination.

Table 4-2 Different key algorithm parameters combinations and its influence on results.

No	Swarm size	Max Iteration of first level PSO	Average Result	Min Result	Time(s)
1	10	300	2270	2231	380
2	10	800	2190	2156	960
3	10	1200	2070	2056	1432.5
4	15	300	2136	2121	1231
5	15	800	1958	1934	3199
6	15	1200	1967	1916	4967
7	20	300	1996.4	1940	2918
8	20	800	1909.6	1891	5404
9	20	1200	1993	1992	9104

To find a good combination for the parameters related to PSO, we set the swarm size as 20, the Max Iteration of the first level PSO as 800, the Max Iteration of the second level PSO as 400 and the Max Iteration of the third level PSO as 400, based on the preliminary testing. We compare the most widely used parameters combination (Eberhart and Shi 2000) with the other four PSO models, namely social only model (there is no cognitive component, $C_1 = 0$) (Kennedy and Eberhart 1995); cognition only model (there is no social acceleration, $C_2 = 0$) (Kennedy and Eberhart 1995); time-varying acceleration coefficient model (C_1 starts with a high value than C_2 and decreases while the social acceleration starts with a lower value and linearly increases) (Ratnaweera, Halgamuge, and Watson 2004), Clerc's constriction method (Eberhart and Shi 2000). The maintenance scheme of resources is shown in Table 4-1 and the medium-sized instance ($40 \times 6 \times 10$) is used, and we test each parameters combination 10 times. The outcomes are illustrated in Table 4-3. In Table 4-3, it could be seen that the result produced by the parameters combination recommended by Eberhart and Shi (2000) is the best. So the parameters combination recommended by Eberhart and Shi (2000) is used in this algorithm.

Table 4-3 The influence of parameters related to PSO on results.

No.	Parameters in the PSO model	Swarm size	Max Iteration of first level PSO	average	Std
1	W_{max}=0.9, W_{min}=0.4, C_1=2, C_2=2	20	800	1909.4	7.1
2	W_{max}=0.9, W_{min}=0.4, C_1=0, C_2=2	20	800	2141	9.4
3	W_{max}=0.9, W_{min}=0.4, C_1=2, C_2=0	20	800	2300	9
4	W_{max}=0.9, W_{min}=0.4, C_{1max}=2.5, C_{1min}=0.5, C_{2max}=2.5, C_{2min}=0.5	20	800	2233.9	102
5	W=0.729, C_1=1.49445, C_2=1.49445	20	800	2294	4.3

4.3.2 Algorithm Comparisons

To show the advantage of the proposed algorithm and strategy, five PSO variants are used as the comparison algorithms, including basic PSO (Kennedy and Eberhart 1995), ABCSPSO (El-Abd 2013), fk-PSO (Nepomuceno and Engelbrecht 2013), and SPSO2011 ((Zambrano-Bigiarini et al 2013). In these algorithms, the joint decision strategy proposed by Wong, Chan, and Chung (2012) is used. To fair comparison, the parameters in these algorithms are the same as the proposed Three-level PSO. Six instances are given, the size of these 6 instances are ($20\times2\times4$), ($30\times3\times5$), ($40\times6\times10$), ($60\times9\times15$), ($80\times10\times16$), ($100\times12\times20$) and the parameters of the instances are set as follows: the operation time of moulds is produced randomly between 30 and 55 units of time; the batch size of jobs is produced randomly between 2 and 6 units. Each algorithm is run 10 times to measure the deviation of solutions obtained. The results are shown in Table 4-4.

From Table 4-4, we can know that for these 6 instances, the TLPSO could always obtain better results than the other PSO variants for the minimum value, the maximum value and the average value. Furthermore, the standard deviation of the solutions by TLPSO is the minimum among all the comparison algorithms, which means that the TLPSO is more stable than other algorithms. However, the time for the TLPSO is longer than other algorithms. It is mainly because that for every solution in the first level PSO, it has to be sent to the second level PSO and the third level PSO, so it needs more time to conduct the deep exploration than other algorithms.

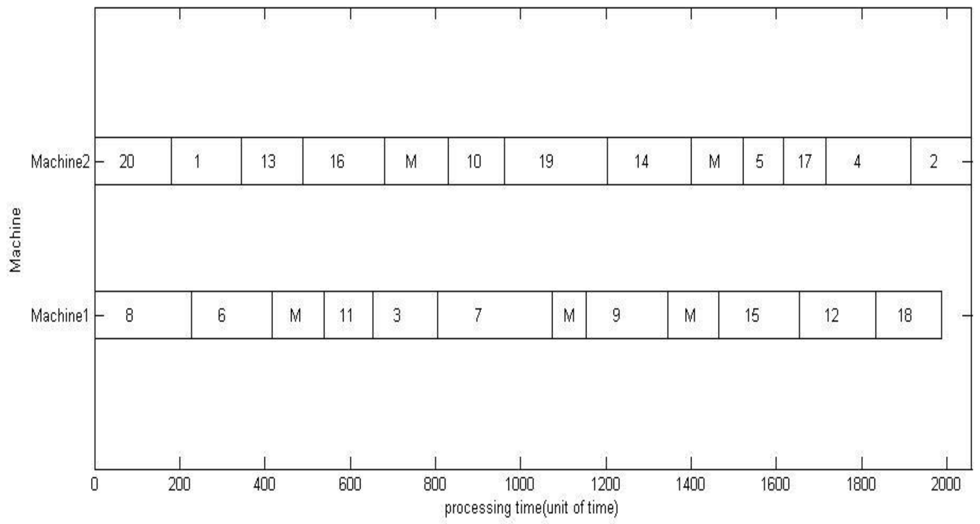


Figure 4-6a The machine schedule for the instance $(20\times2\times4)$

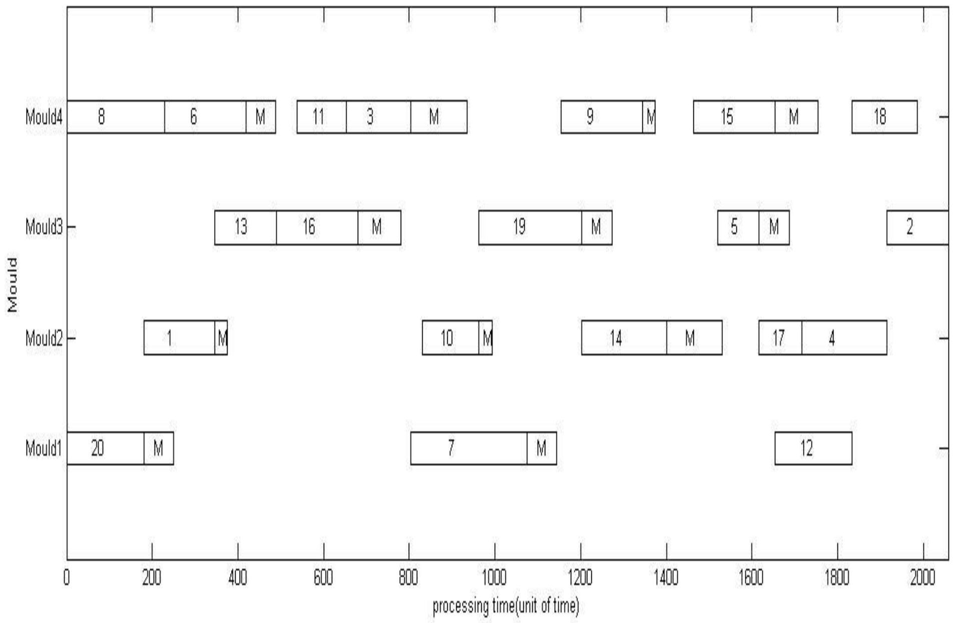


Figure 4-6b The mould schedule for the instance $(20\times2\times4)$

Figure 4-6 The Gantt chart for the instance $(20\times2\times4)$

Particularly, the Gantt chart for instance $(20\times2\times4)$ is shown in Figure 4-6. The Gantt chart for the instance $(20\times2\times4)$. From the Figure 4-6, we can know that the according job sequence(J) is (20 1 8 6 13 16 10 11 19 3 14 5 7 9 15 12 17 4 2 18); the machine sequence(M) is (2 2 1 1 2 2 2 1 2 1 2 2 1 1 1 1 2 2 2 1); the machine maintenance sequence (AM) is (0 0 0 3 0 4 0 0 0 0 3 0 2 3 0 0 0 0 0 0); the mould maintenance sequence(OM) is (2 1 0 2 0 3 1 0 2 4 4 2 2 1 3 0 0 0 0 0); the according mould sequence is (1 2 4 4 3 3 2 4 3 4 2 3 1 4 4 1 2 2 3 4). It can be seen that there are three kinds of maintenances for machine and there are four kinds of maintenances for the mould, which can be distinguished from the time of the maintenance.

Table 4-4 Comparison results with PSO variants.

No	Size	Algorithm	Min	Max	Avg	SD	Time (s)
1	$20\times2\times4$	TLPSO	2058	2088	2064.7	12	3631
		PSO	2032	2146	2081.4	37	9
		ABCSPSO	2030	2180	2095.4	41	177.9
		fk-PSO	2030	2208	2080	49	18
		SPSO2011	2032	2128	2074.2	28	9
2	$30\times3\times5$	TLPSO	1932	2001	1963.6	21	5698
		PSO	2164	2337	2235	53	14.9
		ABCSPSO	2062	2192	2122.3	44	292
		fk-PSO	2462	2094	2289.8	105	27
		SPSO2011	2148	2295	2219.6	61	14.5
3	$40\times6\times10$	TLPSO	1891	2174	1909.4	9	5040
		PSO	2134	2277	2204.4	56	52
		ABCSPSO	2082	2463	2241	125	509
		fk-PSO	2457	2619	2781	142	93
		SPSO2011	2380	2472	2425.6	37	52
4	$60\times9\times15$	TLPSO	1891	2255	1909.4	48	11169
		PSO	2202	2571	2351.7	129	28
		ABCSPSO	1980	2146	2077	50	566
		fk-PSO	2437	2814	2677	121	5.4
		SPSO2011	2309	2699	2508	107	28.7
5	$80\times10\times16$	TLPSO	2742	2852	2774	32	15230
		PSO	2769	3350	3091.4	158	38.6
		ABCSPSO	2777	3155	2881.9	119	770
		fk-PSO	3253	3785	3508.8	170	6.6
		SPSO2011	3107	3580	3371.3	156	37.7
6	$100\times12\times20$	TLPSO	3259	3483	3330.3	73	21020
		PSO	3318	4174	3665	247	25.5
		ABCSPSO	3264	3865	3499.5	260	856
		fk-PSO	3701	4676	4298.3	328	4.2
		SPSO2011	3946	4398	4237.9	141	25.3

4.4 Conclusions

This study proposes a new research problem that there are multiple maintenances in the integrated problem. In the new problem, the maintenance depends on the states of the resource and the maintenance has different influences on the resource. Except for the production scheduling, the kind of maintenance has to be decided when we decide the maintenance planning, which makes the problem more complex. This research uses the problem decomposing mechanism and a Three-Level Particle Swarm Optimization (TLPSO) algorithm is designed for this problem. Furthermore, a new encoding and decoding method is designed aiming at minimizing the overall makespan. The optimization reliability of TLPSO is tested in numerical experiments. The outcomes illustrate that the proposed TLPSO algorithm is effective in generating solutions with good quality in acceptable computation time. It is also shown that the proposed TLPSO algorithm surpasses PSO variants in the CEC'2013 competition when solving this integrated problem and the problem decomposition mechanism employed in TLPSO works well in this integrated problem.

However, as is shown in the results, this algorithm needs more time compared with other algorithms, especially when the scale of the problem increases. Further research may pay more attention to increasing the efficiency of the algorithm to reduce the computing cost. In addition, the problem will be modified, and more factors will be considered to make it more practical. Furthermore, the presented mechanism and algorithm will be used to resolve other integrated problems.

4.5 Summary

This chapter is an extension of chapter 3. Based on the problem of chapter 3, this chapter considers a more complex but more practical problem. Multiple maintenances are considered in the new problem and the algorithm and the strategy proposed in chapter 3 is extended to resolve the new problem. In subsection 4.1, the new problem is presented and the reason why the original problem is modified is given and the details of the multiple maintenances are presented. Then, in subsection 4.2, the algorithm to solve this problem is shown, particularly, a new encoding and decoding method for this problem is given. In subsection 4.3 numerical experiment is done and the proposed TLPSO algorithm is compared with other PSO variants to illustrate the advantages of the developed TLPSO algorithm. Finally, the conclusions are shown in subsection 4.4, which summarizes the content and points out the new research direction.

Chapter 5. A Multi-Objective Pigeon Inspired Optimization Algorithm for Fuzzy Production Scheduling Problem with Mould Maintenance

In Chapter 3 and Chapter 4, the integrated problems are determinate, and the algorithms to solve the problems using the Particle Swarm Optimization (PSO) algorithm as a basis. However, some uncertain factors exist in practice and the uncertainty could be represented by triangular fuzzy numbers. Moreover, some other swarm intelligence algorithms could be explored to address the related problem. There are 5 subsections in this chapter. The first subsection introduces the new problem including four parts related to the Fuzzy Production Scheduling Problem with Mould Maintenance (FPSP-MM). The second subsection presents the Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm designed for this problem, followed by the numerical experiments which verify the efficiency of the developed algorithm. Then, the results and discussion are presented, after which the summary is given.

5.1 Problem Description

The FPSP-MM problem can be described as follows: P jobs are allocated on Q injection machines and N injection moulds. Each problem is denoted as $P \times Q \times N$. At time zero, all jobs are well prepared, and all the machines and moulds are accessible. Each job must use the mould which is given in advance. Each job can choose the machines it uses, but not all the machines are available for all jobs. Different jobs may be performed by the same mould. Each job cannot be performed by more than one machine in a given time slot, and each machine cannot deal with more than one job in a given time slot. Each mould cannot carry out more than one job at a given time slot. The batch size and the unit fuzzy operation time of each job are given. The total operation time of a job is the product of the unit fuzzy operation time and the batch size. The batch size of each job cannot be split and there is no interruption during the production process of a job. The unit fuzzy operation time of a job depends on the mould it uses. The unit fuzzy operation time is represented by a triangular fuzzy number. The maintenance time of the resources depends on the time that the maintenance begins. The maintenance time is shorter when the maintenance is conducted earlier. The maintenance time of a resource depends on the longest accumulated working time of a resource (the accumulated working time of a resource is a triangular fuzzy number, the longest of which means the third number of the triangular fuzzy number). The maintenance time may be fuzzy or crisp. Only perfect maintenance is considered, which means that after the maintenance, the condition of the resource is as good as new. In this paper, the preventive maintenance is assumed to be able to prevent all the random breakdowns. Moreover, the set-up time and the quality issue are not considered in this research. The objective is to find good production scheduling and machine maintenance planning aiming at minimizing the makespan and maximizing the robustness. To extend the determinate single-objective problem to a fuzzy multi-objective problem, four issues need to be solved: the definition of the arithmetic operations on triangular fuzzy numbers, the fuzzy maintenance time, the robustness and the Pareto dominance relationship. These problems are given in Subsections 5.1.1–5.1.4.

