A three-level particle swarm optimization with variable neighbourhood search algorithm for the production scheduling problem with mould maintenance

To improve the reliability of production systems in the plastics industry, researchers are now taking mould maintenance into consideration, besides machine maintenance, in production scheduling problem. Different strategies and approaches are proposed to solve the Production Scheduling with Mould Maintenance (PS-MM) problem. However, it remains a challenge to provide a satisfactory solution. In this research, a new hybrid metaheuristic algorithm (TLPSO-VNS algorithm) is proposed, which is a combination of the Three-Level Particle Swarm Optimization (TLPSO) algorithm devised in this study and Variable Neighbourhood Search (VNS). Differing from the joint scheduling strategies used in existing research, this study divides the integrated problem into three sub-problems and solves them through three interrelated PSOs named TLPSO. Then, the solutions obtained by TLPSO are enhanced by VNS. The key characteristics of TLPSO and VNS are employed simultaneously to achieve superior solutions in solving the addressed optimization problem. In the proposed hybrid algorithm, the TLPSO performs a global search whereas the VNS has a local search role. These two techniques complement each other to enhance the search diversification and intensification. Numerical experiments on a variety of simulated scenarios show the efficiency and effectiveness of the proposed algorithm by comparing it with some other algorithms.

Keywords: production scheduling, machine maintenance, mould maintenance, three-level particle swarm optimization, variable neighbourhood search

1. Introduction

Because of the special merits of plastics such as high strength, low price, light weight, 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, which needs both an injection machine and an injection mould. 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, production scheduling 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). Traditionally, it is postulated that machines are always available throughout the whole production planning stage in most research studies on production scheduling. While, 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 which integrate the production scheduling with machine maintenance planning have been built to optimize productivity by harmonizing both activities. However, only the maintenance of machine is considered in traditional production scheduling with the resource maintenance problem and research on production scheduling with the mould maintenance problem is limited. Since the injection mould is also a significant element in the plastics industry, the integrated problem with mould maintenance consideration should be given more attention.

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 genetic algorithm is limited. The mismatch between production scheduling and resource maintenance is underestimated, which may result in the low production efficiency. In addition, more efficient algorithms need to be explored to improve the quality of the solutions.

To overcome the shortcoming of previous research and improve the overall production efficiency, this research proposes a problem decomposition mechanism to deal with Production Scheduling with the Mould Maintenance (PS-MM) problem, and presents an effective and efficient hybrid metaheuristic algorithm: the TLPSO-VNS algorithm which innovatively combines Three-Level Particle Swarm Optimization (TLPSO) and Variable Neighbourhood Search (VNS). Firstly, this integrated problem is divided into a basic production scheduling problem, a machine maintenance problem and a mould maintenance problem. To minimize the overall makespan, the production scheduling problem is considered at first. The corresponding machine maintenance planning is decided after the production scheduling is determined. Once the production scheduling and machine maintenance are confirmed, the corresponding mould maintenance is determined. Every sub-problem is solved by one single particle swarm optimization (PSO) algorithm and these three PSOs are interrelated. Once good solutions are obtained by the Three-Level PSO (TLPSO), variable neighbourhood search (VNS) is applied to these solutions to conduct the local search. Seven types of neighbourhoods are designed, and they are changed systematically. The best solution is chosen as the final solution when all the processes finish. The proposed TLPSO-VNS algorithm is robust and can find high-quality solutions by effectively exchanging search intensification and diversification. To our knowledge, this algorithm is the first algorithm to hybridize TLPSO with VNS to solve the integrated problem, building on the advantages of these two individual metaheuristic components and overcoming the inherent limitations.

The reminder of this paper is organized as follows: Section 2 is a review of the literature on this topic, which provides an understanding of the previous research in this area, as well as providing a rationale for the choice of the topic in the present study. Section 3 restates the production scheduling with mould maintenance (PS-MM) problem. Section 4 proposes the TLPSO-VNS algorithm. Section 5 presents the computational results acquired and shows the superiority of the TLPSO-VNS algorithm. Section 6 provides the conclusions and suggestions for further research.

2. Literature review

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 improvement of the overall production performance (Guner, Chinnam, and Murat 2016). Traditionally, production scheduling and preventive maintenance planning decisions are made independently 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, Cassady and Kutanoglu (2003) proposed an integrated model which determines production scheduling and preventive maintenance planning decisions simultaneously, and numerical experiments showed the effectiveness of this integration. In addition, they (Cassady and Kutanoglu 2005) proposed an integrated model which harmonizes single-machine scheduling decisions and preventive maintenance planning decisions aiming at

minimizing the total expected weighted completion time of the job, 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 to minimize the makespan, and two meta-heuristic algorithms were proposed. Moreover, El Khoukhi, Boukachour, and Alaoui (2017) proposed two new models to formulate the Flexible Job Shop Scheduling Problem (FJSSP) with machine unavailability constraints due to Preventive Maintenance (PM), 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 problem of integrating production scheduling and maintenance planning.

Different maintenance strategies on machines and the effects on production scheduling are also explored by scholars. Ruiz, Garcia-Diaz, and Maroto (2007) studied different machine preventive maintenance policies regarding flowshop problems aiming at maximizing the availability of machines. They proposed a simple criterion to schedule different maintenance operations with the production sequence and showed the importance of the integration. Furthermore, Aghezzaf and Najid (2008) examined integrating the production and preventive maintenance problem in a system which was made up of parallel failure-prone production lines. They assumed that minimal repair is 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) proposed a double tier genetic algorithm approach for multi-factory production networks, which schedules perfect and imperfect maintenance on machines at the same time in order 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, uncertainty was also considered in the integrated problem. Wang and Liu (2016) studied an optimisation problem which integrated production scheduling and preventive maintenance (PM) in a two-machine flow shop aiming to minimize the makespan. Time to failure of each machine was assumed to be subjected to a Weibull probability distribution and four heuristics based on genetic algorithm (GA) were proposed. Furthermore, Abdelrahim and Vizvari (2017) considered the case in which random failures occurred on a single machine in the integrated problem. It is assumed that the probability of machine failure is an increasing function of the age and the length of the time interval, and they only considered perfect maintenance. The consideration of different maintenance strategies and uncertainty makes the integrated model more complicated but more in line with the actual situation.

However, in the plastics industry, the mould is also an important element that guarantees the normal production and needs to be considered. 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, Wong, Chan, and Chung (2013) considered the 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, with the objective of minimizing the total weighted tardiness and earliness. Different maintenance strategies were taken into account and a genetic algorithm was used to find the optimal solution. Integrating production scheduling with mould maintenance is still a key issue for some industries, but research on this topic is still limited and more efficient algorithms for solving such an integrated problem need to be developed.

As an intelligent optimization algorithm inspired by the social behaviour of animals, Particle Swarm Optimization (PSO) was firstly proposed by Kennedy and Eberhart (Kennedy and Eberhart 1995). 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 such as the Similar Particle Swarm Optimization Algorithm (SPSOA) (Lian, Gu, and Jiao 2006), the Novel Particle Swarm Optimization (NPSO) algorithm (Lian, Gu, and Jiao 2008), and the Improved Particle Swarm Optimization (IPSO) algorithm (Tang and Wang 2010). Furthermore, the Variable Neighbourhood Search (VNS) algorithm was proposed by Mladenović and Hansen (Mladenović and Hansen 1997), which was an effective and simple metaheuristic algorithm based on the systematic change of neighbourhoods. It can be used for many production scheduling problems, such as the identical parallel machine scheduling problem with two conflicting objectives (Liang and Tien 2011), the flexible job-shop scheduling problem (FJSP) (Bagheri and Zandieh 2011), and the identical parallel machine scheduling problems (Bathrinath et al. 2015). Moreover, some new algorithms based on VNS were also developed, such as the Population-based Variable Neighbourhood Search (PVNS) algorithm (Mokhtari, Mozdgir, and Abadi 2012), the Improved Variable Neighbourhood Search (IVNS) algorithm (Lan et al. 2016), and the General Variable Neighbourhood Search (GVNS) (Komaki and Malakooti 2017). So far, PSO and VNS are very popular and efficient algorithms in dealing with production scheduling problems.

