[Egyptian Informatics Journal 24 (2023) 100383](https://doi.org/10.1016/j.eij.2023.05.008)

Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/11108665)

Egyptian Informatics Journal

journal homepage: [www.sciencedirect.com](http://www.sciencedirect.com/)

Full length article

[](http://crossmark.crossref.org/dialog/?doi=10.1016/j.eij.2023.05.008&domain=pdf)Minimizing tardiness and makespan for distributed heterogeneous unrelated parallel machine scheduling by knowledge and Pareto-based memetic algorithm

Hua Wang [a](#_bookmark0), Rui Li [b](#_bookmark1), Wenyin Gong [b](#_bookmark1),[⇑](#_bookmark2)

a *School of Mechanical and Power Engineering, Nanjing Tech University, Nanjing 211816, China*

b *School of Computer Science, China University of Geosciences, Wuhan 430074, China*

a r t i c l e i n f o

*Article history:*

Received 7 April 2023

Revised 28 April 2023

Accepted 12 May 2023

Available online 29 May 2023

*Keywords:*

Distributed heterogeneous factory Unrelated parallel machine scheduling Memetic algorithm

Knowledge-based heuristic strategies Multi-objective optimization

a b s t r a c t

This work aims to deal with the distributed heterogeneous unrelated parallel machine scheduling prob- lem (DHUPMSP) with minimizing total tardiness (TDD) and makespan. To solve this complex combina- torial optimization problem, this work proposed a knowledge and Pareto-based memetic algorithm (KPMA) which contains the following features: 1) four heuristic rules are designed including the shortest processing time rule, the minimum factory workload rule, the minimum machine finish time rule, and the earliest due date rule. Meanwhile, a hybrid heuristic initialization is developed to construct a popu- lation with great convergence and diversity; 2) four problem feature-based heuristic neighborhood struc- tures are designed to increase the success rate of local search; and 3) a simple elite strategy is developed to enhance the usage of historical elite solutions. Finally, to evaluate the performance of KMPA, it is com- pared to five state-of-art and run on 20 instances with different scales. The results of numerical experi- ments show that the proposed hybrid heuristic initialization can efficiently save computation resources to improve the initialized convergence. In addition, the knowledge-based neighborhood structures can vastly accelerate exploration. Moreover, the elite strategy can efficiently improve the diversity of the final non-dominated solutions set. The proposed KPMA has better performance than the state-of-art and has a strong ability to solve DHUPMSP.

© 2023 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Computers and Artificial Intel-

ligence, Cairo University. This is an open access article under the CC BY-NC-ND license ([http://creative-](http://creativecommons.org/licenses/by-nc-nd/4.0/)

[commons.org/licenses/by-nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

1. Introduction

Nowadays, traditional manufacturing is emergent to update and transfer the artificial production scheduling method to intelli- gent advanced planning and scheduling (APS) system [[1,2]](#_bookmark31). The core of APS is modeling and simulating complex production sys- tems and solving problems by intelligent optimization algorithms [[3–5]](#_bookmark32). The parallel machine scheduling problem (PMSP) is one of the classical combinatorial optimization problems in APS which

\* Corresponding author.

*E-mail addresses:* [wanghua@njtech.edu.cn](mailto:wanghua@njtech.edu.cn) (H. Wang), [liruicug@163.com](mailto:liruicug@163.com) (R. Li), [wygong@cug.edu.cn](mailto:wygong@cug.edu.cn) (W. Gong).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

**Production and hosting by Elsevier**

is applied in different areas of manufacturing systems including the electric wire-harness industry [[6]](#_bookmark43), freight system [[7]](#_bookmark45), and earth observation satellite scheduling [[8]](#_bookmark46). The PMSP consists of two problems including determining the processing machine for each job and the job processing sequence on each parallel machine. The processing time of each job is the same on each parallel machine. When the scales of the problem increase, the difficulty of solving PMSP are growing exponentially which makes optimizer hard to find the optimal solution. Since PMSP has a wide range of application value, it is very important to study how to solve PMSP well, which is helpful to improve the efficiency of practical applications.