5.1.1 Arithmetic Operations on Triangular Fuzzy Numbers

The unit fuzzy processing time is a triangular fuzzy number (TFN), denoted as $A=(a^1,a^2,a^3)$, a^1 is the best processing time, a^3 is the worst processing time and a^2 is the most possible processing time. When the original deterministic model is extended to a model with uncertainty, two difficulties need to be solved. First, the arithmetic operations of addition and maximum need to be given when deterministic numbers are changed into TFNs. Second, the definition of minimal makespan also needs to be given when the makespan is a triangular fuzzy number. According to Fortemps (1997), for two TFNs, $A=(a^1,a^2,a^3)$ and $B=(b^1,b^2,b^3)$. Their addition is defined as:

$$A+B=((a^1+b^1),(a^2+b^2),(a^3+b^3)) \tag{5-1}$$

The maximum operation is defined as:

$$\max(A,B)=(\max(a^1,b^1),\max(a^2,b^2),\max(a^3,b^3)) \tag{5-2}$$

To find the minimal makespan, three ranking criteria are given as follows:

$$C_1(A)=\frac{a^1+2\times a^2+a^3}{4} \tag{5-3}$$

$$C_2(A)=a^2 \tag{5-4}$$

$$C_3(A)=a^3-a^1 \tag{5-5}$$

To compare two TFNs, the values of C_1 are compared. If the values of C_1 are the same for two TFNs, then the values of C_2 are compared. If the values of C_1 and C_2 are the same for two TFNs, then the values of C_3 are compared.

5.1.2 Uncertain Maintenance Time

In the determinate model proposed by Wong et al. (2012), the accumulated working time of a resource is defined as the resource age (the idle time is not included). A piecewise linear function is used to describe the relationship between maintenance time and resource (machine or mould) age. In practice, a mould has a higher possibility of breakdown than a machine. So, the maximum age of the machine (MA) is longer than the maximum age of the mould (NA). Maintenance has to be conducted after completion of the current job once a resource reaches its maximum age. In the uncertainty model, since the processing time is a triangular fuzzy number, the resource age is also a triangular fuzzy number. The maintenance time is decided by the worst value in the triangular fuzzy number of the resource age. The age of the machine is represented by $A = (a^1, a^2, a^3)$. The age of the mould is represented by $B = (b^1, b^2, b^3)$. The relationship between the maintenance time and the resource age is shown in Table 5-1.

Table 5-1 Maintenance time based on machine/mould age

Machine age	Maintenance time	Mould age	Maintenance time
$0<a^3<=180$	(150,150,150)	$0<b^3<=120$	(150,150,150)
$180<a^3<=420$	$(94,94,94)+(a^1,a^2,a^3)$	$120<b^3<=280$	$150+(b^1, b^2, b^3)/2+4-60$
$420<a^3<=600$	$(160.160,160)+(a^1, a^2, a^3)/3$	$280<b^3<=400$	$(160,160,160)+(b^1, b^2, b^3)/2$
$600<a^3$	(720,720,720)	$400<b^3$	(720,720,720)

5.1.3 Objective Measure

There are two objectives in this problem. The first is the fuzzy makespan and the second is the robustness. According to Palacios et al (2017), the objective related to the fuzzy makespan C is defined as the expected value of the triangle fuzzy number

$$E(C)=\frac{C^1+2\times C^2+C^3}{4} \tag{5-6}$$

The ranking method based on the expected value is shown to be convenient and it is proven that the ranking result is similar to other ranking methods. It is obvious that the smaller the expected value, the better the objective value. The robustness is defined as

$$Rob(C)=\max\{(C^2-C^1),(C^3-C^2)\} \tag{5-7}$$

and it measures the maximum possible difference between the makespan of the real execution and the most likely estimated makespan. It is a priori measure and the smaller the value, the better the robustness. So the objectives of this problem are shown as follows:

$$\min \quad E(C) \tag{5-8}$$

$$\min \quad Rob(C) \tag{5-9}$$

5.1.4 Pareto Domination Relationship

For the multi-objective optimization problems with two or more objectives to be optimized,

$$\min \quad f(x)=(f_1(x),f_2(x),\cdots,f_m(x)) \tag{5-10}$$

Where $x=(x_1,x_2,\cdots,x_n)$ is an n-dimensional decision vector, $f(x)$ is an m-dimensional objective vector. The n-dimensional space, comprised of all the possible values of the decision vector x, is known as the decision space, and the m-dimensional space consisting of all the possible values of the objective vector $f(x)$ is the objective space. For multi-objective optimization problems, there may exist many different solutions and these solutions could be given a priority relationship by the Pareto dominance. For two solutions x and y that are feasible, the solution y is defined to be dominated by the solution x if $f_i(x) \leq f_i(y)$, $i = 1,2,\cdots,m$ and there is no less than one $j \in 1,2,\cdots,m$ so that $f_i(x) < f_i(y)$. A solution is defined to be non-dominated if no other solutions dominate it. The Pareto-optimal set (PS) is defined as the group consisted of the overall non-dominated solutions in the decision space. The Pareto Front (PF) is defined as the group of all the vectors in the objective space that corresponds to the PS.

5.2 Algorithm Design

In this subsection, the basic Pigeon Inspired Optimization (PIO) algorithm is introduced, followed by the encoding and decoding method of pigeons. Then, the Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm is presented.

5.2.1 Basic Pigeon Inspired Optimization Algorithm

PIO simulates the homing behavior of pigeons. There are two main operators in the algorithm.

(1) **Map and compass operator.** Through shaping the map in their brains by magnetoreception, pigeons are able to sense the earth’s magnetic field. Moreover,

pigeons adjust their directions based on the altitude of the sun, which has the same function as a compass. When pigeons fly to their destination, they rely less and less on the sun and magnetic particles. Each pigeon has a position X_i and a velocity V_i in a D-dimension search space. Both the positions and the velocities of the pigeons are updated in each iteration. The new position X_i and velocity V_i of the pigeon i at the t^{th} iteration can be calculated by the following equations:

$$V_i(t) = V_i(t-1) \times e^{-Rt} + rand \times (X_g - X_i(t-1)) \tag{5-11}$$

$$X_i(t) = X_i(t-1) + V_i(t) \tag{5-12}$$

Where $V_i(t-1)$ and $V_i(t)$ are the velocities of the pigeon i at $(t-1)^{th}$ and t^{th} iteration $X_i(t-1)$ and $X_i(t)$ are the positions of the pigeon i at $(t-1)^{th}$ and t^{th} iteration. R is the map and compass factor, $rand$ is a random number and X_g is the current global best position, which can be obtained by comparing all the positions among all the pigeons.

(2) **Landmark operator.** The pigeons depend on landmarks that they are near when the pigeons are close to their destination. They will fly directly to the destination if they are familiar with the landmarks. They will follow pigeons who are familiar with the landmarks if they are far from the destination and unfamiliar with the landmarks. In the landmark operator, half the number of pigeons is decreased in every generation which means pigeons that are far from the destination and unfamiliar with the landmarks will follow the pigeons that are familiar with the landmarks. Then, the pigeons close to their destination will fly to their destination quickly, which is

represented by $X_C(t)$ (the centre of some pigeons positions at the t^{th} iteration). The position updating rule for the pigeon i at the t^{th} iteration can be given by

$$N_p(t) = \frac{N_p(t-1)}{2} \tag{5-13}$$

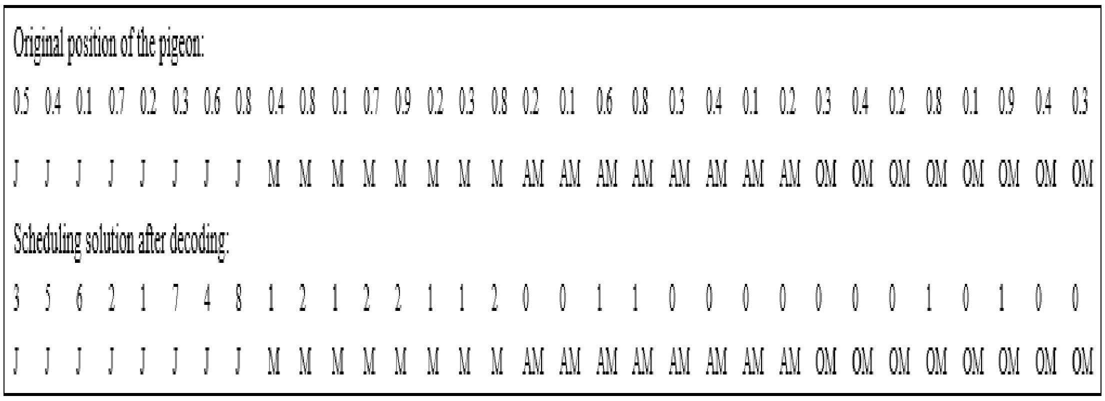
$$X_C(t) = \frac{\sum X_i(t) \times fitness(X_i(t))}{N_P \times \sum fitness(X_i(t))} \tag{5-14}$$

$$X_i(t) = X_i(t-1) + rand \times (X_C(t) - X_i(t-1)) \tag{5-15}$$

Where $fitness(X_i(t))$ is the quality of the pigeon i at the t^{th} iteration, and for minimum optimization problems, it is usually chosen as $fitness(X_i(t)) = \frac{1}{f_{\min}(X_i(t)) + \xi}$. For maximum optimization problems, it is usually chosen as $fitness(X_i(t)) = f_{\max}(X_i(t))$. For each individual pigeon, the optimal position of the Nc^{th} iteration can be denoted with X_P, and $X_P = min(X_i(1), X_i(2), \cdots X_i(N_c))$.

5.2.2 Encoding and Decoding of Pigeons

In MOPIO, each pigeon has information on the job sequence (J), the corresponding machine sequence (M), machine maintenance (AM) and information on the mould maintenance (OM). In the evolution process of MOPIO, the values of the positions for these pigeons always fluctuate in the space of a real number. Random key representation (Bean 1994) and the smallest position value (SPV) rule (Tasgetiren 2007) are applied to decode the positions of these pigeons into a suitable scheduling solution for this problem. After decoding, the values of the J parameters are integers between 1 and P (P is the number of jobs); the values of the M parameters are integers between 1 and Q (Q is the number of machines); The AM parameter is the maintenance decision on the machine, with value 0 or 1; The OM parameter is the maintenance decision on the mould, with value 0 or 1; If the relevant AM or OM is denoted as 1, the corresponding resource is maintained after finishing the job, otherwise they are not maintained. An example pigeon before and after decoding is shown in Figure 5-1. In this example, there are 8 jobs, 2 machines, and 2 moulds. Jobs 1, 2, 3, 4 can only be produced by mould 1 and jobs 5, 6, 7, 8 can only be produced by Mould 2. From Figure 5-1, it can be seen that the value of the job sequence (J) in the original position of the pigeon is (0.5 0.4 0.1 0.7 0.2 0.3 0.6 0.8), and it is transferred into (3 5 6 2 1 7 4 8). The sequence (0.5 0.4 0.1 0.7 0.2 0.3 0.6 0.8) is ranked according to the ascending order and given (0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8). Since the number 0.1 is in the third position of the original sequence, it is decoded into 3. Since the number 0.2 is in the fifth position of the original sequence, it is decoded into 5. Using this method, the original positions can be decoded into a suitable scheduling solution (3 5 6 2 1 7 4 8). The value of the corresponding machine sequence (M) in the original position of the pigeon is (0.4 0.8 0.1 0.7 0.9 0.2 0.3 0.8). The interval [0.1 0.9] (0.1 is the minimum and 0.9 is the maximum among all the numbers) is divided into 2 intervals, [0.1 0.5), [0.5 0.9] (there are 2 machines in this example). Since 0.4, 0.1, 0.2, and 0.3 are in the first interval, after decoding, the value in the relevant position is 1. Since 0.8, 0.7, and 0.9 are in the second interval after decoding, the value in the relevant position is 2. So, the corresponding machine sequence (M) can be transferred into (1 2 1 2 2 1 1 2). The value of the corresponding machine maintenance sequence (AM) in the original position of the pigeon is (0.2 0.1 0.6 0.8 0.3 0.4 0.1 0.2) and the interval [0.1 0.8] (0.1 is the minimum and 0.8 is the maximum among all the numbers) is divided into 2 intervals, [0.1 0.45) and [0.45 0.8]. Since 0.1, 0.2, 0.3 and 0.4 are in the interval [0.1 0.45), the value in the relevant position is decoded into 0. Since 0.6 and 0.8 are in the interval [0.45 0.8], the value in the relevant position is decoded into 1. So, the corresponding machine maintenance sequence (AM) is decoded into (0 0 1 1 0 0 0 0). A similar decoding method can be applied to mould maintenance (OM). After decoding, it is known that job 3 is distributed on machine 1 and machine 1 will not be maintained after job 3, and the injection mould on machine 1 will not be maintained either. Job 2 is allocated to machine 2, but machine 2 will be maintained after job 2 since the corresponding AM parameter is 1 and the injection mould on machine 2 will also be maintained because the corresponding OM parameter is 1.


Original position of the pigeon:

0.5 0.4 0.1 0.7 0.2 0.3 0.6 0.8 0.4 0.8 0.1 0.7 0.9 0.2 0.3 0.8 0.2 0.1 0.6 0.8 0.3 0.4 0.1 0.2 0.3 0.4 0.2 0.8 0.1 0.9 0.4 0.3

J J J J J J J J M M M M M M M M AM AM AM AM AM AM AM AM OM OM OM OM OM OM OM OM

Scheduling solution after decoding:

3 5 6 2 1 7 4 8 1 2 1 2 2 1 1 2 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0

J J J J J J J J M M M M M M M M AM AM AM AM AM AM AM AM OM OM OM OM OM OM OM OM

Figure 5-1 Encoding and decoding of the example pigeon

Furthermore, we give the fuzzy scheduling charts of the example pigeon (3 5 6 2 1 7 4 8 1 2 1 2 2 1 1 2 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0). The machine scheduling chart is shown in Figure 5-2 and the mould scheduling chart is shown in Figure 5-3. In this example, the unit fuzzy processing time of Mould 1 is (8 10 11). The corresponding batch sizes of jobs 1, 2, 3, 4 are 2, 3, 1, 2. The unit fuzzy processing time of Mould 2 is (9 11 13). The corresponding batch sizes of jobs 5, 6, 7, 8 are 3, 2, 2, 1. The fuzzy maintenance times on Machine 1 and Machine 2 are (2 5 7) and (3 4 6) respectively, and the fuzzy maintenance times on Mould 1 and Mould 2 are (2 3 6) and (4 6 8) respectively. The fuzzy number under the line is the start time of each job and the fuzzy number above the line is the end time of each job. The maintenance on the machine and the mould is represented in black. From Figure 5-2 and Figure 5-3, the fuzzy makespan is (86, 107, 122).

Moreover, to illustrate the influence of the fuzziness on the problem, we give the normal scheduling without fuzziness. For fair comparison, the example pigeon (3 5 6 2 1 7 4 8 1 2 1 2 2 1 1 2 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0) is used. The batch sizes of the jobs are not changed. We only modify the fuzzy properties of the processing time and maintenance time into the normal properties. The unit processing time of Mould 1 is 10, which is the most possible value in the triangular fuzzy number (8 10 11). The unit processing time of Mould 2 is 11, which represents the most possible value in the triangular fuzzy number (9 11 13). The maintenance times on Machine 1 and Machine 2 are 5 and 4, respectively. The maintenance times on Mould 1 and Mould 2 are 3 and 6, respectively. The Gantt charts of the example are shown in Figure 5-4 and Figure

5-5, and the makespan is 107. It represents the most possible value in the triangular fuzzy makespan (86, 107, 122).