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 1.

Table 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, 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)

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.

3. Problem description

3.1 Problem description

The production scheduling with mould maintenance (PS-MM) problem can be described as follows: P jobs are distributed on Q injection machines and R injection moulds. Each problem is denoted as $P*Q*R$. 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. Each job can only be performed on one machine with one mould at a time slot. Each machine can only conduct one job with one machine at a time slot. Each mould can only carry out one job on one machine at a time slot. The unit operation time of a job is decided by the specific mould being used but not the machine. Each job has its batch size 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 once a job begins, it should be finished without interruption. The cumulated operating time of a resource is defined as the resource age (not including idle time). In the practical situation, the earlier the maintenance is conducted, the less maintenance time is needed, 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 (NA) is shorter than the maximum age of the machine (MA). Once a resource reaches its maximum age, maintenance should

be conducted after the 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 the preventive maintenance can prevent all the random breakdowns of the resources. Furthermore, we assume that set-up times are sequence-independent and included in 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 Mathematical model of the problem

This problem is described as an integer programming model (Wong, Chan, and Chung 2012) to find optimal solutions. Based on the model of Wong, Chan, and Chun （2012），a new model is built and used as the further step to understand the problem. Table 2 lists all the notations used in this paper.

Table 2: Notations list

Index	Descriptions
P:	Number of jobs.
Q:	Number of machines.
R:	Number of moulds
p:	Index for job, $p = 1, \cdots P$, where P is the number of jobs.
q_p:	Index for machine, $q_p = 1, \cdots Q$, where Q is the number of machines. q_p is the machine used by job p. $S(p)$ is a set that contains all the available machines for job p, then $q_p \in S(p)$.

r_p :	Index for mould, $r_p = 1, \cdots R$, where R is the number of moulds. r_p is the specific mould for job p.
t :	Index for time slot, $t = 1, \cdots T$, where T is the maximum time horizon.
$T_{pq_pr_p}$:	Operating time of job p on machine q_p with mould r_p.
MA:	Maximum machine age.
NA:	Maximum mould age.
S_P :	Starting time of job p.
C_P :	Completion time of job p.
$C_{\max}$:	Makespan of jobs.
$X_{pq_pr_p}$:	=1, if job p is allocated on machine q_p with mould r_p. =0, otherwise.
$Y_{pq_pr_pt}$:	=1, if job p occupies time slot t, on machine q_p with mould r_p. =0, otherwise.
$Z^1_{pr_pkr_kq_p}$:	=1, if job p (with its specific mould r_p) is sequenced after job k (with its specific mould r_k) on machine q_p. =0, otherwise.
$Z^2_{pq_psq_sr_p}$:	=1, if job p (with its machine q_p)is sequenced after job s (with its machine q_s) on mould r_p. =0, otherwise.
$M^1_{kq_pr_kt}$:	=1, if machine q_p is maintained after job k (with mould r_k) and the maintenance occupies time slot t. =0, otherwise.
$M^2_{sq_sr_pt}$:	=1, if mould r_p is maintained after job s (with machine q_s) and the maintenance occupies time slot t. =0, otherwise.

In this problem, the number of jobs (P), the number of machines (Q), the number of moulds (R) and the operating time of job p on machine q_p with mould r_p ($T_{pq_pr_p}$) are known in advance. $X_{pq_pr_p}$, $Y_{pq_pr_pt}$, $Z^1_{pr_pkr_kq_p}$, $Z^2_{pq_psq_sr_p}$, $M^1_{kq_pr_kt}$ and $M^2_{sq_sr_pt}$ are

decision variables. Once they are decided, S_P and C_P can be calculated and then we can obtain the $C_{\max}$. The objective is to minimise the makespan of jobs and the objective function is as follows:

$$f = \min(C_{\max}) \tag{1}$$

The problem is subjected to the following constraints:

Processing time constraints:

$$C_p - S_p = \sum_{q_P r_P} X_{pq_p r_p} T_{pq_p r_p} \qquad (p = 1,2,\cdots,P) \tag{2}$$

$$\sum_{q_P r_P t} Y_{pq_p r_p t} = \sum_{q_P r_P} X_{pq_p r_p} T_{pq_p r_p} \qquad (p = 1,2,\cdots,P) \tag{3}$$

Job constraints:

$$\sum_{q_P r_P} X_{pq_p r_p} = 1 \qquad (p = 1,2,\cdots,P) \tag{4}$$

Processing job constraints:

$$\sum_{q_P r_P} Y_{pq_p r_p t} \le 1 \qquad (p = 1,2,\cdots P; t = 1,2,\cdots T) \tag{5}$$

Job sequence constraints:

$$Z^1_{pr_p kr_k q_p} \cdot X_{pq_p r_p} \cdot X_{kq_p r_k} = 1 \ (p = 1,2,\cdots P; k = 1,2,\cdots P, k \neq p, q_p = 1,2,\cdots Q) \tag{6}$$

$$Z^2_{pq_p sq_s r_p} \cdot X_{pq_p r_p} \cdot X_{sq_s r_p} = 1 \ (p = 1,2,\cdots P; k = 1,2,\cdots P, s \neq p, r_p = 1,2,\cdots R) \tag{7}$$

Machine maintenance constraints:

$$\sum_{pr_p} Y_{pq_p r_p t} + \sum_{kr_k} M^{1}_{kq_p r_k t} \le 1 \qquad (q_p = 1,2,\cdots Q; t = 1,2,\cdots T) \tag{8}$$

Mould maintenance constraints:

$$\sum_{pq_p} Y_{pq_p r_p t} + \sum_{sq_s} M^{2}_{sq_s r_p t} \le 1 \qquad (r_p = 1,2,\cdots R; t = 1,2,\cdots T) \tag{9}$$

In this model, Constraint (2) ensures that there is no interruption in the production processing of a job. Constraint (3) ensures that the allocation of the time slot equals the required operation time. Constraint (4) ensures that only one job can be carried out on one machine with one mould. Constraint (5) ensures that one job can only be carried out by on one machine with one mould. Constraint (6) ensures the job sequence on the same machine. Constraint (7) ensures the job sequence on the same mould. Constraint (8) ensures that the maintenance and the job processing on the same machine cannot happen in the same time slot. Constraint (9) ensures that the maintenance and the job processing on the same mould cannot happen at the same time slot.

4. Optimization methodology

The optimization algorithm, named the TLPSO-VNS algorithm, is introduced in this part. The overall algorithm structure is introduced firstly. The overall algorithm includes two stages: the stage of swarm initialization and swarm improvement by TLPSO and 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.

4.1 Overall algorithm description

For this integrated scheduling problem, we divide it into three subproblems: production scheduling problem, machine maintenance problem and 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 1. shows the process of the problem analysis and the design of the overall algorithm.