Unrelated PMSP (UPMSP) is an extension of PMSP that defines that all jobs’ processing time is completely different on every par- allel machine. The UPMSP is more complex to solve and closer to real-world manufacturing than PMSP which considers the job’s processing time is the same on each machine. However, With eco- nomic globalization and the growth of global trade volume, con- ventional single-factory UPMSP cannot satisfy the fast

<https://doi.org/10.1016/j.eij.2023.05.008>

1110-8665/© 2023 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Computers and Artificial Intelligence, Cairo University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

manufacturing requirement. The enterprise has to reduce the orig- inal producing due date a half to occupy more part of the market. Thus, multiple factories are built for distributed manufacturing, and distributed UPMSP (DUPMSP) starts to get more consideration in recent years [[9]](#_bookmark49). Researchers study the DUPMSP and consider how to dispatch several orders to different identical factories to minimize the total maximum completion time for all factories. Nevertheless, the assumption of multiple factories is too idealistic. In recent years, Lu proposed the concept of heterogeneous factories and considered the processing time, shop types, machine numbers, and machine types should be different during practical manufac- turing [[10]](#_bookmark51). Thus, the distributed heterogeneous shop scheduling problems start to be an emergency research topic such as dis- tributed heterogeneous hybrid flow shop scheduling [[10]](#_bookmark51), job shop scheduling [[11]](#_bookmark52), flow shop scheduling [[12,13]](#_bookmark26), and flexible job shop scheduling (FJSP) [[14]](#_bookmark26). Due to its difficulty and complexity, there are a few works for distributed heterogeneous UPMSP (DHUPMSP) [[15,16]](#_bookmark26). The search space of DHUPMSP has been improved by an order of magnitude than UPMSP. Because the DHUPMSP has to solve three coupled sub-problems: consider the factory flexibility and dispatch each job to a heterogeneous factory, determine the processing sequence of the jobs in all heterogeneous factories, and select an unrelated machine for each job under machine flex- ibility. Thus, how to efficiently solve DHUPMPS becomes an emerg- ing topic.

The term memetic algorithms (MAs) is a combination of a population-based evolutionary algorithm and one or more local refinement strategies [[17–19]](#_bookmark26). In scheduling problem, the MAs usually execute local search refinement after the population updating which make the population rapidly converge to the real-world Pareto Front. Thus, the MAs have been widely applied in scheduling problems [[20,21]](#_bookmark26). As for multi-objective shop scheduling, the MAs can be classified into two types which are the decomposition-based MAs [[22]](#_bookmark27) and Pareto-based MAs [[23]](#_bookmark28). The decompos-ition-based MAs are based on the theory of MOEA/D [[24]](#_bookmark29) and the Pareto-based MAs are originated from the framework of NSGA-II [[25]](#_bookmark33). The decomposition-based MAs mainly depend on the definition of the lower bound for objective which is hard to define. Thus, the Pareto-based MAs are more flexible and suitable to solve DHUPMSP.

This study aims to solve a bi-objective distributed heteroge- neous unrelated parallel machine scheduling problem with mini- mizing total tardiness (TTD) and the completion time of the whole heterogeneous factories (makespan). To solve DHUPMSP, a knowledge and Pareto-based memetic algorithm (KPMA) is pro- posed for DHUPMSP. The main contributions of this work are sum- marized below:

1. A DHUPMSP with minimizing makespan and TTD simultane- ously is first considered.
2. Four problem-features-based heuristic initialization rules are proposed to generate a high-quality population with great convergence and diversity.
3. Four knowledge-based heuristic neighborhood structures are designed to rapidly reduce the makespan and TTD.
4. Our approach KPMA gets better results for solving DHUPMSP than five state-of-art.