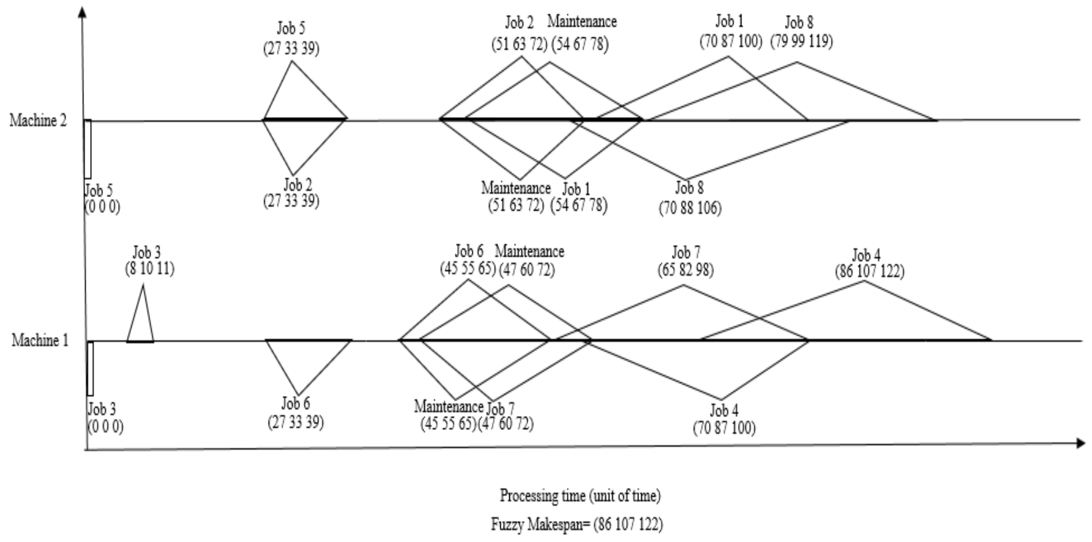


Figure 5-2 Fuzzy machine scheduling of the example pigeon

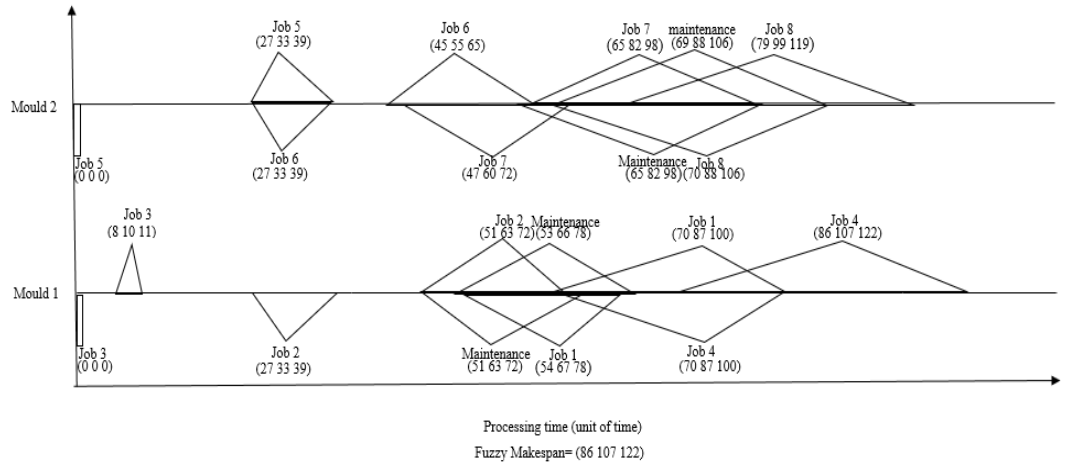


Figure 5-3 Fuzzy mould scheduling of the example pigeon.

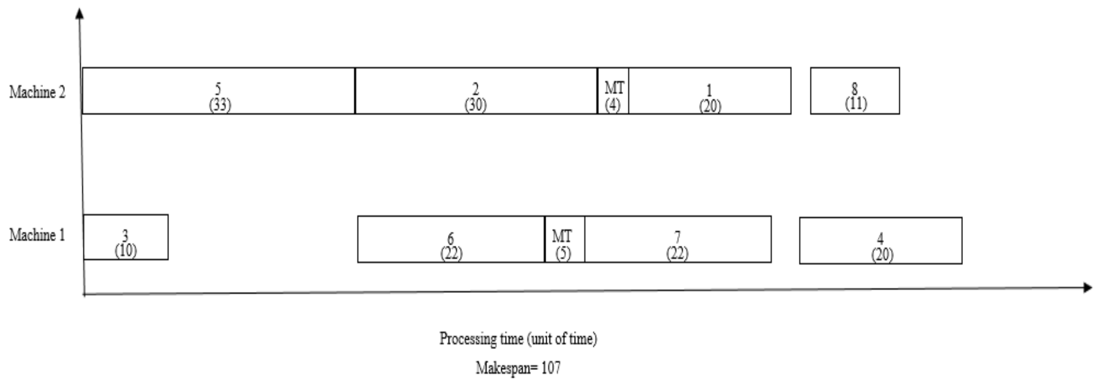


Figure 5-4 Machine scheduling of the example pigeon without fuzziness.

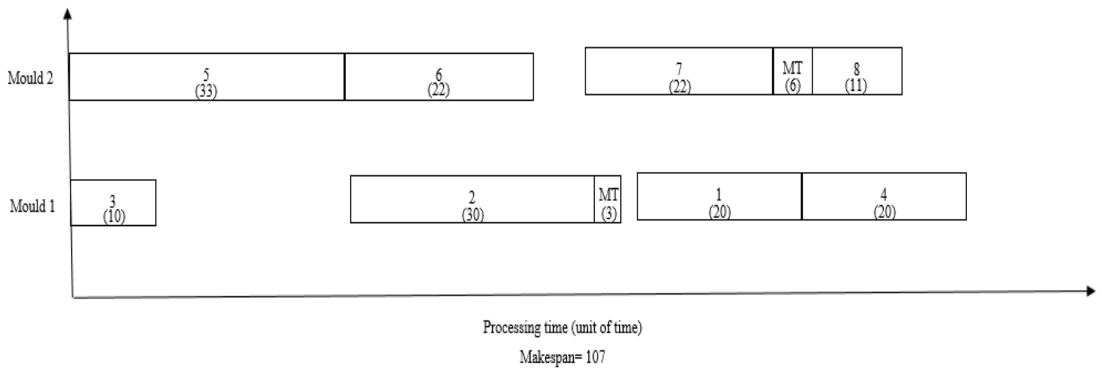


Figure 5-5 Mould scheduling of the example pigeon without the fuzziness.

From the comparative results, the solution is still applicable when the fuzzy problem is modified into a normal case and can illustrate that the rules to calculate the makespan for the fuzzy problem are appropriate. The solutions obtained by the proposed MOPIO algorithm can also be used for normal cases. However, when the fuzziness is considered in this model, more factors and objectives need to be considered to make it closer to reality.

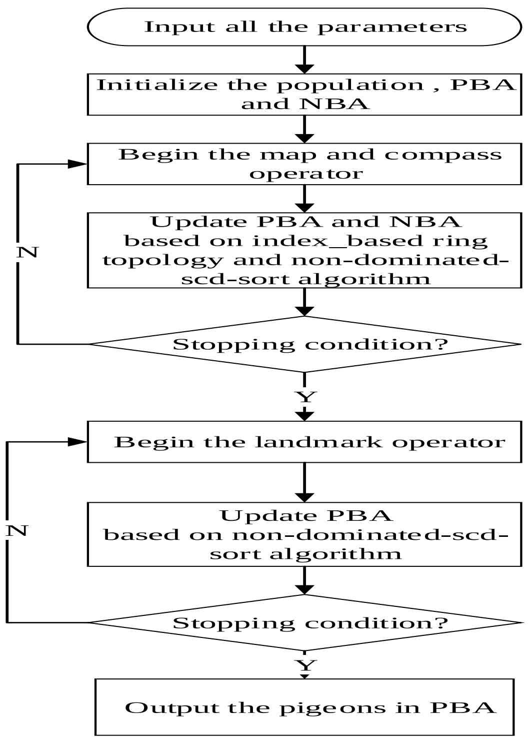

Input all the parameters
Initialize the population , PBA and NBA
Begin the map and compass operator
Update PBA and NBA based on index_based ring topology and non-dominated-scd-sort algorithm
Stopping condition?
N
Y
Begin the landmark operator
Update PBA based on non-dominated-scd-sort algorithm
Stopping condition?
N
Y
Output the pigeons in PBA

Figure 5-6 Flowchart of the MOPIO.

5.2.3 Multi-Objective Pigeon Inspired Optimization Algorithm

Inspired by the multi-objective particle swarm optimizer using ring topology proposed by Yue et al. (2018), a MOPIO algorithm is proposed using ring topology and a special non-dominated sorting method.

In the map and compass operation of MOPIO, we use the best pigeon in the neighbourhood of each pigeon instead of the global best pigeon in order to avoid the population convergence to a single point. So Equation 5-11 is modified into

$$V_i(t) = V_i(t-1)\times e^{-Rt} + rand \times (X_{nbest_i} - X_i(t-1)) \tag{5-16}$$

Where X_{nbest_i} is the best pigeon in the neighbourhood of the i^{th} pigeon. Other symbols have the same meanings as the symbols in Equation 5-11. Two archives are established: the Personal Best Archive (PBA) and Neighborhood Best Archive (NBA). The personal best pigeon and the neighbourhood best for each pigeon are chosen from the corresponding PBA and NBA. For the NBA, $NBA\{i\}$ denotes the best position within the i^{th} particle neighbourhood. Each neighbourhood includes three particles, the i^{th} particle and its immediate neighbours on its right and left. Moreover, an index-based ring topology is used to build the neighbourhood, and pigeons in different neighbourhoods cannot interact with each other directly. The use of the NBA promotes the formation of multiple niches by restricting information transmission through the population. Furthermore, a special sorting scheme, named the non-dominated-scd-sort algorithm, is used to rank the pigeons.

In the landmark operator, the number of pigeons is chosen as the numbers of pigeons in PBA, and the fitness is defined as $fitness(X_i(t)) = \frac{1}{obj1_{\min}(X_i(t)) + obj2_{\min}(X_i(t)) + \xi}$, where $obj1_{\min}(X_i(t)) + obj2_{\min}(X_i(t))$ is the sum of two objective values. The positions of all pigeons are updated according to Equation 5-13 to Equation 5-15, and pigeons are ranked based on the non-dominated-scd-sort algorithm. When all the iterations end, the best pigeons from the PBA represent the final optimization solutions.

The non-dominated-scd-sort algorithm was proposed by Yue et al. (2018) and involves two steps. In the first step, the pigeons are sorted according to the non-dominated sorting scheme (Deb et al. 2002). In the second step, the special crowding distances of the non-dominated pigeons are calculated, which also involves two steps. In the first step, the Crowding Distance (CD), for each pigeon in the decision space and the corresponding image in the objective space are calculated. In the second step, the CDs from the first step are used to assign a Special Crowding Distance (SCD) for each pigeon. The SCD concept involves a max or min selection step that involves crowding metrics from the decision and objective spaces. The diversity in the solution and objective spaces are promoted simultaneously by this methodology. Finally, the non-dominated solutions are ranked in descending order according to their special crowding distances. After sorting, the first particle is the non-dominated solution with the largest SCD.

The flowchart of the proposed MOPIO is shown in Figure 5-6 and the details of each step are given as follows.

Step 1. Input the parameters of the fuzzy production problem with mould maintenance, including the numbers of jobs, machines, and moulds, the batch size of each job, the corresponding mould of each job, the available machines for each job, the unit fuzzy operation time of each job.

Step 2. Initialize the parameters of MOPIO, including the dimensions of the solution space, the size of the population, NUM, map and compass factor R, the number of iteration $Nc_1\max$ and the number of iteration $Nc_2\max$ for two operators.

Step 3. Each pigeon is randomly allocated a position and a velocity. Calculate the objective values of all the pigeons. $POP_i(t)$ represents the pigeon i at the t^{th} iteration. Initialize the number of elements in the PBA and the NBA. $PBA\{i\}$ saves the best position for pigeon i and $NBA\{i\}$ saves the best position in the neighborhood of the pigeon i.

Step 4. Begin the map and compass operator. Set $t_1 = 1$. When the iteration t_1 is smaller than $Nc_1\max$, for all the NUM pigeons, sort the pigeons in $PBA\{i\}$ and $NBA\{i\}$ based on the non-dominated-scd-sort algorithm. Select the first pigeon of the sorted $NBA\{i\}$ as the $nbest_i$. Update the $POP_i(t_1)$ according to (5-16) and (5-12). Calculate the objective values and the SCD of $POP_i(t_1+1)$.

Step 5. Update PBA. For all the NUM pigeons, put $POP_i(t_1+1)$ into $PBA\{i\}$ and remove all the pigeons dominated by $POP_i(t_1+1)$.

Step 6. Update NBA. Use the index-based ring topology to decide the neighborhood of pigeon i, which includes three pigeon groups, the $PBA\{i\}$, $PBA\{i-1\}$ and $PBA\{i+1\}$. Particularly, if $i=1$, the neighborhood of the pigeon i is defined as $PBA\{NUM\}$, $PBA\{1\}$, $PBA\{2\}$; if $i=NUM$, the neighborhood is defined as $PBA\{NUM-1\}$, $PBA\{NUM\}$ and $PBA\{1\}$. Then obtain the non-dominated pigeons based on the non-dominated-scd-sort algorithm in the neighborhood and put them into the $NBA\{i\}$.

Step 7. Set $t_1 = t_1 + 1$. If the iteration t_1 is smaller than $Nc_1 \max$, return to step 4. If the iteration t_1 is bigger than $Nc_1 \max$, go to step 8.

Step 8. Begin with the landmark operator. Set $t_2 = 1$. When the iteration t_2 is smaller than the $Nc_2 \max$, the landmark operator is activated. The size $N_P(1)$ is chosen as the number of pigeons in the archive PBA at the last iteration of the map and compass operation. The size $N_p(t_2)$ is chosen as the number of pigeons in the archive PBA at iteration t_2. Then the size N_p is decreased by half in every iteration according to Equation 5-13, which means that half of the pigeons will follow the other half of the pigeons that are familiar with the landmark. The fitness is set as $fitness(X_i(t)) = \dfrac{1}{obj1_{\min}(X_i(t)) + obj2_{\min}(X_i(t)) + \xi}$, where $obj1_{\min}(X_i(t)) + obj2_{\min}(X_i(t))$ is the sum of two objective values. The centre of the pigeons $X_C(t)$ will be calculated according to Equation 5-14, and the positions of the pigeons will be updated according to Equation 5-15.

Step 9. Update PBA. Calculate the objective values and the SCD of all the pigeons $POP_i(t_2+1)$. Put the new pigeons $POP_i(t_2+1)$ in the PBA and delete the pigeons dominated by $POP_i(t_2+1)$ based on the non-dominated-scd-sort algorithm.

Step 10. Set $t_2 = t_2 + 1$. If the iteration t_2 is smaller than $Nc_2 \max$, return to step 8. If the iteration t_2 is bigger than $Nc_2 \max$, these pigeons in PBA are taken as the final optimization results.