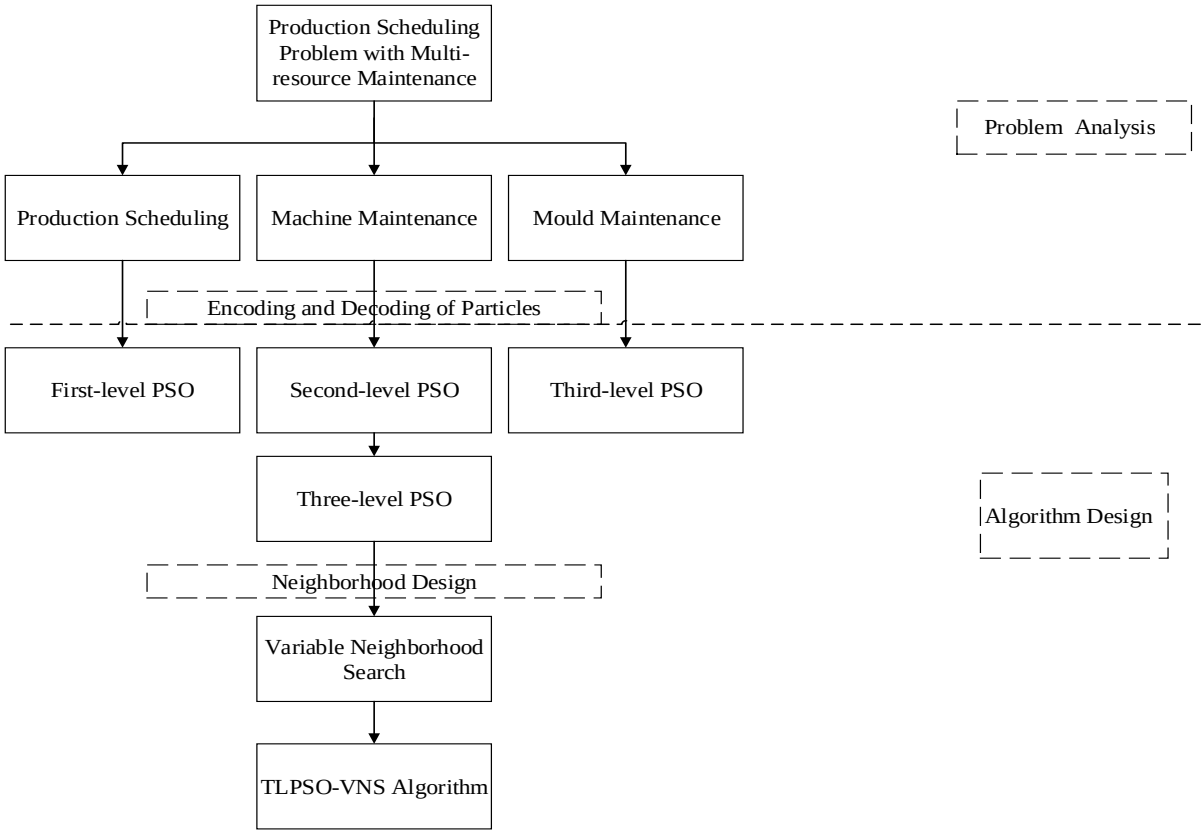

Figure 1． Framework of the overall algorithm

4.2 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, whose dimension is $2*P$ (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,

whose dimension is $3*P$. 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, whose dimension is $4*P$. 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 2- Figure 4 show examples of JM- Particle, JMAM-Particle, JMAMOM-Particle before and after decoding. There are 5 jobs, 3 machines and 2 moulds in this example. Mould 1 can be used to produce Job 1, 3, 5 and Mould 2 can be used to produce Job 2, 4. From Figure 4, 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 position 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) 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 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, 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, 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, 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 because the corresponding OM parameter is 0.

To illustrate the influence of the preventive maintenance on the value of the objective function, the Gant charts of the example in Figure 2 (without resource maintenance consideration) and the Gant charts of the example in Figure 4 (with resource maintenance consideration) are shown in Figure 5 and Figure 6. In the Gant charts, MT means maintenance. The numbers in the brackets are the processing time that each job needs and the maintenance time that the resource needs. Since the beginning time of a job is decided by the available time of the machine and the mould that the job use, 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. But 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 can know that the makespan is changed from 125 units of time to 180 units of time because of the influence of maintenance on both machine and mould.

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 2. 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. 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 4. Encoding and decoding of JMAMOM-particle