The rest parts of this study are organized as follows: The recent works are introduced in Section [2](#_bookmark3). The problem descrip- tion and MILP model of DHUPMSP are introduced in Section [3](#_bookmark4). Section [4](#_bookmark9) illustrates our approach KPMA. The results of detailed numerical experiments are demonstrated in Section [5](#_bookmark16). Finally, the conclusion of this study and some future directions are sta- ted in Section [6](#_bookmark30).

1. Literature review
   1. *Related works of UPMSP*

UPMSP considers that all jobs’ processing time is different on each machine. A basic approximate algorithm is proposed for UPMSP by Pei [[26]](#_bookmark34). Chen studied an extension model of UPMSP which considered the time-of-use electricity price [[27]](#_bookmark35). Chen proposed its MILP model and applied CPLEX opti- mizer to solve it. Fang designed an adaptive large neighbor- hood search-based tabu search for UPMSP and combined a learning based automata to improve the efficiency of local searches [[28]](#_bookmark36). Zheng proposed a collaborative fruit fly algo- rithm for UPMSP with resource constraints and obtained better results than compared algorithm [[29]](#_bookmark37). Ding studied UPMSP with job deteriorating effects and designed a hybrid memetic algorithm for it [[30]](#_bookmark38). Wang developed an evolutionary discrete particles swarm optimizer for UPMSP and the proposed local search strategies greatly improved the convergence of the algo- rithm [[31]](#_bookmark39). An iterated greedy method was proposed in [[32]](#_bookmark40) for UPMSP and got good results. Wang studied UPMSP with a min–max regret criterion and designed an enhanced regret evaluation method to accelerate optimizing [[33]](#_bookmark41). Cao proposed a two-phase based memetic algorithm for stochastic UPMSP [[34]](#_bookmark42). Chen studied UPMSP with dual resource constraints and proposed the genetic algorithm with a local search for it [[35]](#_bookmark44). Wang researched UPMSP with controllable processing times and designed logic-based Benders decomposition for rapidly solving it [[36]](#_bookmark47).

* 1. *Related works of distributed UPMSP*

DUPMSP aims to parallelly produce all jobs in multiple dis- tributed factories which is harder to solve than UPMSP. Neverthe- less, there are few works for DUPMSP due to its complexity. Hatami proposed four fast and high-performing heuristics for DUPMSP [[9]](#_bookmark49). In [[37]](#_bookmark48), a hybrid distribution estimation algorithm was designed to minimize the makespan of DHUPMSP. Zhou designed a VNS-based imperialist competitive memetic search for DHUPMSP and got the best results [[38]](#_bookmark50). Lei developed a division strategy based artificial bee colony algorithm for DHUPMSP with preventive maintenance and the division strategy can efficiently accelerate convergence [[15]](#_bookmark26). Mnch studied DHUPMSP with the total weighted delivery time [[39]](#_bookmark53). Pan designed a two-population co-evolution based on knowledge and feedback for green DHUPMSP [[16]](#_bookmark26).

Algorithm 1: The Framework of MA.







* 1. *Memetic algorithms applied in shop scheduling*



The memetic algorithms have been widely applied in many kinds of shop scheduling problems because MAs can rapidly get close to the real Pareto Front. The framework of MAs is shown in Algorithm 1. The MAs are divided into two steps global search and local search which are efficient to solve shop scheduling prob- lems. Zhang proposed a MA for scheduling hybrid differentiation flowshop to enhance the convergence of co-evolution [[40]](#_bookmark54). Ding applied a hybrid MA for PMSP and got a 100% success rate to update neighborhood solutions [[30]](#_bookmark38). Lei developed a MA for hybrid flow shop scheduling (HFS) which can obtain solutions with great objectives values in short time [[41]](#_bookmark55). Abedi designed a multi- population MA for job-shop scheduling and outperformed other algorithms [[42]](#_bookmark55). Kurdi proposed a semi-constructive crossover based MA with mutation operators for flowshop scheduling and improved the metrics by 37.92% [[43]](#_bookmark55). Shao designed a MA for dis- tributed heterogeneous HFS and obtained the best results [[44]](#_bookmark55).