5.3 Numerical Experiments

The main objective of the numerical experiments is to test the optimization performance of the proposed MOPIO algorithm. Since the problem is an extension of the problem proposed by Wong et al. (2012), the benchmark datasets used in this paper are generated by randomly fuzzifying the crisp benchmarks from Wong et al. (2012) To fuzzy a crisp benchmark dataset and generate a triangular fuzzy processing time $P_{ij} = (P_{ij}^1, P_{ij}^2, P_{ij}^3)$, according to Lei (2011), the most plausible value P_{ij}^2 of the fuzzy processing time is equal to the value of the crisp processing time P_{ij} in the crisp datasets. The values of P_{ij}^1 and P_{ij}^2 are randomly generated from [0.85 P_{ij}, 0.95 P_{ij}] and [1.1 P_{ij}, 1.19 P_{ij}]. The sizes of these three problems are ($30 \times 3 \times 5$), ($40 \times 6 \times 10$) and ($60 \times 9 \times 15$). The quality of the solutions produced by the proposed MOPIO algorithm is verified by comparing the results obtained by adapted NSGA-II (Deb et al. 2002) and MOPSO (Coello et al. 2004). Furthermore, three more randomly generated datasets of sizes ($20 \times 2 \times 4$), ($35 \times 4 \times 6$) and ($65 \times 8 \times 10$) are used as benchmarks to further illustrate the performance of the proposed MOPIO algorithm. Numerical experiments are implemented in the MATLAB environment on a personal computer with Intel(R) Core (TM) i7-6700 CPU 3.40 GHz CPU.

5.3.1 Performance Indicator

To evaluate an algorithm, quantitative analysis is as important as qualitative analysis. Except listing the non-dominated solutions found over a certain number of runs by different algorithms, this study uses two performance metrics to measure the performance of different algorithms: the HV (Zitzler 2003) and the CR which is a modification from the cover rate in the research of Yue et al. (2018).

The HV metric is used to measure the volume of hypercube enclosed by PF A and a reference vector $r_{ref} = (r_1, r_2, \cdots, r_n)$ with a larger value representing better performance is calculated as follows:

$$HV(A) = \bigcup_{a \in A} vol(a) \tag{5-17}$$

Where $vol(a)$ is the hypercube volume enclosed by the solution a in the PF A and the reference vector $r_{ref} = (r_1, r_2, \cdots, r_n)$. The bigger the value, the better the algorithm.

The CR represents the overlap ratio between different PFs obtained by different algorithms. The definition of the CR ($CR(A,B)$) is as follows:

$$CR(A,B)=\left(\prod_{l=1}^{n}\delta_l\right)^{1/2n} \tag{5-18}$$

$$\delta_l=\begin{cases} 1 & F_l^{\max}=F_l^{\min} \\ 0 & f_l^{\min}\geq F_l^{\max} \,\|\, f_l^{\max}\leq F_l^{\min} \\ \left(\dfrac{\min(f_l^{\max},F_l^{\max})-\max(f_l^{\min},F_l^{\min})}{F_l^{\max}-F_l^{\min}}\right)^2 & otherwise \end{cases} \tag{5-19}$$

Where n is the dimensionality of the objective space; $f_l^{\max}$ and $f_l^{\min}$ are respectively the maximum and minimum of the l^{th} objective value obtained by algorithm A. $F_l^{\max}$ and $F_l^{\min}$ are the maximum and minimum of the l^{th} objective value obtained by algorithm B. If $CR(A,B)$ is bigger than $CR(B,A)$, it means that the scope of the PF obtained by the algorithm A is larger than the scope of the PF obtained by algorithm B, which means that algorithm A is better than the algorithm B in terms of the CR.

5.3.2 Parameters Tuning

Since the parameters of the algorithm have a considerable influence on the results of the solution, we test different combinations of parameters on a medium-size dataset ($40\times6\times10$) to decide the final parameters for this algorithm. There are three main parameters in the proposed MOPIO: swarm size, the max iteration (the iteration of each operation is half of the overall iteration) and the map and compass factor R. For each parameter, three levels are selected to find an appropriate parameter combination. The details of the parameter sets are shown in Table 5-2. The Taguchi method is used to reduce the number of experiments. Each parameter combination is run ten times. The orthogonal arrays based on Taguchi methods, the average values and the standard deviation of the HV for each parameter combination are shown in Table 5-3. Based on Li et al. (2014), the impact trend of the different parameters with different levels is shown in Figure 5-7, where the performance of the proposed algorithm is better when the swarm size is at level 2, the max iteration is at level 3 and the R is at level 1. The larger the max iteration, the better the result, however, a larger swarm size does not mean better results. The smaller the R, the better the result. The swarm size is related to the search ability, but it does not mean that a larger size is the better. If the swarm size is too large, there may be a higher repetitive rate of the swarm which decreases the search efficiency. The max iteration can improve the performance under some given conditions, and it may increase the search time. The increment of the max iteration is not useful when the solution reaches its extremum. The map and compass factor R decides the influence of the previous velocity on the present velocity. The smaller the R, the bigger the influence of the previous velocity on the present velocity. Based on the above analysis, the parameter combination {swarm size = 50, iteration = 400 and R = 0.01} is chosen as the parameter used for the following tests.

Table 5-2 Levels of different parameters

Level	Swarm size	Max iteration of each operation	R
1	20	100	0.01
2	50	200	0.2
3	80	400	0.4

5.3.3 Comparison with Other Multi-objective Algorithms

To test the performance of the proposed MOPIO, this research compares it with the other two well-known multi-objective algorithms: adapted NSGA-II (Deb et al. 2002) and MOPSO (Coello et al. 2004). For unbiased comparison, the swarm size is set as 50, and the max iteration is set as 400 for these three algorithms. The parameters related to the MOPSO and NSGA-II are the same as the parameters in the original literature. The R is set as 0.01 for MOPIO. The reference vector for the calculation of HV is set as (5200, 400), and each algorithm is run ten times. For the fuzzed benchmark datasets, the PFs of three algorithms are shown in Figure 5-8–Figure 5-10. The comparison results of the performance indicators (the HV and the CR) are shown in Table 5-4 and Table 5-5. Avg(HV) is the average value of the HV for the 10 runs, SD(HV) is the standard deviation of the HV for the 10 runs and Avg(CR) is the average value of the CR for the 10 runs. SD(CR) is the standard deviation of the CR for the 10 runs.

Table 5-3 Different parameters combinations and its influence on results

No.	Swarm size	Max iteration of each operation	R	Avg(HV)	Sd(HV)
1	1	1	1	184570	25677
2	1	2	2	183710	52467
3	1	3	3	224810	95112
4	2	1	2	244600	34749
5	2	2	3	240460	54875
6	2	3	1	304520	34394
7	3	1	3	237050	42451
8	3	2	1	264770	24049
9	3	3	2	262280	15366

Furthermore, three more instances are produced randomly to better illustrate the performance of the proposed MOPIO algorithm. The size of these three problems are $(20\times 2\times 4)$, $(35\times 4\times 6)$ and $(65\times 8\times 10)$. The most plausible value P_{ij}^{2} of the fuzzy processing time is produced randomly between 30 and 55 units of time. The values of P_{ij}^{1} and P_{ij}^{3} are randomly generated from $[0.85P_{ij}^{2}, 0.95P_{ij}^{2}]$ and $[1.1P_{ij}^{2}, 1.19P_{ij}^{2}]$, and the batch size of the jobs is produced randomly between 2 and 6 units. The comparison results of the performance indicators are shown in Table 5-6 and Table 5-7.

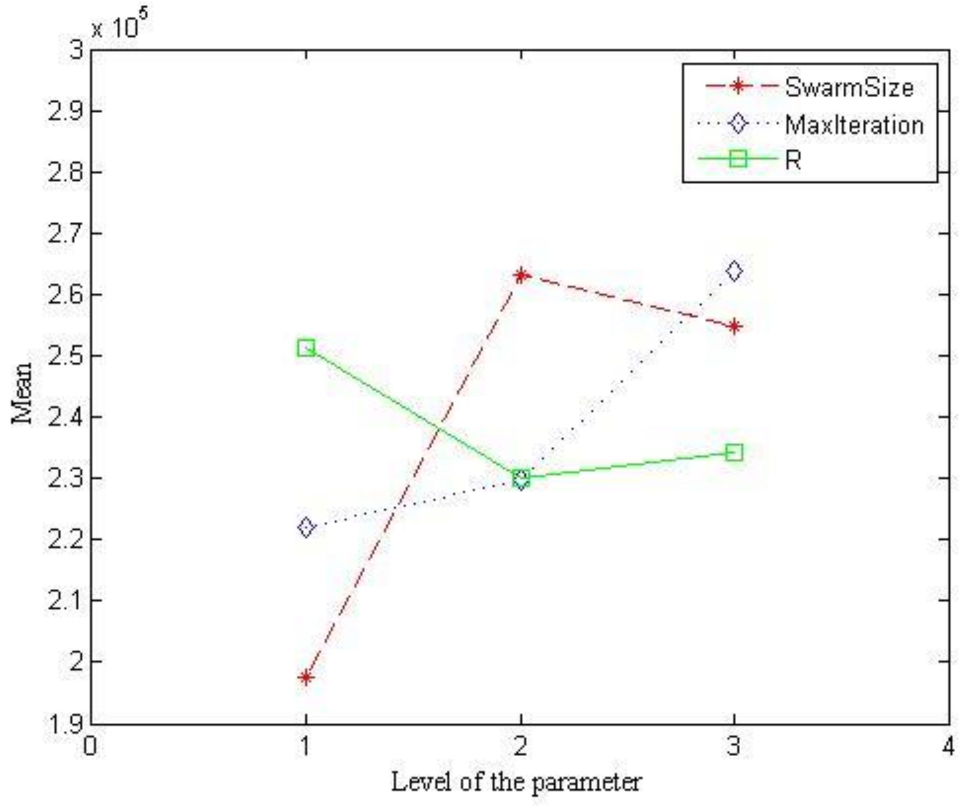


Figure 5-7 Factor level trend

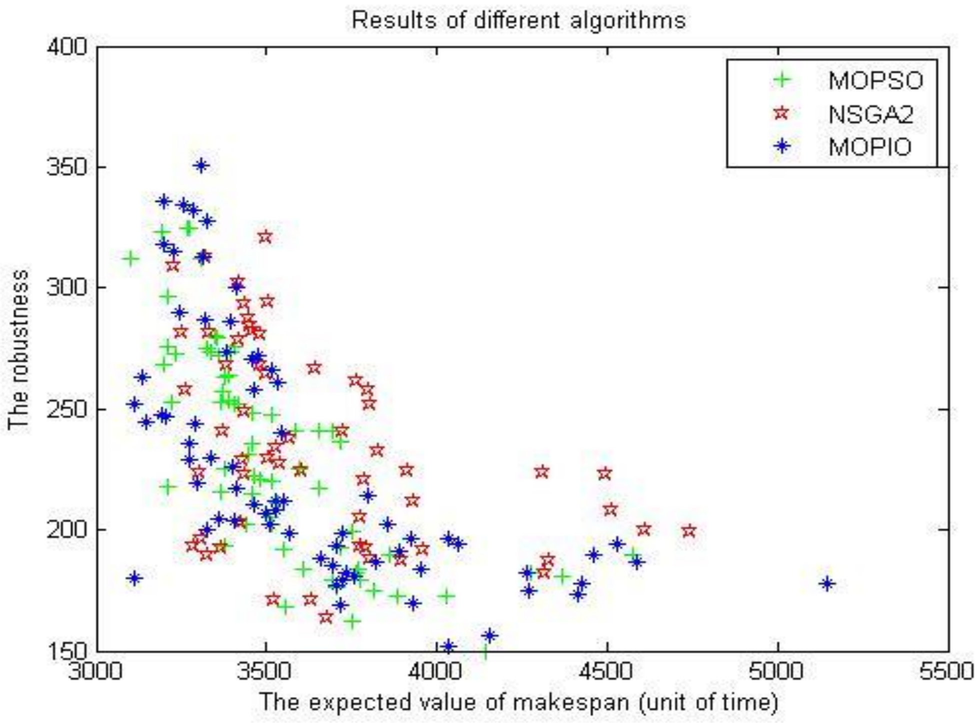


Figure 5-8 Pareto fronts of the fuzzed benchmark (30 × 3 × 5).

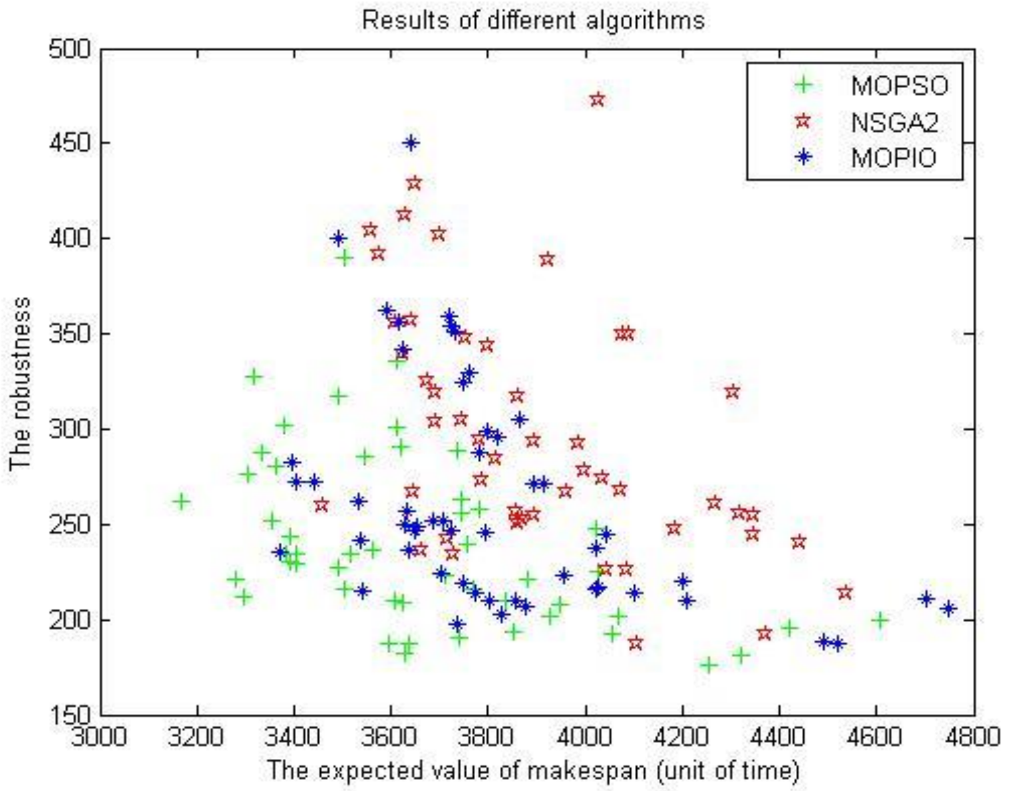


Figure 5-9 Pareto fronts of the fuzzed benchmark (40 × 6 × 10).