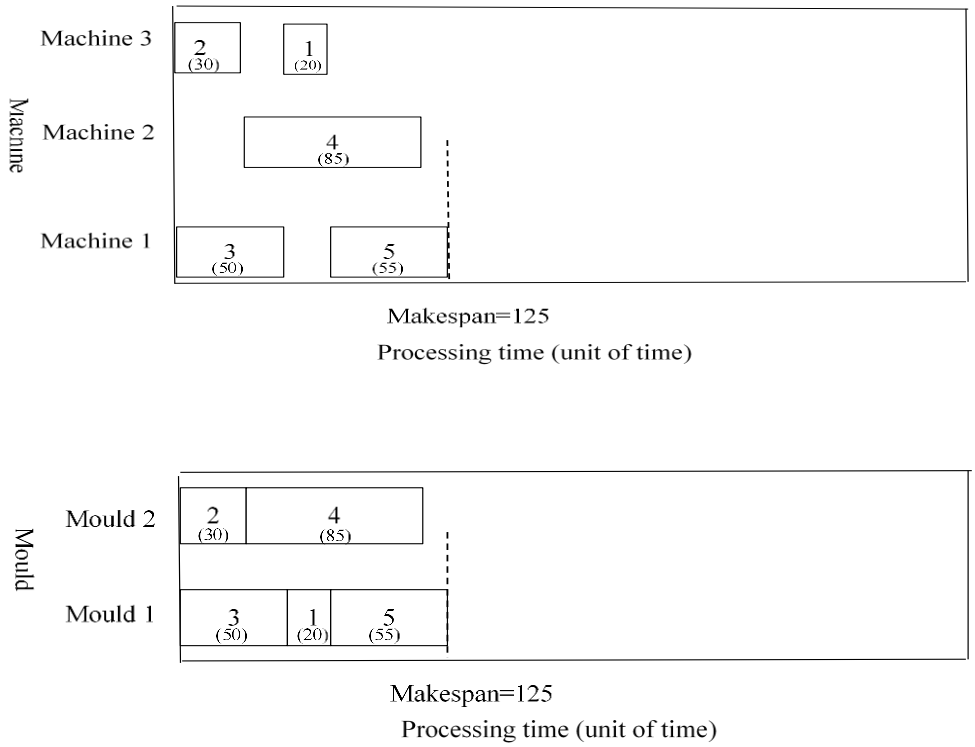

Figure 5. Gant charts of the example without maintenance

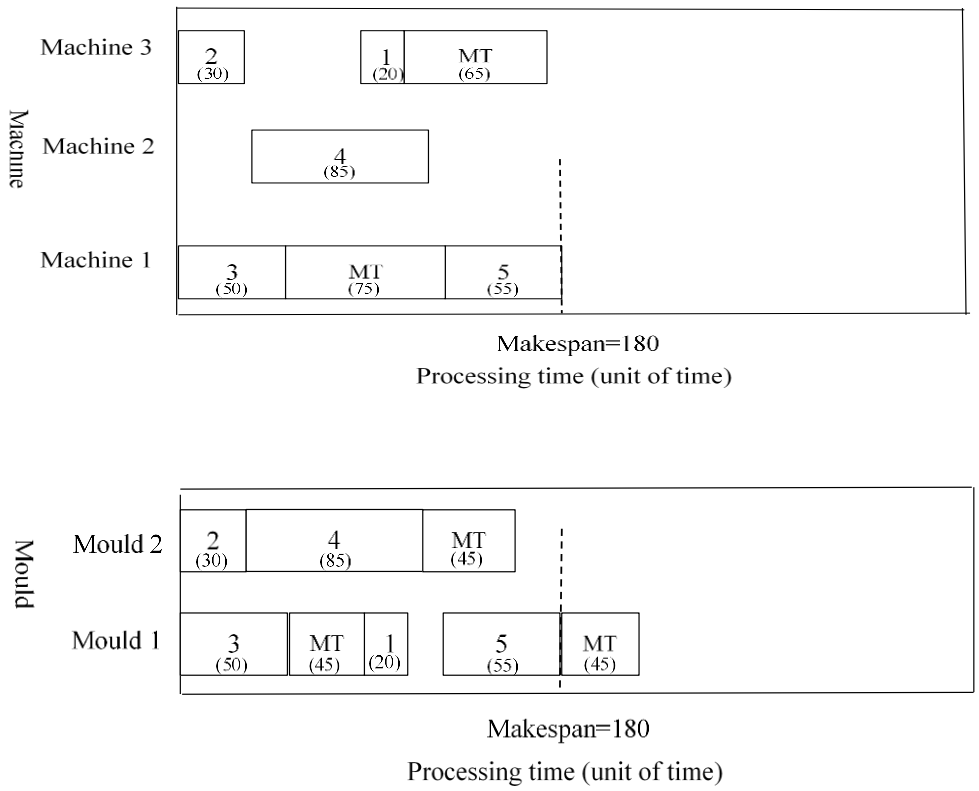

Figure 6. Gant charts of the example with maintenance

4.3 Swarm initialization and swarm improvement through Three-Level PSO

The TLPSO algorithm consists of three levels, hereafter called 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 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 $2 * P$ (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-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 $3*P$ dimensions. The first $2*P$ 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 are produced randomly, and the first $2*P$ 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)-Step2b(v)** will be conducted until the stopping condition is reached, and the best JMAMOM-Particle will be recorded.

3rd level PSO

Step2b(i): Initialization. For each JMAM-Particle, initialize a population of JMAMOM-Particles with $4*P$ dimensions. The first $3*P$ 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 are produced, and the first $3*P$ 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. 10 and then the position is updated according to Equation. 11 based on the velocity from Equation 10. Finally, inertia factor W is updated according to Equation. 12. 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. $P_{ij}(r_3)$ 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.

$$V_{ij}(r_3+1) = W \times V_{ij}(r_3) + C_1 \times R_1 \times (P_{ij}(r_3) - X_{ij}(r_3)) + C_2 \times R_2 \times (P_{gj}(r_3) - X_{ij}(r_3)) \quad (10)$$

$$X_{ij}(r_3+1) = X_{ij}(r_3) + V_{ij}(r_3+1) \quad (11)$$

$$W = W_{\max} - \frac{W_{\max} - W_{\min}}{r_{3\max}} \times r_3 \tag{12}$$

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 $r_3 <= r_{3\max}$, 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. 10 and then the position is updated according to Equation. 11 based on the velocity obtained from Equation 10. Finally, inertia factor W is updated according to Equation. 12. 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, $P2_{ibset}$ 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 $r_2 <= r_{2\max}$, then go to Step 2e, start a new iteration.; otherwise, terminate the second level PSO and calculate 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. 10 and then the position is updated according to Equation. 11, Finally inertia factor W is updated according to Equation. 12. 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_{ibset} 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 $r_1 <= r_{1\max}$, 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 7 shows the flowchart of TLPSO. In this flowchart, an example of 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, 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. 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 0.5 0.3 0.35 0.6 0.8); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.4 0.6 0.1 0.2 0.9); (0.1 0.8 0.17 0.2 0.4 0.3 0.8 0.9 0.3 0.6 0.6 0.7 0.2 0.7 0.1). The first 10 dimensions of these three JMAM-particles are the same with 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, 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 with 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 JMAMOM-particles are decoded into suitable scheduling solutions. The fitness value (makespan) of each JMAMOM-particle can be calculated. 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 until the stopping condition is met. 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 10).

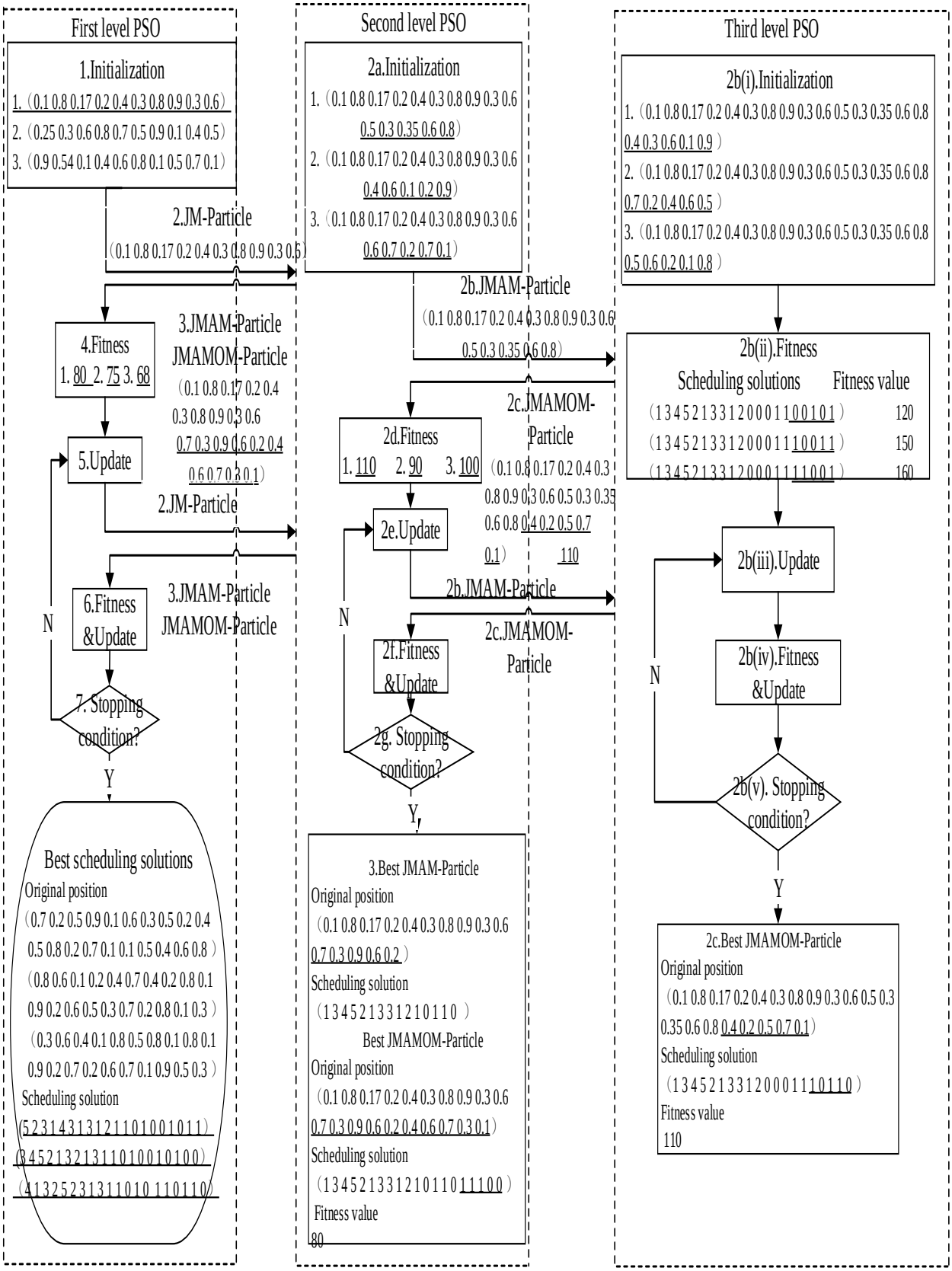

Figure 7. Flowchart of TLPSO

4.4 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, 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, n are random integers in [1, P] (P is the number of jobs) and m is smaller than n. U is a scheduling solution obtained from TLPSO. The definitions of these neighbourhoods are given as follows:

1. The first neighbourhood search is to change the $(m+3*P)^{th}$ dimension of scheduling solution from 0 to 1 or from 1 to 0 (change the mould maintenance of job m), which is abbreviated as Change (U, $m+3*P$).

Original scheduling solution:																			
5	2	3	1	4	3	1	3	1	2	1	1	0	1	0	0	1	0	1	1
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM
The first neighborhood of the Original scheduling solution:																			
5	2	3	1	4	3	1	3	1	2	1	1	0	1	0	0	0	0	1	1
J	J	J	J	J	M	M	M	M	M	AM	AM	AM	AM	AM	OM	OM	OM	OM	OM

Figure 8. The first neighbourhood

2. The second neighbourhood search is to insert the $(m+3*P)^{th}$ dimension of scheduling solution to the $(n+3*P)^{th}$ dimension (change the mould maintenance from job m to job n), which is abbreviated as Insert (U, $m+3*P$, $n+3*P$).

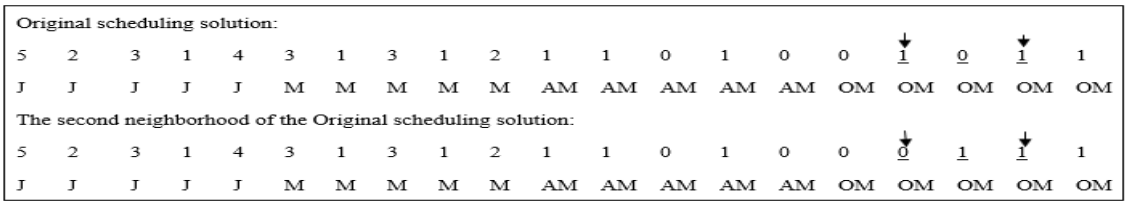

Figure 9. The second neighbourhood

3. The third neighbourhood search is to change the $(m+2*P)^{th}$ dimension of scheduling solution from 0 to 1 or from 1 to 0 (change the machine maintenance of job m), which is abbreviated as Change (U, $m+2*P$)

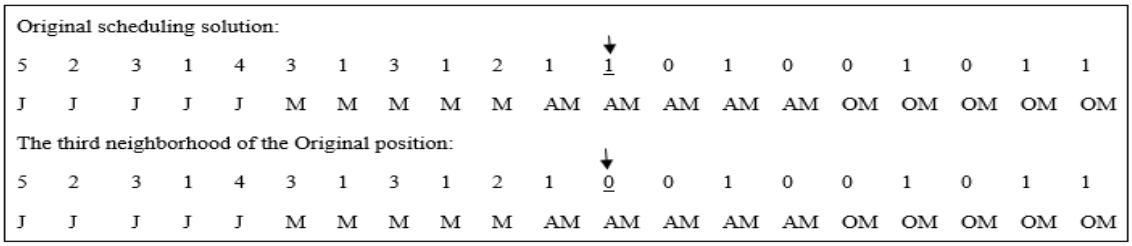

Figure 10. The second neighbourhood

4. The fourth neighbourhood search is to insert the $(m+2*P)^{th}$ dimension of scheduling solution to the $(n+2*P)^{th}$ dimension (change the machine maintenance from job m to job n), which is abbreviated as Insert (U, $m+2*P$, $n+2*P$)

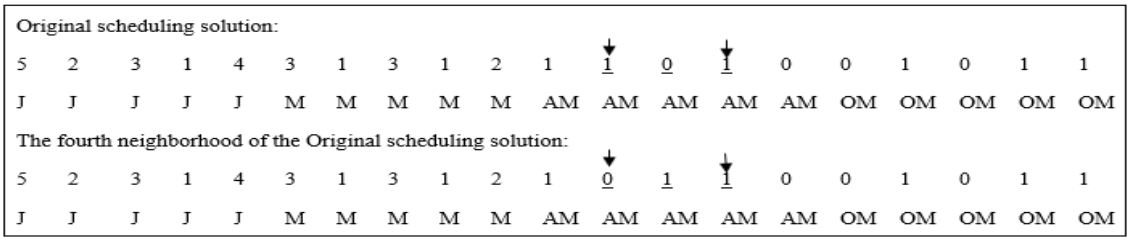

Figure 11 The fourth neighbourhood

5. The fifth neighbourhood search is to exchange the m^{th} dimension of scheduling solution with n^{th} dimension (change the job sequence of job m and job n), which is abbreviated as Exchange (U, m, n)

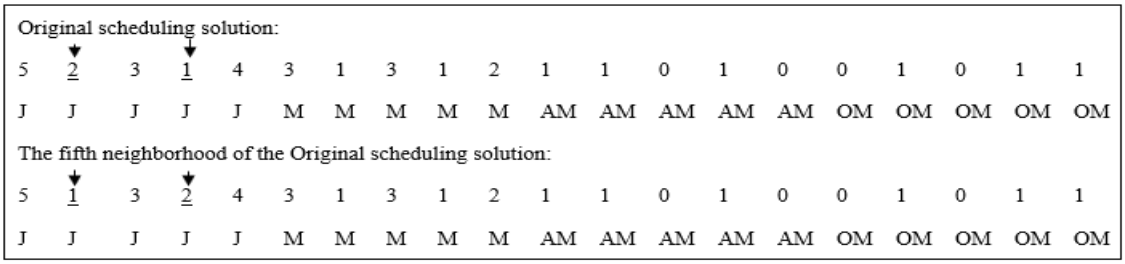

Figure 12 The fifth neighbourhood

6. The sixth neighbourhood search is to insert the m^{th} dimension of scheduling solution to n^{th} dimension (change the job sequence from job m to job n), which is abbreviated as Insert (U, m, n)

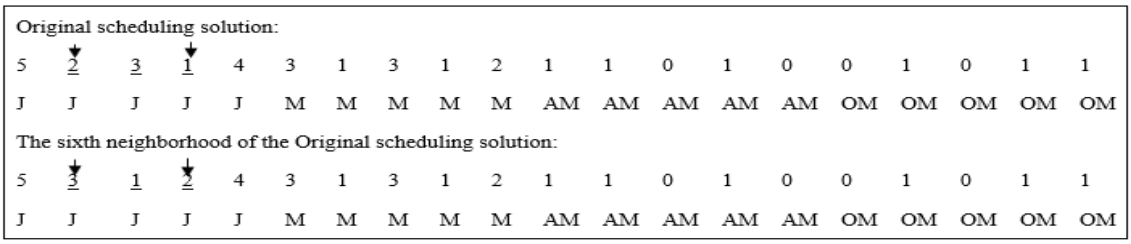

Figure 13. The sixth neighbourhood

7. The seventh neighbourhood search is to change the $(m+P)^{th}$ dimension to an available number (change machine of job m to another available machine), which is abbreviated as Changemachine (U, m)