* 1. *Research gaps and discussions*

The previous work for DHUPMSP and MA has been reviewed in detail. However, there are some problems in the research of DHUPMSP which are stated below:

1. The previous research on DHUPMSP lacks the analysis for problem features. Thus, the local search strategies of the pre- vious works have strong randomness which leads to a low success rate for updating the neighborhood solutions.
2. The previous works usually take random initial rule which leads to a cold-start problem and consumes many computa- tion resources to converge.
3. The previous works lack elite strategy and the historical elite solutions are always abandoned due to the limited popula- tion size.

Thus, based on the discussion above, this work proposed a knowledge and Pareto-based MA for solving DHUPMSP. First, the problem features of DHUPMSP are analyzed. Second, based on problem knowledge, four heuristic initialization rules are designed. Next, four neighborhood structures based on problem features are designed to accelerate convergence. Finally, an elite strategy is developed to improve the usage rate of historical elite solutions to increase diversity.

1. Problem and model description
   1. *Description for DHUPMSP*

As for DHUPMSP, each order is regarded as a job. An instance of DHUPMSP has *n* jobs and *nf* heterogeneous factories. Every factory has *m* unrelated parallel machine and every job has only one stage.

Each job’s processing time *Tf* ;*i*;*k* is different for each machine, and

the processing time of every job on the same parallel machine is also different from every factory. Every job is determined a due date *Di*. The main goal is to dispatch *n* job to *nf* factories, select a machine for every job and, determine the job processing sequence in all unrelated parallel machines to minimize total tardiness (TTD) and makespan (*Cmax*).

Some assumptions of DHUPMSP are introduced following: i) transportation and setup time are not studied; ii) Meanwhile, at time zero, all unrelated parallel machines can be used. All jobs start being processed at stage one at time zero; iii) each job is allowed to choose only one factory. Meanwhile, every job is not allowed to be assigned to two different machines at a time; iv) all jobs’ process- ing times are certain; and v) one job can only be processed by one machine at the same time and cannot be interrupted during pro- cessing; Moreover, dynamic events such as machine breakdown, preventive maintenance are not considered.

* 1. *MILP model for DHUPSMP*

The notations of DHUPMSP are introduced below: Decision variables:

*f* ;*k*

subject to:

X*f* ;*i*;*k*;*t* 6 1; 6*i* ∈ *I*; *t* ∈ *Zf* ;*k* (4)

XX

*f* ∈*F k*∈*M*

- Y*i*;*f* : The binary value is set to one when job *Ii* is allocated to fac-

XXX*f* ;*i*;*k*;*t* P XXX*f* ;*i*;*k*;*t*+1; 6*f* ∈ *F*; *t* ∈ *Z*'

(5)

tory *f*; Otherwise, the value equals zero;

- X*f* ;*i*;*k*;*t*: The binary value is set to one when job *Ii* is dispatched to the position *t* of machine *Mk* in factory *f*; Otherwise, the value equals zero;

*i*∈*I k*∈*M*

Y*i*;*f*

X

*f* ∈*F*

*i*∈*I k*∈*M*

= 1; 6*i* ∈ *I* (6)

- *Cf* ;*k*;*t*: the completion time of *tth* position of machine *Mk* in fac-

tory *f*;

*Sf* ;*i*

+ X *T*

*f* ;*i*;*k*

* X*f* ;*i*;*k*;*t*

6 *Ff* ;*i*

; 6*i* ∈ *I*; *k* ∈ *M*; *f* ∈ *F* (7)

* *Ff* ;*i*

*f* ;*k*

: the finishing time of job *Ii* in factory *f*;

*t*∈*Zf* ;*k*

* *Bf* ;*k*;*t*: the beginning time of machine *Mk* at position *t* in factory *f*;
* *Sf* ;*i*: the starting time of job *Ii* in factory *f*;

*Bf* ;*k*;*t*+1

— *Bf* ;*k*;*t*

P X*f* ;*i*;*k*;*t*

*i*∈*I k*∈*M*

XX

* *Tf* ;*i*;*k*

; 6*f* ∈ *F*; *t* ∈ *Z*'