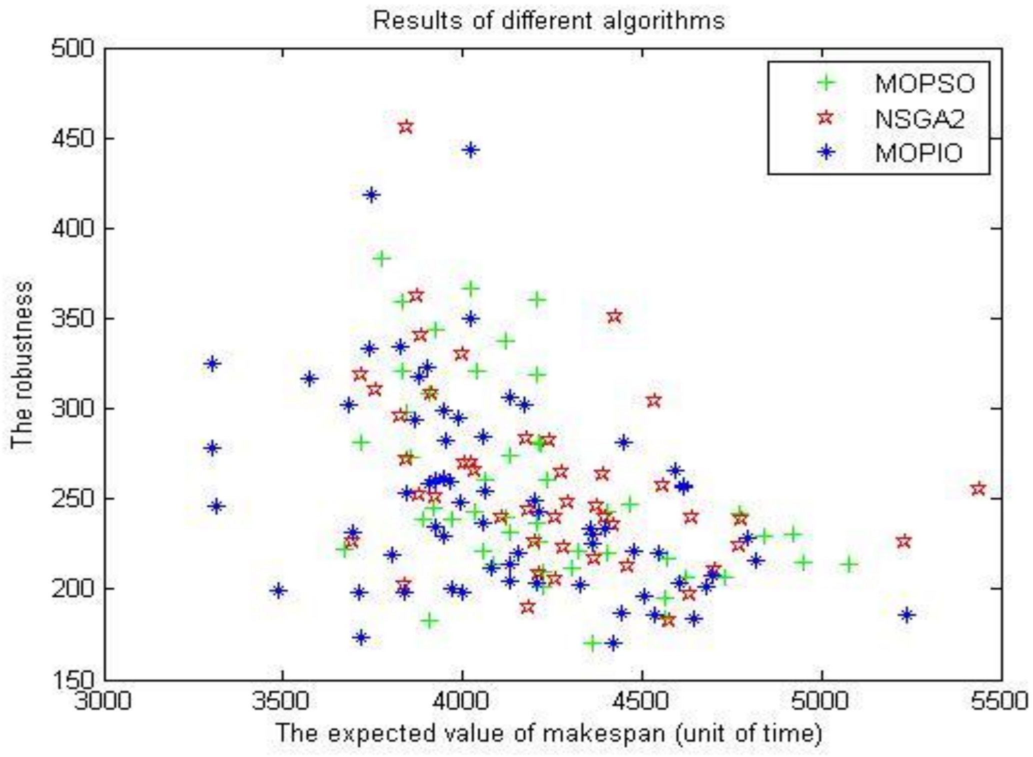


Figure 5-10 Pareto fronts of the fuzzed benchmark (60 × 9 × 15).

In the proposed MOPIO algorithm, the neighborhood best is used instead of the global best in the original PIO algorithm when the velocity is updated. To explain the influence of this modification, we compare the proposed MOPIO algorithm which uses the neighbourhood best with another version of MOPIO named MOPIO-GBA which uses the global best. In the MOPIO-GBA, the Global Best Archive (GBA) is built. In each iteration of the map and compass operation, we add the first pigeon in every $PBA\{i\}$ based on the ranking by the non-dominated-scd-sort algorithm into the GBA. Then, all the pigeons in the GBA are ranked based on the non-dominated-scd-sort algorithm and the global best is selected as the first pigeon in the GBA. Except for the velocity updating formula, other steps in the MOPIO-GBA are the same as the proposed MOPIO. The above six instances are used to test the performance of MOPIO-GBA and MOPIO. The comparison results of the performance indicators are shown in Table 5-8 and Table 5-9.

Moreover, a solution to the instance ($20 \times 2 \times 4$) is shown in Figure 5-11 and Figure 5-12. For the instance ($20 \times 2 \times 4$), the batch sizes of these 20 jobs are (5 3 4 6 2 5 6 6 5 4 3 4 3 6 5 4 3 4 5 4) and the moulds used by these jobs are (2 3 4 2 3 4 1 4 4 2 4 1 3 2 4 3 2 4 3 1). The unit fuzzy processing times by different moulds are shown in Table 5-10. The maintenance time is based on the relationship in Table 5-1. The numbers in the triangles are the job numbers, and the triangles under the line represent the starting time of each job. The triangles above the line represent the ending time of each job. MT means maintenance on the machine or the mould. After decoding, the solution is (4 17 16 11 14 10 18 9 5 20 7 3 6 2 13 12 15 19 8 1 2 2 2 1 2 2 1 1 2 1 1 1 2 2 1 1 2 1 2 1 1 1 0 1 0 0 1 0 1 1 0 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 1 1 1).

5.4 Results and Discussion

From Figure 5-8, the PF of the dataset ($30\times3\times5$) by MOPIO is in the left bottom of the coordinate system compared with the PFs by MOPSO and NSGA-II. From Table 5-4, the dataset ($30\times3\times5$), the value of Avg(HV) by MOPIO, is bigger than the values of Avg(HV) by MOPSO and NSGA-II. Furthermore, the boundary of the PF by MOPIO is larger than the PFs by MOPSO and NSGA-II. From Table 5-5, the value of the Avg(CR(MOPIO, NSGA-II)) is bigger than the value of the Avg(CR(NSGA-II, MOPIO)) and the value of the Avg(CR(MOPIO, MOPSO)) is bigger than the value of the Avg(CR(MOPSO, MOPIO)). It can be concluded that for the dataset ($30\times3\times5$), the quality of the PF by MOPIO is better than the PFs by MOPSO and NSGA-II. For the dataset ($60\times9\times15$), by comparing the location of PFs in the Figure 5-10 and the values for the dataset ($60\times9\times15$) in Table 5-4 and Table 5-5, it can be concluded that for the dataset ($60\times9\times15$), the quality of the PF by MOPIO is better than the PFs by MOPSO and NSGA-II based on the two performance indicators.

Table 5-4 HV comparison results of the fuzzed benchmarks

Instance	Criterion	MOPIO	NSGA-II	MOPSO
$30\times3\times5$	Avg(HV)	410140	369320	394800
	Sd(HV)	35771	38432	34513
$40\times6\times10$	Avg(HV)	296680	227000	340670
	Sd(HV)	33643	36029	42257
$60\times9\times15$	Avg(HV)	260000	198240	225760
	Sd(HV)	66773	60707	45271

Table 5-5 CR comparison results of the fuzzed benchmarks

Instance	Criterion	MOPIO vs. NSGA-II	NSGA-II vs. MOPIO	MOPIO vs. MOPSO	MOPSO vs. MOPIO
$30\times3\times5$	Avg(CR)	0.7019	0.3904	0.7762	0.6362
	Sd(CR)	0.3260	0.2850	0.3016	0.2784
$40\times6\times10$	Avg(CR)	0.5584	0.4338	0.5456	0.5219
	Sd(CR)	0.2912	0.2627	0.2530	0.3083
$60\times9\times15$	Avg(CR)	0.5519	0.4986	0.71	0.5342
	Sd(CR)	0.334	0.3163	0.2967	0.34

Table 5-6 HV comparison results of the random benchmarks

Instance	Criterion	MOPIO	NSGA-II	MOPSO
$20\times2\times4$	Avg(HV)	370120	259960	420310
	Sd(HV)	42687	52709	44402
$35\times4\times6$	Avg(HV)	257290	115370	117340
	Sd(HV)	15086	26714	23554
$65\times8\times10$	Avg(HV)	357070	210300	337900
	Sd(HV)	24190	38107	36947

Table 5-7 CR comparison results of the random benchmarks

Instance	Criterion	MOPIO vs. NSGA-II	NSGA-II vs. MOPIO	MOPIO vs.MOPSO	MOPSO vs.MOPIO
$20\times2\times4$	Avg(CR)	0.6842	0.4734	0.7957	0.7018
	Sd(CR)	0.3189	0.2486	0.1846	0.2225
$35\times4\times6$	Avg(CR)	0.8664	0.2704	0.5145	0.5269
	Sd(CR)	0.1158	0.2374	0.4783	0.4566
$65\times8\times10$	Avg(CR)	0.7290	0.3524	0.5578	0.5319
	Sd(CR)	0.3821	0.3453	0.3433	0.3339

Table 5-8 HV comparison results of MOPIO and MOPIO-GBA

Instance	Criterion	MOPIO	MOPIO-GBA
20×2×4	Avg(HV)	370120	174130
	Sd(HV)	42687	11377
30×3×5	Avg(HV)	410140	160680
	Sd(HV)	35771	11802
35×4×6	Avg(HV)	257290	126860
	Sd(HV)	42687	59960
40×6×10	Avg(HV)	296680	157030
	Sd(HV)	33643	48312
60×9×15	Avg(HV)	260000	103150
	Sd(HV)	66773	51170
65×8×10	Avg(HV)	357070	276271
	Sd(HV)	24190	25374

Table 5-9 CR comparison results of MOPIO and MOPIO-GBA

Instance	Criterion	MOPIO vs. MOPIO-GBA	MOPIO-GBA vs. MOPIO
20×2×4	Avg(CR)	0.6571	0.3122
	Sd(CR)	0.4645	0.1836
30×3×5	Avg(CR)	0.3667	0.0989
	Sd(CR)	0.5507	0.1713
35×4×6	Avg(CR)	0.3490	0.1724
	Sd(CR)	0.5643	0.2529
40×6×10	Avg(CR)	0.4713	0.4111
	Sd(CR)	0.4417	0.4070
60×9×15	Avg(CR)	0.5360	0.4029
	Sd(CR)	0.3725	0.3071
65×8×10	Avg(CR)	0.8996	0.3242
	Sd(CR)	0.1273	0.0963

Table 5-10 The unit fuzzy processing time of different moulds for instance (20×2×4)

No.	Machine 1	Machine 2
Mould 1	(43 45 53)	(43 45 53)
Mould 2	(0 0 0)	(29 33 37)
Mould 3	(0 0 0)	(45 48 57)
Mould 4	(33 38 46)	(33 38 46)

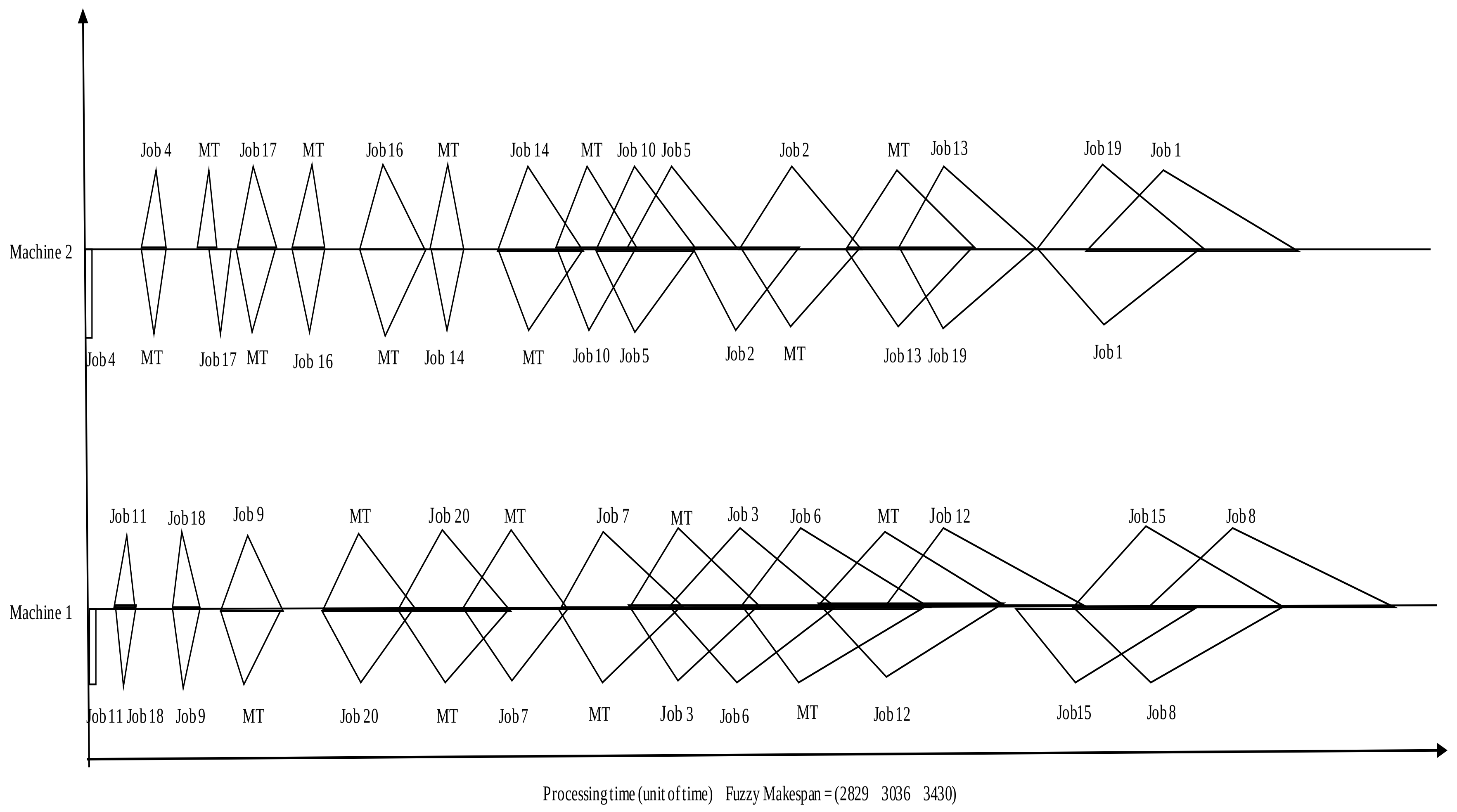


Figure 5-11 The machine scheduling for the instance（20 × 2 × 4）.

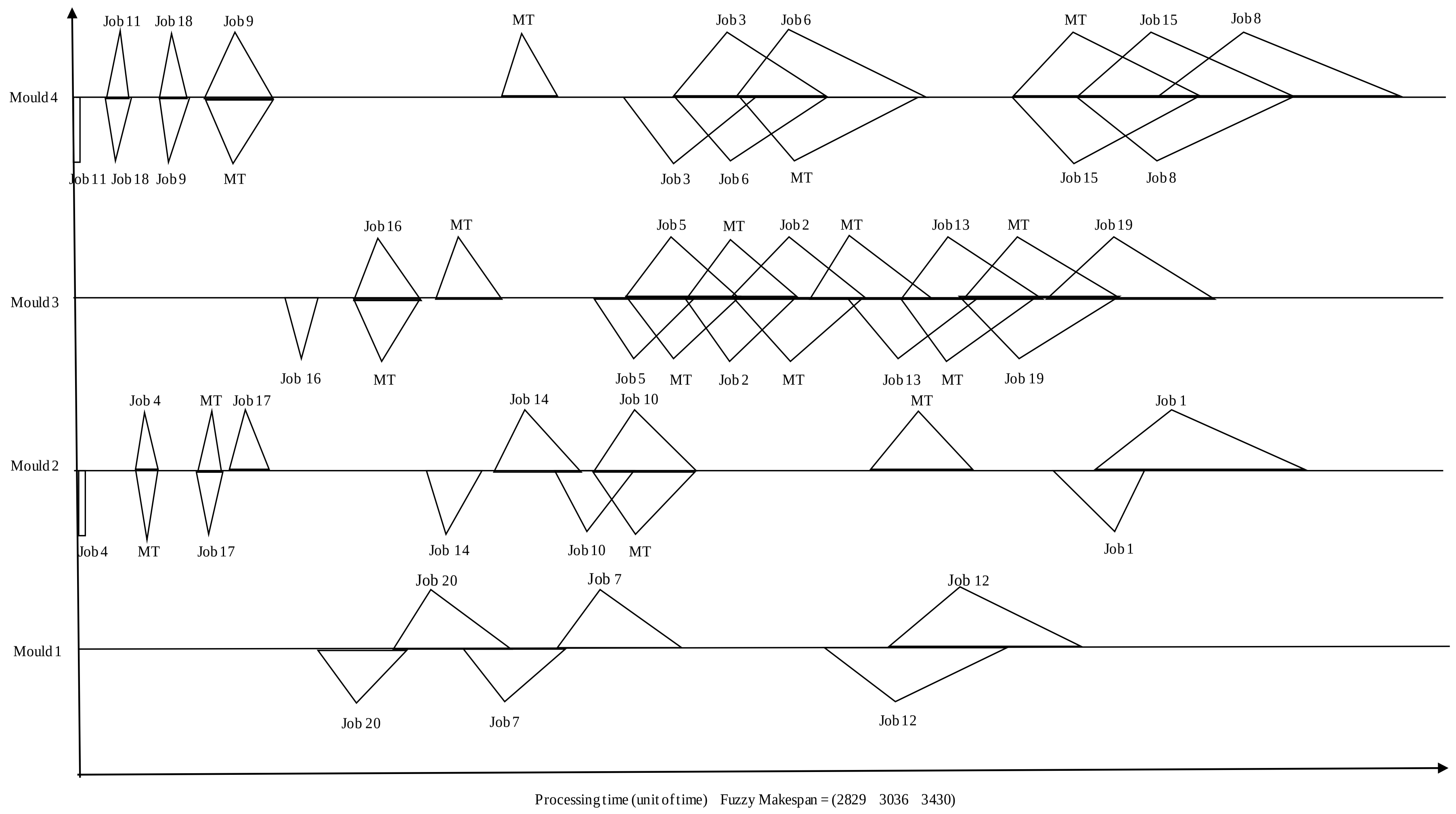


Figure 5-12 The mould scheduling for the instance（20 × 2 × 4）

From Figure 5-9, although the PF by MOPSO is in the left bottom of the coordinate system compared with the PFs by MOPIO and NSGA-II, the boundary of the PF by MOPIO is larger than the PFs by MOPSO and NSGA-II. Turning back to the values in Table 5-4 and Table 5-5, it can be seen that for the dataset $(40 \times 6 \times 10)$, the value of Avg(HV) of the PF by MOPSO is bigger than the values of the PFs by MOPIO and NSGA-II. However, the value of the Avg(CR(MOPIO, MOPSO)) is bigger than the value of the Avg(CR(MOPSO, MOPIO)) and the value of the Avg(CR(MOPIO, NSGA-II)) is bigger than the value of the Avg(CR(NSGA-II, MOPIO)).