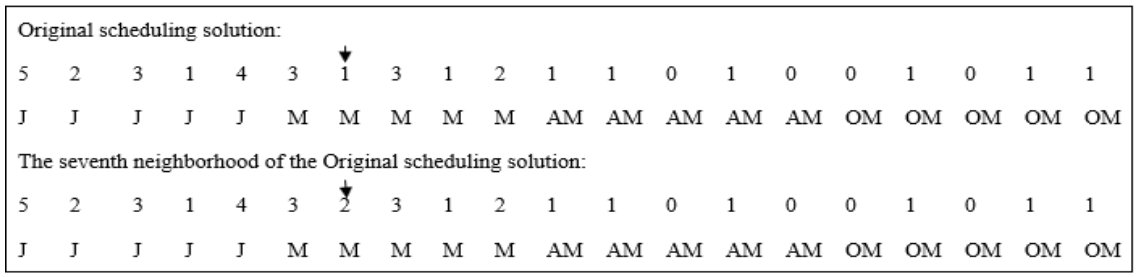

Figure 14. The seventh neighbourhood

The first scheduling solutions obtained from TLPSO is taken as an example to illustrate these neighbourhoods. Assuming $m=2, n=4$, examples of these neighbourhoods are shown in Figure 8-Figure 14. The pseudo-code of the local search is

given in Figure 15. After all the solutions are finished in the processing of VNS, the solution with the minimum fitness is chosen as the final solution.

```
S0=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 {S1=Change (S, m+3*P)}
      If(count==1) then {S1=Insert (S, m+3*P, n+3*P)}
      If(count==2) then {S1= Change (S, m+2*P)}
      If(count==3) then {S1=Insert (S, m+2*P, n+2*P)}
      If(count==4) then {S1=Exchange (S, m, n)}
      If(count==5) then {S1=Insert (S, m, n)}
      If(count==6) then {S1=Changemachine (S, m)
      If (f(S1) <f(S)) then {
      Count=0;
      S=S1
      }
      Else{count++}
      } while (count<Max)
Loop++
} while loop<=P*(P-1)
If f(S)<=f(U) then {
U=S
}
```

Figure 15. pseudo-code of VNS

5. 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 resources and three datasets generated by Wong, Chan, and Chung (2012) are adopted. The sizes of these three problems are (30*3*5), (40*6*10) and (60*9*15). 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, three PSO varients (ABCSPSO, fkPSO, SPSO2011) proposed in the CEC'2013 competitions are used as the comparison algorithms to further illustrate the performance of the TLPSO-VNS algorithm. Six random instances are used and the Wilcoxon test is used to analysis the comparison results.

5.1 Parameters tuning

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.

There are two kinds of parameters. One kind is the key algorithm parameters, including the swarm size, maximum number of generation. 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 in the literature related to PSO. The swarm size and the maximum number of generation are decided. Then, the key algorithm parameters are fixed, we try different parameters combinations related to PSO and find the best parameters combination.

Since 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. According to Mladenović and Hansen (1997), the Max Iteration of VNS is set as $P*(P-1)$ (P is the number of jobs), which is sufficient to obtain a steady solution for the largest problem. 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 is the most often used by the literature related to PSO. In the preliminary testing about 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*6*10) generated by Wong et al. (2012) 10 times. There are 9 combinations in total. Table 3 shows the different parameters combinations, the corresponding average results and the time needed for each run. From Table 3, we can see that the swarm size has great influence on the time needed for each run. When the swarm size increases by 2 times (from 5 to 10), the time needed for each run increase by more than 5 times. When the swarm size increases by 3 times (from 5 to 15), the time needed for each run increase 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 quality of results and the time needed for each run, we choose the parameter combination{10, 1000}.

Table 3. Different Key algorithm parameters combinations and its 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,

Halgamuge, and Watson 2004); Clerc's constriction method (Eberhart and Shi 2000). The maintenance scheme of resources and the medium-sized instance (40*6*10) generated by Wong et al. (2012) is used and we test each parameters combination 10 times. The results are shown in table 4. From Table 4, we can see that there is little difference between these parameters combinations, which means that the PSO related parameters have little influence on the results. Finally, we use the parameters combination recommended by Eberhart and Shi (2000) to solve this problem. In our random test, the average value and the standard deviation are the minimum compared with other parameters combinations. But other parameters combination are also able to produce good results for this problem.

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

No.	Parameters in 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

5.2 Comparison with results in existing literature

The maintenance scheme of resources and three datasets with different sizes generated by Wong et al. (2012) are adopted. The comparison results are shown in Table 5.

Table 5. Comparison with results in the existing literature.

	30*3*5			40*6*10			60*9*15		
	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 5, 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 by NSGA-II, the improvement is between 0.19% and 6%. Moreover, the improvement increases as the problem size increases. It can be concluded that TLPSO-VNS can provide better solutions than GADG and NSGA-II when the size of the problem increases. 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 can get better minimum solutions than GADG and NSGA-II, except for the minimum result for the 30*3*5 instance. But the gap between the minimum results obtained by TLPSO-VNS and NSGA-II is really small, only 0.7%. Furthermore, the TLPSO-VNS algorithm is more robust than GADG and NSGA-II because that the standard deviations of the TLPSO-VNS algorithm are smaller for all these three instances with different sizes. The details of the production scheduling for the optimal results of the three instances are shown in Figure 16 and Figure 17. Specifically, MT means maintenance, the numbers in the box are the sequence of jobs.