(8)

Parameters:

*Bf* ;*k*;*t* = *Sf* ;*i* · X*f* ;*i*;*k*;*t*; 6*i* ∈ *I*; *k* ∈ *M*; *f* ∈ *F*; *t* ∈ *Zf* ;*k* (9)

* *nf* : the number of factories;
* *F*: set of factories and *F* = {1; 2; ... ; *nf* };

0 6 *Sf* ;*i* ; *Bf* ;*k*;*t*

6 *L*; 6*i* ∈ *I*; *k* ∈ *M*; *f* ∈ *F*; *t* ∈ *Z*

*f* ;*k*

# (10)

* *m*: the number of all machines;
* *M*: set of machines and *M* = {1; 2; .. . ; *m*};
* *n*: the number of all jobs;
* *I*: set for jobs and *I* = {1; 2; ... ; *n*};
* *nt*: the number of all positions;
* *Zf* ;*k*: positions set on machines *MK* in factory *f* and

*Zf* ;*k* = {1; 2; ... ; *nt*};

where Eq. [(3)](#_bookmark8) are objective functions which are TTD and *Cmax*. Eq. [(4)](#_bookmark6) ensures that each job can not be processed on two different machi- nes at the same time. Eq. [(5)](#_bookmark6) guarantees that each position of a machine is available only when its preceding position is selected. Eq. [(6)](#_bookmark6) makes sure a job only be dispatched to one factory. Eq. [(7)](#_bookmark6) states the relationship between the start time and finish time of a job. Eq. [(8)](#_bookmark6) guarantees the correction of the beginning time between

'

- *Z*

*f* ;*k*

: top *nt* — 1 positions set on machines *MK* in factory *f* and

two adjacent positions. Eq. [(9)](#_bookmark6) ensures the constrain between oper-

*Zf* ;*k* = {1; 2; ... ; *nt* — 1};

* *Tf* ;*i*;*k*: The processing time job *Ii* processed by machine *Mk* in fac- tory *f*;
* *Di*: The duedate of job *Ii*;
* *L*: a large integer for keeping the consistency of the inequality; Indices:
* *f*: factory index;
* *k*; *k*': machine index;
* *i*; *i*': job index;
* *t*: position index;

ation start time and machine start time. [Eq.(10)](#_bookmark6) is values’ boundaries.

* 1. *Problem features analysis*

The DHUPMSP has two objectives the makespan and TTD. The problem features and proofs are stated below:

*Feature1:* The makespan only depends on the machine with the max workload of all factories.

*Proof1:* In DHUPMSP, each job has only one operation and there is no idle time on every parallel machine. Thus, the finish time of each job *Ft* = *Ft*—1 + *Pi*. The makespan depends on the completion

time of the last finished job. Thus,

*t*=1

The objectives of DHUPMSP in this work are TTD and *Cmax*,

*Cmax* = *Flast* = P*nt*

*Pi* \* X*f* ;*i*;*k*;*t*; 6*i* ∈ *I*; *f* ∈ *F*; *k* ∈ *M*. Thus, the make-

which are elaborated as follows:

*Toal tardiness criteria:* TTD is the economic metric for the enter- prise. Satisfying the due date of each job can increase the order number and income of the enterprise. The TTD criteria are defined as follows

*n*

XX }

span depends on the machine with the max workload and has no relationship with other machines.

*Feature2:* On every parallel machine, changing the job sequence cannot reduce the makespan.