From Table 5-6 and Table 5-7, it can be known that the value of the Avg(HV) for MOPIO is better than the values of Avg(HV) for MOPSO and NSGA-II for the bigger datasets which are randomly generated. However, for the small dataset, the value of the Avg(HV) for MOPSO is better than the value of the Avg(HV) for MOPIO and NSGA-II. For these three datasets, the value of the Avg(CR(MOPIO, MOPSO)) is bigger than the value of the Avg(CR(MOPSO, MOPIO)) and the value of the Avg(CR(MOPIO, NSGA-II)) is bigger than the value of the Avg(CR(NSGA-II, MOPIO)).

Furthermore, from the Table 5-8 and Table 5-9, it can be seen that for all these six instances, the proposed MOPIO using the neighbourhood best for each pigeon always has a better performance than the MOPIO using the global best pigeon in the map and compass operation, in terms of the CR and the HV.

After analyzing the PFs of the different algorithms in different aspects, we can conclude that the PFs obtained by MOPIO always have a better CR compared with MOPSO and NSGA-II. For most of the datasets, the HV of the solutions obtained by MOPIO is better than MOPSO and NSGA-II. Moreover, it can be seen that the modification of the velocity updating equation in the map and compass operation is effective after comparing the proposed MOPIO with neighbourhood best with MOPIO with the global best. Because of the mechanism of the index-based ring topology used in the MOPIO, more Pareto-optimal solutions are located compared with using the global best pigeon for all the pigeons. Furthermore, since stable niches are induced by the ring topology in decision space, each pigeon is able to advance in its own niche. In each niche, a pioneer is selected and if these pioneers have good distribution, there is a high possibility that more Pareto-optimal solutions can be located. Moreover, the non-dominated-scd-sort algorithm applied in the MOPIO for sorting the pigeons helps in choosing the pigeons with better performance and distribution in every iteration. The solutions which are less crowded have more opportunities to survive. At the same time, if solutions are near each other in the objective space but not crowded in the decision space, they also have opportunities to be reserved, based on the ranking algorithm. The diversity of the population is improved by the ring topology and the non-dominated-scd-sort algorithm used in the proposed algorithm (Yue et al. 2018), which makes it have a better performance when dealing with the multi-objective problem.

5.5 Summary

In this section, the FPSP-MM is studied. In subsection 5.1, some basic information about this problem is given. The processing time and maintenance time are represented by triangular fuzzy numbers. Two objectives are optimized, the fuzzy makespan and the robustness. In subsection 5.2, a MOPIO algorithm is proposed to solve this multi-objective fuzzy problem. To extend the basic PIO algorithm from the single-objective case to the multi-objective case, a special non-dominated sorting method is used to obtain solutions that are used as candidates for the leader pigeon, and a good distribution of solutions in the objective space and in the corresponding decision space is guaranteed. Moreover, we make each pigeon exchange information with its closed neighbors, instead of the global best pigeon, with the help of index-based ring topology. The diversity of the population is improved by forming more niches. In the subsection 5.3 and subsection 5.4, a series of experiments on the fuzzified benchmarks from existing literature and some randomly generated instances show the efficiency and effectiveness of the proposed MOPIO algorithm by comparing it with other algorithms.

Chapter 6. Conclusions and Future Work

This chapter presents a summary of the research, which provides the main contributions of the research, followed by the limitations of the research. After analysing the model and algorithm deeply, the future research direction is pointed out.

6.1 Summary of the Thesis

(1) A literature review regarding the production scheduling problem with maintenance from various aspects is provided to facilitate the understanding of the current state-of-the-art research related to the production scheduling problem. Traditional production scheduling problems are always under the assumption that the resources are available during the whole operation. While in practice, resources may break down, which interrupts the planned production scheduling. Integrated production scheduling problem with maintenance attracts the attention of more and more researchers. Nevertheless, most of these studies only pay attention to machine maintenance and the investigation of the availability of other critical resources such as injection moulds is limited. Therefore, it is identified that the optimization of the production scheduling problem with maintenance is a promising research area.

(2) Most of the research related to Production Scheduling with Mould Maintenance (PS-MM) problem is solved by a Genetic Algorithm (GA). While, in this study, Particle Swarm Optimization (PSO) is designed to resolve this problem. A new encoding and decoding method is proposed. And the numerical experiments illustrate

the benefits of the developed Particle Swarm Optimization (PSO) by comparing it with a Genetic Algorithm (GA).

(3) This research proposes a new hybrid algorithm named the TLPSO-VNS algorithm for Production Scheduling with Mould Maintenance (PS-MM) problem, which combines the Three-Level Particle Swarm Optimization (TLPSO) algorithm and Variable Neighbourhood Search (VNS). Differing from the joint scheduling approach, this integrated problem is divided into three sub-problems: production scheduling problem, machine maintenance problem and mould maintenance problem. Three interrelated PSOs are used in the solution and is named as Three-Level Particle Swarm Optimization (TLPSO). After obtaining good solutions from TLPSO, VNS is conducted to all these solutions to improve the ability of local exploration. Among all the solutions enhanced by VNS, the solution with the best performance is selected. This is the first algorithm to hybrid TLPSO with VNS to solve the Production Scheduling with Mould Maintenance (PS-MM) problem, aiming at minimizing the overall makespan. The optimization reliability of TLPSO-VNS is tested in numerical experiments based on benchmarks from the literature and other data generated randomly. The results illustrate that the developed TLPSO-VNS algorithm is effective in generating solutions with superior quality in acceptable computation time. It is also shown that the problem decomposition mechanism employed in TLPSO works well in this integrated problem, and VNS is effective in enhancing this hybrid algorithm’s

ability of local search.

(4) Except for the primary problem, a new research question that has never been studied is proposed in this research. In this new research question, multiple maintenances for resources are considered. In the traditional integrated problem, maintenance is preventive maintenance and there is usually one kind of maintenance (perfect maintenance). However, in practice, there are various maintenances and these maintenances have different influences on the states of the maintenance. The integrated problem becomes complex when all these factors are considered. To address this problem, a TLPSO algorithm is proposed. In this algorithm, the first level PSO is mainly designed for the production scheduling problem; the second level PSO is mainly for the machine maintenance problem and the third level PSO is designed for the mould maintenance. For this problem, a special encoding and decoding method is designed. Finally, to prove the efficiency of this algorithm, numerical experiments are done, and other PSO variants are selected to compare with the proposed TLPSO. The results indicate that the proposed TLPSO surpasses other PSO variants.

(5) This study was also an attempt to investigate the fuzzy production scheduling with mould maintenance. In the existing research on the production scheduling problem with mould maintenance, all the information is determinate, which is not realistic. This research considers the uncertainty in this model and uses a triangular fuzzy number to

represent the fuzziness. Except for the fuzzy makespan, the robustness is also considered in this research. A new algorithm with the name of multiple-objective pigeon inspired optimization algorithm is developed to resolve this problem. This is the first time that the variant of the pigeon inspired optimization algorithm is applied to resolve the production scheduling problem and the efficiency is indicated by the numerical experiments.

6.2 Limitations of the Thesis

In this study, there are many constraints and assumptions. For example, in all the research questions, it is assumed that the maintenance could prevent all the breakdowns. However, in a practical situation, breakdowns may also occur even if there are preventive maintenances. These breakdowns usually randomly appear. Besides, in this research, all the jobs were well prepared at the beginning time, however, in real cases, some new jobs may arrive during the production process. When the new tasks are considered, the original schedule needs to be adjusted to obtain better solutions. Moreover, this research only considers the time-based preventive maintenance, some more maintenance strategies such as periodic preventive and condition-based maintenance tasks are not explored. All these factors are not considered in the research questions.

Furthermore, transportation time and setup times are neglected in this thesis. If these factors are considered, the models could be more practical. Also, the batch size of the jobs is assumed to be unsplit, which is also not practical. Moreover, only two kinds of objectives are considered in the research questions (the fuzzy makespan and the robustness), while in a real situation, a decision-maker need to consider multi-objective such as the maintenance cost, the earliness, tardiness. And these objectives may conflict with each other.

Last but not the least, although the proposed TLPSO-VNS algorithm and TLPSO algorithm could achieve solutions with better quality, the search time needed is longer compared with other algorithms because there are repetitions in the searching process. Some new mechanisms need to be added to enhance search efficiency. Moreover, as a new proposed algorithm, the Pigeon Inspired Optimization (PIO) algorithm has enormous potential in the application of the production scheduling problem. More mechanisms should be explored to improve the effectiveness of the algorithm based on MOPIO algorithm.

6.3 Directions of Future Research

The overall research framework is introduced in Figure 6-1, which indicates the overall research procedure. For the optimization problem, it contains a specific model and a specific algorithm. In the beginning, a basic problem is resolved by a simple algorithm. Then, the model or algorithm is improved. In the end, both the model and the algorithm are enhanced.

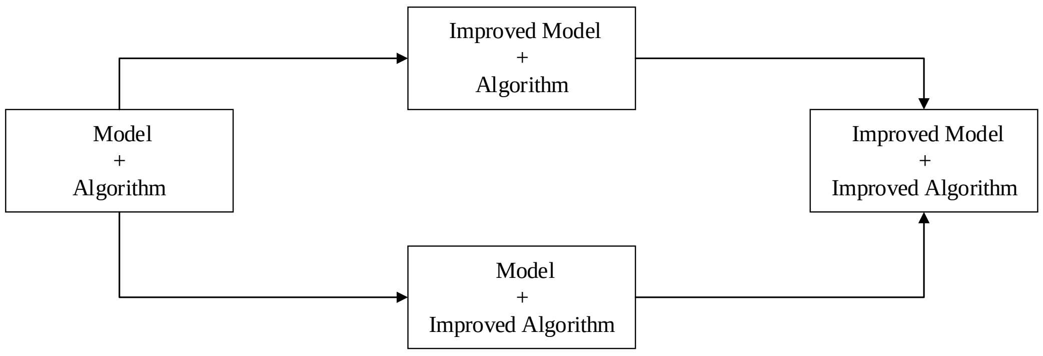


Figure 6-1 The overall research framework

Future research has two directions. From the aspect of the model, random breakdowns will be considered in the model. In addition, if possible, some other factors such as the transportation time, the random insertion, and other maintenance strategies will be added in the model. Besides, other objectives such as maintenance cost, tardiness, will also be considered in the model in the future. From the aspect of the algorithm, more strategies will be added to the TLPSO algorithm to reduce the search repetition, thus improving the efficiency of the TLPSO algorithm. Moreover, the application of the Pigeon Inspired Optimization (PIO) algorithm in the production scheduling problem with maintenance is will be explored and more mechanisms will be investigated.

Reference

Abdelrahim, E. H., and Vizvari, B., 2017. Simultaneous scheduling of production and preventive maintenance on a single machine. Arabian Journal for Science and Engineering, 42(7), 2867-2883. doi:10.1007/s13369-016-2290-4

Abdullah, S., and Abdolrazzagh-Nezhad, M., 2014. Fuzzy job-shop scheduling problems: A review. Information Sciences, 278, 380-407. doi:10.1016/j.ins.2014.03.060

Aghezzaf, E. H., and Najid, N. M. 2008. Integrated production planning and preventive maintenance in deteriorating production systems. Information Sciences, 178(17), 3382-3392. doi:10.1016/j.ins.2008.05.007

Amir-Mohammad, G., Hamid, B.-A., Hamed Jafar, Z., and Hamid, T., 2016. A genetic algorithm for preemptive scheduling of a single machine. International Journal of Industrial Engineering Computations, 7(4), 607-614. doi:10.5267/j.ijiec.2016.3.004

Bagheri, A., and Zandieh, M., 2011. Bi-criteria flexible job-shop scheduling with sequence-dependent setup times-Variable neighborhood search approach. Journal of Manufacturing Systems, 30(1), 8-15. doi:10.1016/j.jmsy.2011.02.004

Bathrinath, S., Sankar, S. S., Ponnambalam, S. G., and Leno, I. J., 2015. VNS-Based heuristic for identical parallel machine scheduling problem. Artificial

Intelligence and Evolutionary Algorithms in Engineering Systems. Advances in Intelligent Systems and Computing, 324, 693-699. doi: 10.1007/978-81-322-2126-5_74

Bean, J. C., 1994. Genetic algorithms and random keys for sequencing and optimization. ORSA Journal on Computing, 6(2), 154-160. doi:10.1287/ijoc.6.2.154

Ben M. A., Chelbi, A. and Radhoui, M., 2016. Optimal imperfect maintenance strategy for leased equipment. International Journal of Production Economics, 178: 57-64. doi:10.1016/j.ijpe.2016.04.024.

Behnamian, J., and Ghomi, S., 2011. Hybrid flowshop scheduling with machine and resource-dependent processing times. Applied Mathematical Modelling, 35(3), 1107-1123. doi:10.1016/j.apm.2010.07.057

Berrade, M. D., Cavalcante, C. A., and Scarf, P. A., 2012. Maintenance scheduling of a protection system subject to imperfect inspection and replacement. European Journal of Operational Research, 218 (3): 716–725. doi: 10.1016/j.ejor.2011.12.003

Berrichi, A., Yalaoui, F., Amodeo, L., and Mezghiche, M., 2010. Bi-Objective ant colony optimization approach to optimize production and maintenance scheduling. Computers & Operations Research, 37(9), 1584-1596. doi:10.1016/j.cor.2009.11.017

Borkowski, S., and Stasiak-Betlejewska, R., 2015. The machines maintenance conditions assessment in the plastic industry. Production Engineering Archives, 9(4): 17-20.

Cassady, C. R., and Kutanoglu, E., 2003. Minimizing job tardiness using integrated preventive maintenance planning and production scheduling. Iie Transactions, 35(6), 503-513. doi:10.1080/07408170390187951

Cassady, C. R., and Kutanoglu, E., 2005. Integrating preventive maintenance planning and production scheduling for a single machine. Ieee Transactions on Reliability, 54(2), 304-309. doi:10.1109/tr.2005.845967

Coello, C. A. C., Pulido, G. T., and Lechuga, M. S., 2004. Handling multiple objectives with particle swarm optimization. IEEE Transactions on evolutionary computation, 8(3), 256-279.