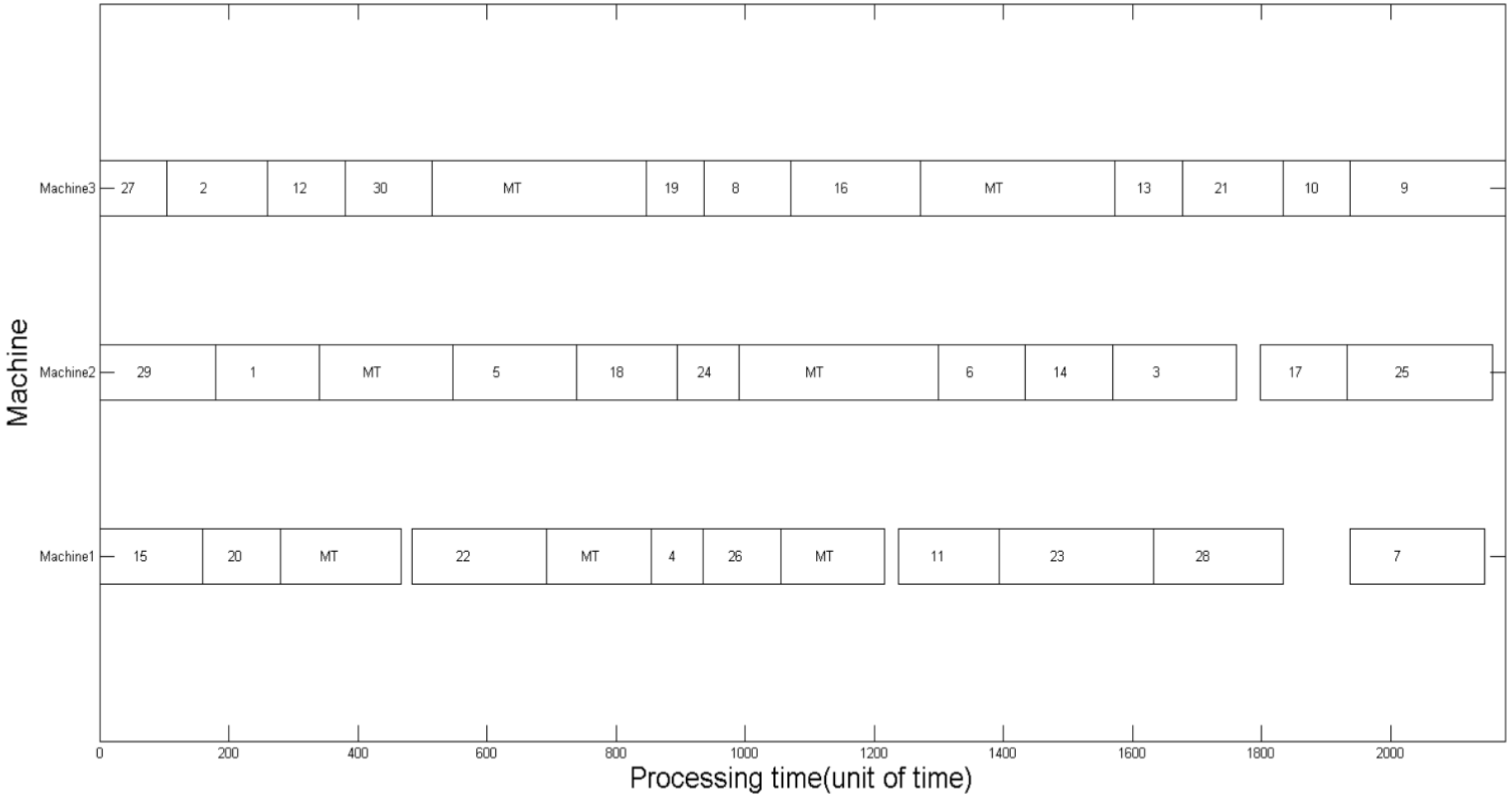

Figure 16 Machine schedule for instance 1

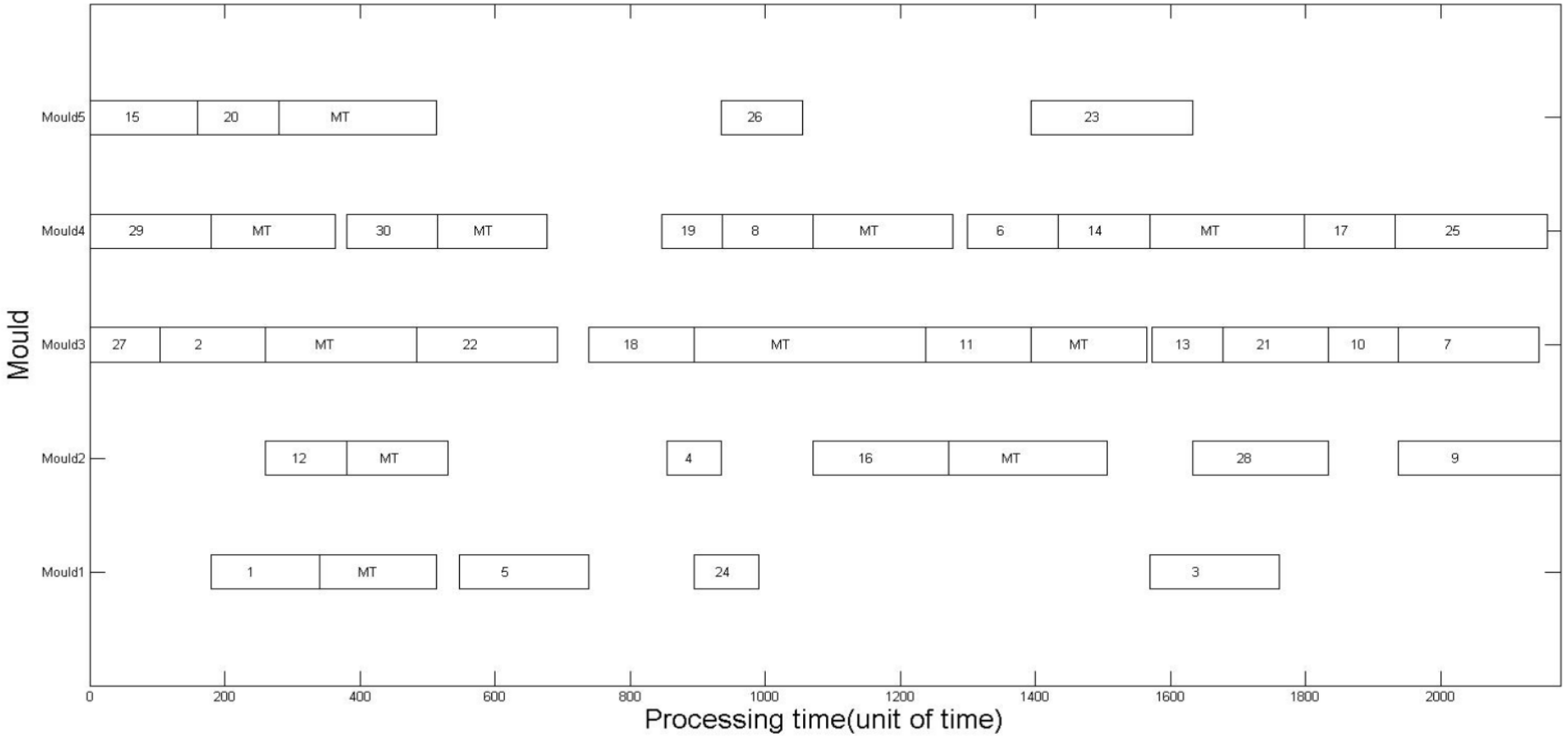

Figure 17 Mould schedule for instance 1

5.3 Comparison with PSO variants in CEC'2013 Competition

There are three PSO variants in CEC'2013: ABCSPSO (El-Abd 2013), fk-PSO (Nepomuceno and Engelbrecht 2013), and SPSO2011 (Zambrano-Bigiarini, Clerc, and Rojas 2013). ABCSPSO is also a hybrid algorithm where the PSO algorithm is augmented with an ABC component to improve the personal best of the particles. In every iteration, a new candidate solution NS is produced for each particle using the ABC update equation. The new candidate solution NS replaces the individual best if it has a better objective function value. fk-PSO is a self-adaptive heterogeneous PSO, which automatically alters its exploration and exploitation balance. The fk-PSO consists of six kinds of behaviors: TVAC-PSO for early exploration and later exploitation; TVIW-PSO for early exploration and later exploitation; sPSO for its high exploitation ability; cPSO for its hill-climbing ability; modBB-PSO with the exploitation probability set to linearly increase from 0 to 1; QSO with the cloud radius set to linearly decrease from the search bounds to 0 for better initial exploration. The particles in the fk-PSO change behaviors when they stagnant, which means that their individual bests do not improve for a number of iterations. Each behavior is conducted k iterations. In our experiment, the sequence of these behaviors is given instead of using tournament selection to decide the next behavior. SPSO2011 exploits the idea of rotational invariance. For each particle and at each time step, a center of gravity $(\vec{G}_i)$ is defined by three points: the current position, a point a little "beyond" the best previous personal position, and a point a little "beyond" the best previous position in the neighourhood. A random point is defined in the hypersphere

whose center is gravity ($\vec{G}_i$) and the radius is the distance between the gravity ($\vec{G}_i$) and the current position. Furthermore, a new velocity equation is given.