*Proof2:* Assume that there are two adjacent jobs *I*1 and *I*2. The processing time *P*1 > *P*2, and the completion time *F*1 < *F*2. Because

*n f*

*TTD* = max *Ff* ;*i* · Y*i*;*f* — *Di*; 0 . (1)

*i*=1 *f* =1

*Makespan cirteria:* Makespan is an efficiency metric for the shop. The workers wish to finish their job as much as possible and to reduce their work time in the whole production cycle. The make- span criteria are stated below:

*Cmax* = max{*Ff* ;*i*}; 6*f* ∈ *F*; *i* ∈ *I*. (2)

The MILP model of bi-objectives DHUPMSP is introduced as follows:

there is no idle time on each machine. *F*2 = *F*1 + *P*2 and *F*1 = *F*0 + *P*1. Then, swap *I*1 and *I*2 to process *I*2 ahead. The new fin- ish time of *I*1 and *I*2 is *F*3 and *F*4. Meanwhile, *F*3 > *F*4; *F*4 = *F*0 + *P*2,

and *F*3 = *F*4 + *P*1 = *F*0 + *P*1 + *P*2. Thus, *F*3 == *F*2 which means the finish time does not change and nor does makespan.

*Feature3:* When there are many jobs over due date, the job with earlier due date and smaller processing time should be processed ahead.

*Proof3:* Assume that there are two adjacent jobs *I*1 and *I*2. The processing time *P*1 > *P*2, the due data *D*1 < *D*2, and the completion time *F*1 < *F*2. Because there are no idle time on each machine.

min *F*1 = *TTD*

*F*2 = *F*1 + *P*2 and *F*1 = *F*0 + *P*1. Moreover, *F*2 = *F*0 + *P*1 + *P*2. The tar-

# min *F*2

= *Cmax*

(3)

diness of two jobs is *T*1 = *F*1 — *D*1 and *T*2 = *F*2 — *D*2. The total tardi- ness is *TDD* = *T*0 + *T*1 + *T*2 = *T*0 + *F*1 + *F*2 — *D*1 — *D*2 =

*T*0 + 2 \* *F*0 + 2 \* *P*1 + *P*2 — *D*1 — *D*2. Then, swap *I*1 and *I*2 to pro- cess *I*2 ahead. The new finish time of *I*1 and *I*2 is *F*3 and *F*4. Mean- while, *F*3 > *F*4; *F*4 = *F*0 + *P*2 , and *F*3 = *F*4 + *P*1 = *F*0 + *P*1 + *P*2. Next,

the new tardiness *T*3 = *F*3 — *D*1 and *T*4 = *F*4 — *D*2. The total tardi- ness *TDD*'= *T*0 + *T*3 + *T*4 = *T*0 + 2 \* *F*0 + 2 \* *P*2 + *P*1 — *D*1 — *D*2.

D*TDD* = *TDD*'— *TDD* = *P*2 — *P*1 < 0. Thus, feature3 has been proven.

Based on the analysis for DHUPMSP above, some conclusions are obtained below:

*Conclusion1:* Based on *feature 1* and *feature 2*, it is obvious that moving the job from the max workload machine can reduce the makespan.

*Conclusion2:* This conclusion is based on *feature 3*. Find the job which is over due date and search the front job which has bigger due date on the same machine. Then, moving the over due date job to the searched place can reduce tardiness.

1. Our approach: KPMA

In this section, our algorithm: KPMA will be introduced in detail.

* 1. *Motivation*

Based on the research gap mentioned in Section [2.4](#_bookmark5) and prob- lem features stated in Section [3.3](#_bookmark7), this work proposed a knowl- edge and Pareto-based MA for DHUPMSP. First, the previous works lack efficient initialization which results in a cold-start problem. Four heuristic initialization rules are proposed to con- struct high-quality solutions to enhance convergence. Second, the local search strategies of previous works are based on the random selection which is inefficient and wastes many computa- tion resources. Thus, four heuristic neighborhood structures are designed to increase the success rate of local search. Finally, an

elite strategy is proposed to enhance the usage rate of historical solutions.

* 1. *Framework of KPMA*

Algorithm 2 states the framework of KPMA. First, KPMA also ini- tialized two swarms *P* and *C*. Meanwhile, *P* is initialized by a hybrid heuristic initialization to get great convergence and diver- sity simultaneously. Second, *P* will execute NSGA-II [[25]](#_bookmark33) for global search. Moreover, after the environmental selection, *C* obtains non-dominated solutions from *P*. Then, the variable neighborhood search (VNS) is adopted to rapidly get close to the real Pareto Front. Next, all non-dominated solutions are stored int the elite archive to increase diversity. Finally, the searched optimal Pareto solutions will be output from the elite archive *C*.