Christopher, P. 2006. Production Scheduling: Collins.

Chaudhry, I. A., and Khan, A. A., 2016. A research survey: review of flexible job shop scheduling techniques. International Transactions in Operational Research, 23(3), 551-591. doi:10.1111/itor.12199

Chung, S. H., Lau, H. C. W., Ho, G. T. S., and Ip, W. H., 2009. Optimization of system reliability in multi-factory production networks by maintenance approach. Expert Systems with Applications, 36(6), 10188-10196. doi:10.1016/j.eswa.2008.12.014

Dangel, R. 2016. Injection moulds for beginners: Munich : Hanser Publishers.

Dao, S., Abhary, K., and Marian, R., 2017. An improved genetic algorithm for multidimensional optimization of precedence-constrained production planning and scheduling. Journal of Industrial Engineering International, 13(2), 143-159. doi:10.1007/s40092-016-0181-7

Deng, Y., and Duan, H., 2016. Control parameter design for automatic carrier landing system via pigeon-inspired optimization. Nonlinear Dynamics, 85(1), 97-106. doi.:10.1007/s11071-016-2670-z

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A. M. T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

Do, P., Voisin, A., Levrat, E., and Iung, B., 2015. A proactive condition-based maintenance strategy with both perfect and imperfect maintenance actions. Reliability Engineering and System Safety, 133: 22-32. doi: 10.1016/j.ress.2014.08.011.

Duan, C., Deng, C., Gharaei, A., Wu, J. and Wang, B., 2018. Selective maintenance scheduling under stochastic maintenance quality with multiple maintenance actions. International Journal of Production Research, 56(23), 7160-7178. doi:10.1080/00207543.2018.1436789

Duan, H. B., Qiao, P. X., 2014. Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics, 7(1), 24-37.

Duan, H., and Wang, X., 2015. Echo state networks with orthogonal pigeon-inspired optimization for image restoration. IEEE transactions on neural networks and learning systems, 27(11), 2413-2425. doi: 10.1109/TNNLS.2015.2479117

Diallo, C., Venkatadri, U., Khatab, A., Liu, Z. and Aghezzaf, E. H., 2018. Optimal joint selective imperfect maintenance and multiple repairpersons assignment strategy for complex multicomponent systems. International Journal of Production Research. doi:10.1080/00207543.2018.1505060

Eberhart, R. C., and Shi, Y., 2000. Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation. CEC00. 1, 84-88. doi:10.1109/CEC.2000.870279

El Khoukhi, F., Boukachour, J., and Alaoui, A. E., 2017. The "Dual-Ants Colony": A novel hybrid approach for the flexible job shop scheduling problem with preventive maintenance. Computers & Industrial Engineering, 106, 236-255. doi:10.1016/j.cie.2016.10.019

El-Abd, M., 2013. Testing a particle swarm optimization and artificial bee colony hybrid algorithm on the CEC13 benchmarks. In: 2013 IEEE Congress on Evolutionary Computation, pp. 2215-2220. doi: 10.1109/CEC.2013.6557832

Fortemps, P., 1997. Jobshop scheduling with imprecise durations: a fuzzy approach. IEEE Transactions on Fuzzy Systems, 5(4), 557-569. doi: 10.1109/91.649907

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Chong, C. S., and Cai, T. X., 2016. An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time. Expert Systems with

Applications, 65, 52-67. doi:10.1016/j.eswa.2016.07.046

Gao, K. Z., Suganthan, P. N., Pan, Q. K., and Tasgetiren, M. F., 2015. An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time. International Journal of Production Research, 1-16. doi:10.1080/00207543.2015.1020174

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Tasgetiren, M. F., and Sadollah, A., 2016 . Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion. Knowledge-Based Systems, 109, 1-16. doi:10.1016/j.knosys.2016.06.014

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., and Chong, C. S., 2016. Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. Journal of Intelligent Manufacturing, 27(2), 363-374. doi: 10.1007/s10845-014-0869-8.

Gao, K., Zhang, Y., Sadollah, A., Lentzakis, A., and Su, R., 2017. Jaya, harmony search and water cycle algorithms for solving large-scale real-life urban traffic light scheduling problem. Swarm and evolutionary computation, 37, 58-72.. doi:10.1016/j.swevo.2017.05.002

Gao, K., Zhang, Y., Su, R., Yang, F., Suganthan, P. N., and Zhou, M., 2018. Solving traffic signal scheduling problems in heterogeneous traffic network by using meta-heuristics. IEEE Transactions on Intelligent Transportation Systems. doi: 10.1109/TITS.2018.2873790

Gao, K., Yang, F., Zhou, M., Pan, Q., and Suganthan, P. N., 2019. Flexible job-shop rescheduling for new job insertion by using discrete Jaya algorithm. IEEE transactions on cybernetics, 49(5) 1944-1955. doi: 10.1109/TCYB.2018.2817240

Gao, L., Li, X. Y., Wen, X. Y., Lu, C., and Wen, F., 2015. A hybrid algorithm based on a new neighborhood structure evaluation method for job shop scheduling problem. Computers & Industrial Engineering, 88, 417-429. doi:10.1016/j.cie.2015.08.002

Guner, H. U., Chinnam, R. B., and Murat, A., 2016. Simulation platform for anticipative plant-level maintenance decision support system. International Journal of Production Research, 54(6), 1785-1803. doi:10.1080/00207543.2015.1064179

Hadi, M., and Mehrdad, D., 2015. A flexible job shop scheduling problem with controllable processing times to optimize total cost of delay and processing. International Journal of Supply and Operations Management, 2(3), 871-887.

Ham, A., 2017. Flexible job shop scheduling problem for parallel batch processing machine with compatible job families. Applied Mathematical Modelling, 45, 551-562. doi:10.1016/j.apm.2016.12.034

Holland, J. H., 1992. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press.

Hu, Y., Yin, M., and Li, X., 2011. A novel objective function for job-shop scheduling problem with fuzzy processing time and fuzzy due date using differential evolution algorithm. The International Journal of Advanced Manufacturing Technology, 56(9), 1125-1138. doi:10.1007/s00170-011-3244-3

Huang, S., Tian, N., Wang, Y., and Ji, Z., 2016. An improved version of discrete particle swarm optimization for flexible job shop scheduling problem with fuzzy processing time. Mathematical Problems in Engineering, 2016. doi:10.1155/2016/5958640

Huang, S., Tian, N., and Ji, Z. C., 2016. Particle swarm optimization with variable neighborhood search for multiobjective flexible job shop scheduling problem. International Journal of Modeling Simulation and Scientific Computing, 7(3), 1650024. doi:10.1142/s1793962316500240

José Palacios, J., González-Rodríguez, I., Vela, C. R., and Puente, J., 2017. Robust multiobjective optimisation for fuzzy job shop problems. Applied Soft Computing, 56, 604-616. doi:10.1016/j.asoc.2016.07.004

Khademi Zare, H., and Fakhrzad, M. B., 2011. Solving flexible flow-shop problem with a hybrid genetic algorithm and data mining: A fuzzy approach. Expert Systems with Applications, 38(6), 7609-7615. doi:10.1016/j.eswa.2010.12.101

Kuroda, M., and Wang, Z., 1996. Fuzzy job shop scheduling. Interational Journal of Production Economic., 44(1-2), 45-51. doi:10.1016/0925-5273(95)00091-7

Kennedy, J., and Eberhart, R., 1995. Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks. 4, 1942-1948. doi: 10.1109/ICNN.1995.488968

Kurdi, M., 2016. An effective new island model genetic algorithm for job shop scheduling problem. Computers and Operations Research, 67, 132-142. doi:10.1016/j.cor.2015.10.005

Komaki, M., and Malakooti, B., 2017. General variable neighborhood search algorithm to minimize makespan of the distributed no-wait flow shop scheduling problem. Production Engineering-Research and Development, 11(3), 315-329. doi:10.1007/s11740-017-0716-9

Lan, F., Wang, B., and Zhang, X., 2016. Improved variable neighbourhood search algorithm for robust job shop scheduling problems. In: 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2, 592-595.

Le, M. D., Tan, C. M., 2013. Optimal maintenance strategy of deteriorating system under imperfect maintenance and inspection using mixed inspection scheduling. Reliability Engineering and System Safety, 113, 21-29. doi: 10.1016/j.ress.2012.11.025

Lei, D., 2008. Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems. The International Journal of Advanced Manufacturing Technology, 37(1), 157-165. doi:10.1007/s00170-007-0945-8

Lei, D., 2010a. Fuzzy job shop scheduling problem with availability constraints. Computers & Industrial Engineering, 58(4), 610-617. doi:10.1016/j.cie.2010.01.002

Lei, D., 2010b. A genetic algorithm for flexible job shop scheduling with fuzzy processing time. International Journal of Production Research, 48(10), 2995-3013. doi:10.1080/00207540902814348

Lei, D., 2010c. Solving fuzzy job shop scheduling problems using random key genetic algorithm. The International Journal of Advanced Manufacturing Technology, 49(1), 253-262. doi:10.1007/s00170-009-2379-y

Lei, D., 2011. Population-based neighborhood search for job shop scheduling with interval processing time. Computers and Industrial Engineering, 61(4), 1200-1208. doi:10.1016/j.cie.2011.07.010

Lei, D., 2011. Scheduling fuzzy job shop with preventive maintenance through swarm-based neighborhood search. The International Journal of Advanced Manufacturing Technology, 54(9), 1121-1128. doi:10.1007/s00170-010-2989-4

Lei, D., 2012. Co-evolutionary genetic algorithm for fuzzy flexible job shop scheduling. Applied Soft Computing Journal, 12(8), 2237-2245. doi:10.1016/j.asoc.2012.03.025

Lei, D., and Guo, X., 2012. Swarm-based neighbourhood search algorithm for fuzzy flexible job shop scheduling. International Journal of Production Research, 50(6), 1639-1649. doi:10.1080/00207543.2011.575412

Li, C., and Duan, H., 2014. Target detection approach for UAVs via improved pigeon-inspired optimization and edge potential function. Aerospace Science and Technology, 39, 352-360. doi:10.1016/j.ast.2014.10.007

Li, G., Liu, M. Q., Sethi, S. P., and Xu, D. H., 2017. Parallel-machine scheduling with machine-dependent maintenance periodic recycles. International Journal of Production Economics, 186, 1-7. doi:10.1016/j.ijpe.2017.01.014

Li, J. Q., and Pan, Q. K., 2013. Chemical-reaction optimization for solving fuzzy job-shop scheduling problem with flexible maintenance activities. International Journal of Production Economics, 145(1), 4-17. doi:10.1016/j.ijpe.2012.11.005

Li, J. Q., and Pan, Y. X., 2013. A hybrid discrete particle swarm optimization algorithm for solving fuzzy job shop scheduling problem. The International Journal of Advanced Manufacturing Technology, 66(1), 583-596. doi:10.1007/s00170-012-4337-3

Li, J. Q., Pan, Q. K., Mao, K., and Suganthan, P. N., 2014. Solving the steelmaking casting problem using an effective fruit fly optimisation algorithm. Knowledge-Based Systems, 72, 28-36. doi: 10.1016/j.knosys.2014.08.022

Liao, C. J., and Cheng, C. C., 2007. A variable neighborhood search for minimizing single machine weighted earliness and tardiness with common due date. Computers and Industrial Engineering, 52(4), 404-413.

doi:10.1016/j.cie.2007.01.004

Liao, G. L. and Sheu, S. H., 2011, Economic production quantity model for randomly failing production process with minimal repair and imperfect maintenance. International Journal of Production Economics. 130(1), 118-124 doi.org/10.1016/j.ijpe.2010.12.004.

Lian, Z. G., Gu, X. S., and Jiao, B., 2006. A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan. Applied Mathematics and Computation, 175(1), 773-785. doi:10.1016/j.amc.2005.07.042

Lian, Z. G., Gu, X. S., and Jiao, B., 2008. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan. Chaos Solitons and Fractals, 35(5), 851-861. doi:10.1016/j.chaos.2006.05.082

Liang, Y. C., and Tien, C. Y., 2011. Variable neighborhood search for drilling operation scheduling in PCB Industries. In: International Conference on Intelligent Computing pp: 55-62. Springer, Berlin, Heidelberg. doi: https://doi.org/10.1007/978-3-642-24728-6_8

Lin, J., 2015. A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem. Knowledge-Based Systems, 78, 59-74. doi:10.1016/j.knosys.2015.01.017

Liu, A. J., Yang, Y., Xing, Q. S., Lu, H., and Zhang, Y.-D., 2011. Multi-population genetic algorithm in multiobjective fuzzy and flexible Job Shop scheduling.

Computer Integrated Manufacturing Systems, 17(9), 1954-1961

Liu, B., Fan, Y., and Liu, Y., 2015. A fast estimation of distribution algorithm for dynamic fuzzy flexible job-shop scheduling problem. Computers and Industrial Engineering, 87, 193-201. doi:10.1016/j.cie.2015.04.029

Liu, B., Wang, L., and Jin, Y. H., 2007. An effective hybrid particle swarm optimization for no-wait flow shop scheduling. International Journal of Advanced Manufacturing Technology, 31(9-10), 1001-1011. doi:10.1007/s00170-005-0277-5

Liu, L., and Zhou, H., 2013. Hybridization of harmony search with variable neighborhood search for restrictive single-machine earliness/tardiness problem. Information Sciences, 226, 68-92. doi:10.1016/j.ins.2012.11.007

Liu Y., Huang H. Z., Wang Z. L., Li Y. F. and Yang Y. J., 2013. A Joint redundancy and imperfect maintenance strategy optimization for multi-state systems. IEEE Transactions on Reliability, 62(2): 368-378. doi: 10.1109/TR.2013.2259193

Lokensgard, E. 2017. Industrial plastics : theory and applications (6th editioned.): Boston, MA : Cengage Learning.