Six random instances with different sizes are used to test the performance of the proposed TLPSOVNS. Based on the recommendation of Wong et al. (2012), the parameters of the six random instances are set as follows: the operation time of the 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 maintenance scheme of resources generated by Wong et al. (2012) is adopted. For a fair comparison, algorithm parameters, including the parameters related to PSO, the population size, the maximum number of generation of these three algorithms are the same as the proposed TLPSOVNS. For the parameters of fk-PSO, the parameters turning process similar to the process in section 5.1 is conducted. Finally, the iteration parameter k is set as 1000, and stagnant parameter is set as 20. The comparison results are listed in Table 6.

Table 6. Comparison results with PSO Variants.

N0	Size	Algorithm	Min	Max	Avg	SD	Time
1	20*2*4	TLPSOVNS	2308	2436	2383.3	32	939
		ABCSPSO	2364	2811	2613.8	133	302
		fk-PSO	2640	2935	2748.6	80	1.6
		SPSO2011	2383	2618	2517.8	74	10.7
2	35*4*6	TLPSOVNS	2931	3380	3108.7	119	1112
		ABCSPSO	3029	3827	3545	251	350
		fk-PSO	3874	4707	4234.6	316	3
		SPSO2011	3273	4216	3872.6	276	23
3	50*7*10	TLPSOVNS	2598	2836	2680.9	96	1026
		ABCSPSO	2829	3391	3036.6	169	956
		fk-PSO	3695	4157	3893	138	4
		SPSO2011	3329	3717	3483.4	121	99
4	70*9*12	TLPSOVNS	3672	3988	3847.6	83	3302
		ABCSPSO	3905	4336	4128	149	1061
		fk-PSO	4422	4423	4892	229	5
		SPSO2011	4615	5073	4830	143	359
5	80*10*16	TLPSOVNS	3200	3355	3279	46	3728
		ABCSPSO	3634	4028	3866.0	143	1184
		fk-PSO	4563	5351	5048	229	6
		SPSO2011	4617	5022	4787	142	40
6	100*12*20	TLPSOVNS	3776.3	3997.5	3906.0	60	4626
		ABCSPSO	3942.0	4762.3	4368.0	250	1469
		fk-PSO	5523.7	6101.5	6004.7	269	8
		SPSO2011	5224.8	6433.5	5729.5	264	47.2

The Wilcoxon Signed Rank Test is used to deeply analysis the difference between the TLPSO-VNS and the other three PSO variants. The analysis results of these six instances are shown in table 7. The significant level is 0.05. Particularly, the Wilcoxon Test results for the instance 100*20*12 is given in Table 8-Table 10 as an example.

Table 7. Wilcoxon Signed Rank Test between TLPSO-VNS and PSO variants

N0	Size	Algorithm Comparison			P value
1	20*2*4	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	ABCSPSO	0.005
		TLPSO-VNS	VS	fk-PSO	0.005
		TLPSO-VNS	VS	SPSO2011	0.007
3	50*7*10	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	ABCSPSO	0.005
		TLPSO-VNS	VS	fk-PSO	0.005
		TLPSO-VNS	VS	SPSO2011	0.005
5	80*10*16	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	ABCSPSO	0.005
		TLPSO-VNS	VS	fk-PSO	0.005
		TLPSO-VNS	VS	SPSO2011	0.005

Table 8. Wilcoxon Test of instance 100*12*20 between TLPSOVNS and ABCSPSO

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between TLPSOVNS and ABCSPSO equals 0.	Related-Samples Wilcoxon Signed Rank Test	.005	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Table 9. Wilcoxon Test of instance 100*12*20 between TLPSO-VNS and SPSO2011

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between TLPSOVNS and SPSO2011 equals 0.	Related-Samples Wilcoxon Signed Rank Test	.005	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Table 10.Wilcoxon test of instance 100*12*20 between TLPSO-VNS and fkPSO

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between TLPSOVNS and fkPSO equals 0.	Related-Samples Wilcoxon Signed Rank Test	.005	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

From Table 6, we can know that TLPSO-VNS surpasses ABCSPSO, fk-PSO, and SPSO2011 for these six instances in terms of the average value, the minimum value and the maximum value. Table 7 shows that there are differences between TLPSO-VNS and other PSO variants because all the P values are smaller than the significant level 0.05.

Since the TLPSO-VNS consists of three interrelated PSO, the local search ability is greatly improved but it needs more time to conduct the deep search. VNS, as a good search tool, can produce relative good solutions, however, VNS needs more time when the size of the problem increases. 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.

6. Conclusions

This paper 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 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 TLPSOVNS is shown to surpass all other PSO variants in the CEC'2013 competition.

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 suitable for production scheduling with mould maintenance problem with multi-objectives in the future.

Acknowledgements

The work described in this paper was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 15201414); The Natural Science Foundation of China (Grant No. 71471158); and under student account code RUKH.

References

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

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

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

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

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

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

Christopher, P. 2006. *Production Scheduling*: Collins.

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.

Eberhart, R. C., and Shi, Y. 2000. "Comparing inertia weights and constriction factors in particle swarm optimization." 1, 84-88. doi:10.1109/CEC.2000.870279

El-Abd, M. 2013. "Testing a Particle Swarm Optimization and Artificial Bee Colony Hybrid algorithm on the CEC13 benchmarks." *Evolutionary Computation (CEC), 2013 IEEE Congress on Evolutionary Computation*, 2215-2220. doi: 10.1109/CEC.2013.6557832

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

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

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). doi:10.1142/s1793962316500240

Kennedy, J., and Eberhart, R. (1995). "Particle swarm optimization." 4, 1942-1948. doi: 10.1109/ICNN.1995.488968

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. M., Wang, B., and Zhang, X. X.. 2016. "Improved Variable Neighbourhood Search Algorithm for Robust Job Shop Scheduling Problems" *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics*. 592-595

doi : 10.1109/IHMSC.2016.246

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

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 & 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." *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6838, 55-62.*

doi: https://doi.org/10.1007/978-3-642-24728-6_8

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 & Industrial Engineering, 52*(4), 404-413. doi:10.1016/j.cie.2007.01.004

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

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

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., and Engelbrecht, A. 2013. "A self-adaptive heterogeneous pso for real-parameter optimization." *Evolutionary Computation (CEC), 2013 IEEE Congress on Evolutionary Computation*, 361-368.
doi: 10.1109/CEC.2013.6557592

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

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

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." *Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, 230*(5), 934-953. doi:10.1177/0954405414563554

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. B., Gao, S., and Fang, J. 2016. "Production Scheduling with Mould Maintenance in Flow Shop." In P. Yarlagadda (Ed.), *Proceedings of the 2016 4th International Conference on Sensors, Mechatronics and Automation*. Zhuhai,China

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

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. 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

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 & Industrial Engineering, 102*, 99-112. doi:10.1016/j.cie.2016.10.015

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

Zambrano-Bigiarini, M., Clerc, M., and Rojas, R., 2013. "Standard Particle Swarm Optimisation 2011 at CEC-2013: A baseline for future PSO improvements." *Evolutionary Computation (CEC), 2013 IEEE Congress on Evolutionary Computation*, 2337-2344. doi: 10.1109/CEC.2013.6557848