Lu, B.Y., and Cheng, B.Y., 2010a. Hybrid particle swarm optimization for fuzzy flexible job-shop scheduling problem. Jisuanji Yingyong Yanjiu / Application Research of Computers, 27(10), 3721-3723. doi:10.3969/j.issn.1001-3695.2010.10.030

Lu, B. Y., and Cheng, B. Y., 2010b. Research of fuzzy job-shop scheduling problem based on hybrid DEA-GA algorithm. Jisuanji Yingyong Yanjiu / Application Research of Computers, 27(8), 2933-2935. doi:10.3969/j.issn.1001-3695.2010.08.033

Lu, C., Li, X., Gao, L., Liao, W., and Yi, J., 2017. An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times. Computers and Industrial Engineering, 104, 156-174. doi:10.1016/j.cie.2016.12.020

May, G., Stahl, B., Taisch, M., and Prabhu, V., 2015. Multi-objective genetic algorithm for energy-efficient job shop scheduling. International Journal of Production Research, 53(23), 7071-7089.. doi:10.1080/00207543.2015.1005248

Mladenović, N., and Hansen, P., 1997. Variable neighborhood search. Computers and Operations Research, 24(11), 1097-1100. doi:10.1016/S0305-0548(97)00031-2

Mokhtari, H., 2014. A two-stage no-wait job shop scheduling problem by using a neuro-evolutionary variable neighborhood search. International Journal of Advanced Manufacturing Technology, 74(9-12), 1595-1610. doi:10.1007/s00170-014-6086-y

Mokhtari, H., Mozdgir, A., and Abadi, I. N. K., 2012. A reliability/availability approach to joint production and maintenance scheduling with multiple preventive maintenance services. International Journal of Production Research, 50(20), 5906-5925. doi:10.1080/00207543.2011.637092

Nepomuceno, F. V., and Engelbrecht, A. P., 2013. A self-adaptive heterogeneous pso for real-parameter optimization. In: 2013 IEEE congress on evolutionary computation (pp. 361-368). doi: 10.1109/CEC.2013.6557592

Nguyen, D. T., Dijoux, Y., Fouladirad, M. 2017., Analytical properties of an imperfect repair model and application in preventive maintenance scheduling. European Journal of Operational Research. 256(2), 439-453. doi.org/10.1016/j.ejor.2016.06.026

Olorunniwo, F. O., Izuchukwu, A., 1991. Scheduling imperfect preventive and overhaul maintenance. International Journal of Quality & Reliability Management, 8(4), 67-80. doi.org/10.1108/02656719110141123

Palacios, J. J., González, M. A., Vela, C. R., González-Rodríguez, I., and Puente, J., 2015. Genetic tabu search for the fuzzy flexible job shop problem. Computers and Operations Research, 54, 74-89. doi:10.1016/j.cor.2014.08.023

Palacios, J. J., Puente, J., Vela, C. R., and González-Rodríguez, I., 2016. Benchmarks for fuzzy job shop problems. Information Sciences, 329, 736-752. doi:10.1016/j.ins.2015.09.042

Palacios, J. J., González-Rodríguez, I., Vela, C. R., and Puente, J., 2017. Robust multiobjective optimisation for fuzzy job shop problems. Applied Soft Computing, 56, 604-616. doi:10.1016/j.asoc.2016.07.004

Pang, K. W., 2013. A genetic algorithm based heuristic for two machine no-wait flowshop scheduling problems with class setup times that minimizes maximum lateness. International journal of production economics, 141(1), 127-136. doi: 10.1016/j.ijpe.2012.06.017

Peng, Y., and Dong, M., 2011. A prognosis method using age-dependent hidden semi-Markov model for equipment health prediction. Mechanical Systems and Signal Processing, 25, 237–252. doi:10.1016/j.ymssp.2010.04.002

Pham, H. and Wang, H., 1996. Imperfect maintenance. European Journal of Operational Research. 94(3): 425-438. doi:10.1016/S0377-2217(96)00099-9

Qiu, H., and Duan, H., 2015. Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. Science China Technological Sciences, 58(11), 1915-1923. doi: 10.1007/s11431-015-5860-x

Qiu, H., and Duan, H., 2018. A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. Information Sciences. doi:10.1016/j.ins.2018.06.061

Rajkumar, M., Asokan, P., and Vamsikrishna, V., 2010. A GRASP algorithm for flexible job-shop scheduling with maintenance constraints. International Journal of Production Research, 48(22), 6821-6836. doi:10.1080/00207540903308969

Ratnaweera, A., Halgamuge, S. K., and Watson, H. C., 2004. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on evolutionary computation, 8(3), 240-255.

doi: 10.1109/TEVC.2004.826071

Ribeiro, B., 2005. Support vector machines for quality monitoring in a plastic injection molding process. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 35(3): 401-410. doi: 10.1109/TSMCC.2004.843228

Ruiz, R., Garcia-Diaz, J. C., and Maroto, C., 2007. Considering scheduling and preventive maintenance in the flowshop sequencing problem. Computers and Operations Research, 34(11), 3314-3330. doi:10.1016/j.cor.2005.12.007

Sakawa, M., and Kubota, R., 2000. Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. European Journal of Operational Research, 120(2), 393-407. doi:10.1016/S0377-2217(99)00094-6

Sakawa, M., and Mori, T., 1999. An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate. Computers and Industrial Engineering, 36(2), 325-341. doi:10.1016/S0360-8352(99)00135-7

Seidgar, H., Zandieh, M., Fazlollahtabar, H., and Mahdavi, I., 2016. Simulated imperialist competitive algorithm in two-stage assembly flow shop with machine breakdowns and preventive maintenance. In: Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, 230(5), 934-953. doi:10.1177/0954405414563554

Shaheed, I., Shukor, S., and Nababan, E., 2016. An empirical analysis of the relationship between the initialization method performance and the convergencce speed of a meta-heuristic for fuzzy job shop scheduling problems. Journal of Theoretical and Applied Information Technology, 93(2), 297-311.

Shameem, K., Choudhari, K., Bankapur, A., Kulkarni, S., Unnikrishnan, V., George, S., Kartha, V., and Santhosh, C., 2017. A hybrid LIBS–Raman system combined with chemometrics: an efficient tool for plastic identification and sorting. Analytical and Bioanalytical Chemistry, 409(13), 3299-3308. doi:10.1007/s00216-017-0268-z

Shen, L., Yang, H., Gao, S., and Fang, J., 2016. Production scheduling with mould maintenance in flow shop. In: 2016 4th International Conference on Sensors, Mechatronics and Automation (ICSMA 2016). Atlantis Press.

Tang, L. X., and Wang, X. P., 2010. An improved particle swarm optimization algorithm for the hybrid flowshop scheduling to minimize total weighted completion time in process industry. Ieee Transactions on Control Systems Technology, 18(6), 1303-1314. doi:10.1109/tcst.2009.2036718

Tasgetiren, M. F., Liang, Y.-C., Sevkli, M., and Gencyilmaz, G., 2007. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. European Journal of Operational Research, 177(3), 1930-1947. doi:10.1016/j.ejor.2005.12.024

Todosijević, R., Benmansour, R., Hanafi, S., Mladenović, N., and Artiba, A., 2016. Nested general variable neighborhood search for the periodic maintenance problem. European Journal of Operational Research, 252(2), 385-396. doi:10.1016/j.ejor.2016.01.014

Wang, C., Tian, N., Ji, Z., and Wang, Y., 2017. Multi-objective fuzzy flexible job shop scheduling using memetic algorithm. Journal of Statistical Computation and Simulation, 87(14), 2828-2846. doi:10.1080/00949655.2017.1344846

Wang, J., Zhang, J., Qin, W., Yin, L., and Chen, D., 2015. Robust scheduling on flexible job shop with uncertain processing time. Zhongguo Jixie Gongcheng (China Mechanical Engineering), 26(5), 627-632. doi:10.3969/j.issn.1004-132X.2015.05.010

Wang, L., Zhou, G., Xu, Y., and Liu, M., 2013. A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem. International Journal of Production Research, 51(12), 3593-3608. doi:10.1080/00207543.2012.754549

Wang, L., Zhou, G., Xu, Y., and Liu, M., 2013. A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem. International Journal of Production Research, 51(12), 3593-3608. doi:10.1080/00207543.2012.754549

Wang, S., Liu, G., and Gao, S., 2016. A hybrid discrete imperialist competition algorithm for fuzzy job-shop scheduling problems. Access, IEEE, 4, 9320-

9331. doi:10.1109/ACCESS.2016.2645818

Wang, S., Wang, L., Xu, Y., and Liu, M., 2013. An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time. International Journal of Production Research, 51(12), 3778-3793. doi:10.1080/00207543.2013.765077

Wang, S., and Liu, M., 2015. Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning. Journal of Manufacturing Systems, 37, 182-192. doi:10.1016/j.jmsy.2015.07.002

Wang, S. J., and Liu, M., 2015. Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning. Journal of Manufacturing Systems, 37, 182-192. doi:10.1016/j.jmsy.2015.07.002

Wang, S. J., and Liu, M. 2016., Two-machine flow shop scheduling integrated with preventive maintenance planning. International Journal of Systems Science, 47(3), 672-690. doi:10.1080/00207721.2014.900137

Wang, X. Y., Wang, M. Z., and Wang, J. B., 2011. Flow shop scheduling to minimize makespan with decreasing time-dependent job processing times. Computers and Industrial Engineering, 60(4), 840-844. doi:10.1016/j.cie.2011.02.003

Wang, X. P., and Tang, L. X., 2012., A discrete particle swarm optimization algorithm with self-adaptive diversity control for the permutation flowshop problem with blocking. Applied Soft Computing, 12(2), 652-662. doi:10.1016/j.asoc.2011.09.021

Wong, C. S., Chan, F. T. S., and Chung, S. H., 2012. A genetic algorithm approach for production scheduling with mould maintenance consideration. International Journal of Production Research, 50(20), 5683-5697. doi: 10.1080/00207543.2011.613868

Wong, C. S., Chan, F. T. S., and Chung, S. H., 2013. A joint production scheduling approach considering multiple resources and preventive maintenance tasks. International Journal of Production Research, 51(3), 883-896. doi:10.1080/00207543.2012.677070

Wong, C. S., Chan, F. T. S., and Chung, S. H., 2014. Decision-making on multi-mould maintenance in production scheduling. International Journal of Production Research, 52(19), 5640-5655. doi:10.1080/00207543.2014.900200

Xu, Y., Wang, L., Wang, S.-Y., and Liu, M., 2015. An effective teaching-learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time. Neurocomputing, 148, 260-268. Doi: 10.1016/j.neucom.2013.10.042

Xiao, L., Song, S., Chen, X., and Coit, D. W., 2016. Joint optimization of production scheduling and machine group preventive maintenance. Reliability Engineering and System Safety, 146, 68-78. doi:10.1016/j.ress.2015.10.013

Xia, H., Li, X. Y., and Gao, L., 2016. A hybrid genetic algorithm with variable neighborhood search for dynamic integrated process planning and scheduling. Computers and Industrial Engineering, 102, 99-112. doi:10.1016/j.cie.2016.10.015

Yang, H. A., Wang, Z. F., Lyu, Y. Y., Xi, Z. C., and Wang, H. H., 2014. Interval number solving method for job-shop scheduling problem with processing time variability. Computer Integrated Manufacturing Systems, 20(9), 2231-2240. doi:10.13196/j.cims.2014.09.019

Yue, C., Qu, B., and Liang, J., 2018. A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems. IEEE Transactions on Evolutionary Computation, 22(5), 805-817. doi: 10.1109/TEVC.2017.2754271

Zambrano-Bigiarini, M., Clerc, M., and Rojas, R., 2013. Standard particle swarm optimisation 2011 at cec-2013: A baseline for future pso improvements. In: 2013 IEEE Congress on Evolutionary Computation. pp. 2337-2344. doi: 10.1109/CEC.2013.6557848

Zhang, B., and Duan, H., 2017. Three-dimensional path planning for uninhabited combat aerial vehicle based on predator-prey pigeon-inspired optimization in dynamic environment. IEEE/ACM transactions on computational biology and bioinformatics, 14(1), 97-107. doi: 10.1109/TCBB.2015.2443789

Zhang, B., Pan, Q. K., Gao, L., and Zhang, X. L., 2018. A hybrid variable neighborhood search algorithm for the hot rolling batch scheduling problem in compact strip production. Computers & Industrial Engineering, 116, 22-36. doi: 10.1016/j.cie.2017.12.013

Zhang, B., Pan, Q. K., Gao, L., & Zhang, X. L., 2018. A hybrid variable neighborhood search algorithm for the hot rolling batch scheduling problem in compact strip production. Computers & Industrial Engineering, 116, 22-36. doi:10.1016/j.cie.2017.12.013

Zhang, L., and Wong, T. N., 2015. An object-coding genetic algorithm for integrated process planning and scheduling. European Journal of Operational Research, 244(2), 434-444. doi:10.1016/j.ejor.2015.01.032

Zhang, Q., Manier, H., and Manier, M. A., 2012. A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times. Computers & Operations Research, 39(7), 1713-1723.. doi:10.1016/j.cor.2011.10.007

Zhang, R., and Chiong, R., 2016. Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. Journal of Cleaner Production, 112, 3361-3375. doi:10.1016/j.jclepro.2015.09.097

Zhang, S., and Duan, H., 2015. Gaussian pigeon-inspired optimization approach to orbital spacecraft formation reconfiguration. Chinese Journal of Aeronautics, 28(1), 200-205. doi:10.1016/j.cja.2014.12.008

Zheng, Y. L., Li, Y. X., and Lei, D. M., 2010. Scheduling jobs and preventive maintenance on fuzzy job shop using genetic algorithm. In: 2010 International Conference on Machine Learning and Cybernetics, 3, 1583-1589.

Zheng, Y. L., Li, Y. X., and Lei, D. M., 2012. Multi-objective swarm-based neighborhood search for fuzzy flexible job shop scheduling. The International Journal of Advanced Manufacturing Technology, 60(9-12), 1063-1069. doi:10.1007/s00170-011-3646-2

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and Da Fonseca Grunert, V., 2002. Performance assessment of multiobjective optimizers: An analysis and review. TIK-Report, 139. doi:10.3929/ethz-a-004363410

Zobolas, G. I., Tarantilis, C. D., and Ioannou, G., 2009. A hybrid evolutionary algorithm for the job shop scheduling problem. Journal of the Operational Research Society, 60(2), 221-235. doi:10.1057/palgrave.jors.2602534

Appendix

The best scheduling for Data 1 in chapter 3

Job sequence	Machine sequence	Machine maintenance	Mold maintenance
27	3	0	0
29	2	0	1
2	3	0	1
12	3	0	1
30	3	1	1
1	2	1	1
15	1	0	0
20	1	1	1
22	1	1	0
5	2	0	0
4	1	0	0
18	2	0	1
19	3	0	0
26	1	1	0
24	2	1	0
11	1	0	1
23	1	0	0
8	3	0	1
6	2	0	0
16	3	1	1
14	2	0	1
13	3	0	0
3	2	0	0
17	2	0	0
21	3	0	0
10	3	0	0
28	1	0	0
7	1	0	0
9	3	1	0
25	2	1	1

The best scheduling for Data 2 in chapter 3

Job sequence	Machine sequence	Machine maintenance	Mould maintenance
21	6	1	1
9	4	0	0
22	4	0	0
39	1	0	1
24	4	1	0
3	2	1	1
38	5	1	1
15	3	0	0
8	1	1	1
37	5	0	1
12	1	0	0
14	3	0	1
31	6	1	0
29	2	1	1
36	6	1	1
16	5	1	1
32	1	1	0
19	2	0	1
34	2	1	0
4	3	1	1
7	4	0	1
6	4	1	0
26	4	0	0
1	5	0	1
10	5	0	0
11	4	0	1
17	1	0	1
30	6	0	0
25	1	0	0
40	3	0	1
23	6	0	0
27	4	0	1
18	3	1	0
20	1	0	0
33	6	0	0
13	4	1	0
28	5	0	0

5	3	1	1
2	2	0	0
35	1	1	0

The best scheduling for Data 3 in chapter 3

Job sequence	Machine sequence	Machine maintenance	Mould maintenance
41	6	1	0
36	7	0	0
20	1	1	0
27	2	1	0
23	9	0	0
1	2	1	0
42	5	0	1
8	3	1	1
19	5	1	0
54	9	1	1
40	5	1	1
44	2	0	0
47	9	0	0
38	2	1	1
52	6	0	0
30	3	0	1
56	8	0	1
49	9	0	0
28	5	0	0
16	1	1	1
37	8	1	1
35	7	1	1
7	6	1	1
26	1	1	1
60	8	0	1
33	7	1	1
25	4	0	1
59	1	1	0
46	6	0	0
12	3	0	1
6	7	0	1

2	9	1	0
58	9	0	0
17	7	0	0
15	8	1	0
24	6	0	1
13	4	1	0
21	2	0	0
9	9	0	0
5	2	1	1
22	6	1	1
34	8	1	0
53	9	0	0
4	4	1	0
45	1	1	1
51	6	0	0
55	8	0	0
31	5	1	0
32	2	0	1
14	8	0	0
43	6	1	0
29	3	1	1
18	5	1	0
48	1	1	0
3	2	0	0
39	9	0	0
11	7	0	0
10	8	0	1
57	5	0	0
50	3	0	1