* 1. *Encoding and decoding*

*Encoding schema*: In DHUPMSP, job sequence (*JS*), factory assignment (*FA*) and machine selection (*MS*) are represented by three vectors. [Fig. 1](#_bookmark10) shows the encoding schema for DHUPMSP.

In *FA* and *MS*, the job order is from *J*1 to *Jn* and the correspondence is unchangeable. Nevertheless, the job processing order in *JS* needs to be permuted.

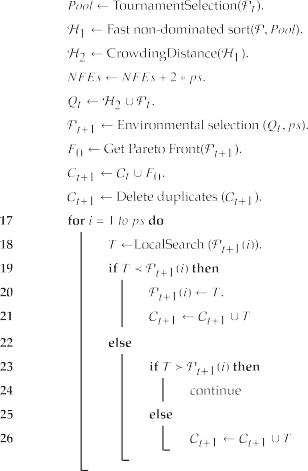
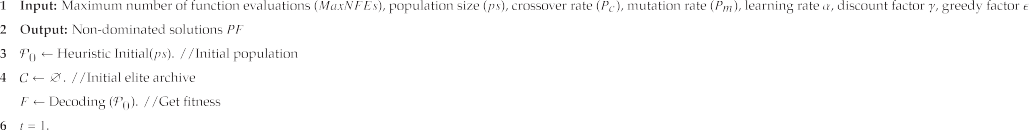
*Decoding schema*: First, according to the *FA* vector, all jobs are assigned to every heterogeneous factory. Next, the job processing sequences in every factory are got from the *JS* vector. Then, all jobs are allocated to each unrelated parallel machine according to the

*MS* vector and the processing time *Tf* ;*i*;*k* can be got. Then, the start

and completion time of every job is calculated. Furthermore, the max finish time can be obtained and *Cmax* is got. Finally, if a job’s



Algorithm 2: The Framework of KPMA.



finish time is bigger than its due date *Di*, the tardiness will be sum- marized and the TTD can be got.

* 1. *Hybrid heuristic initialization*

Initialization plays an important role in the global search stage. KPMA combines four heuristic initialization rules to generate ini-

tialized population with great convergence and diversity. The rules are stated below:

*SPT rule:* The SPT rule aims to select the shortest processing time machine. Due to problem *feature 1* and *feature 2*, reducing machine processing time can lower the makespan. The SPT rule is stated in Algorithm 3. As for each job, greedily choose the parallel machine with the smallest processing time.





Algorithm 3: The shortest processing time (SPT) rule.





Algorithm 4: The min factory workload (MFW) rule.







Algorithm 5: The min finish time (MFT) rule.





*MFW rule:* The MFW rule focuses to balance the factory work- load. Due to problem features, reducing the job number gap can

the earlier due date ahead. Second, repeat the step until the job sequence is determined.

efficiently reduce the finish time gap between each factory. The

MFW rule is described in Algorithm 4. First, record the processing time of each job on the selected machine in all factories. Second, sort the record and count the job number in each factory. Finally, select the factory with the smallest job number for each job.

*MFT rule:* The MFT rule aims to balance the workload of each machine. Based on problem *feature 1*, *feature 2* and *conclusion 1*, balancing the workload of all machines can efficiently reduce the makespan. The MFT rule is stated in Algorithm 6. First, divide all jobs into every heterogeneous factory according to the factory assignment vector. Then, in each factory, calculate the start and finish time of each job. Next, select the machine with the smallest finish time in the selected factory. Finally, update the start and fin- ish time of each job, and the finish time of the selected machine.

*EDD rule:* The goal of the EDD rule is to satisfy the order require- ment to reduce tardiness. The EDD rule is stated in Algorithm 7. First, compare the due date of each job and process the job with

Algorithm 6: The earliest due date (EDD) rule.





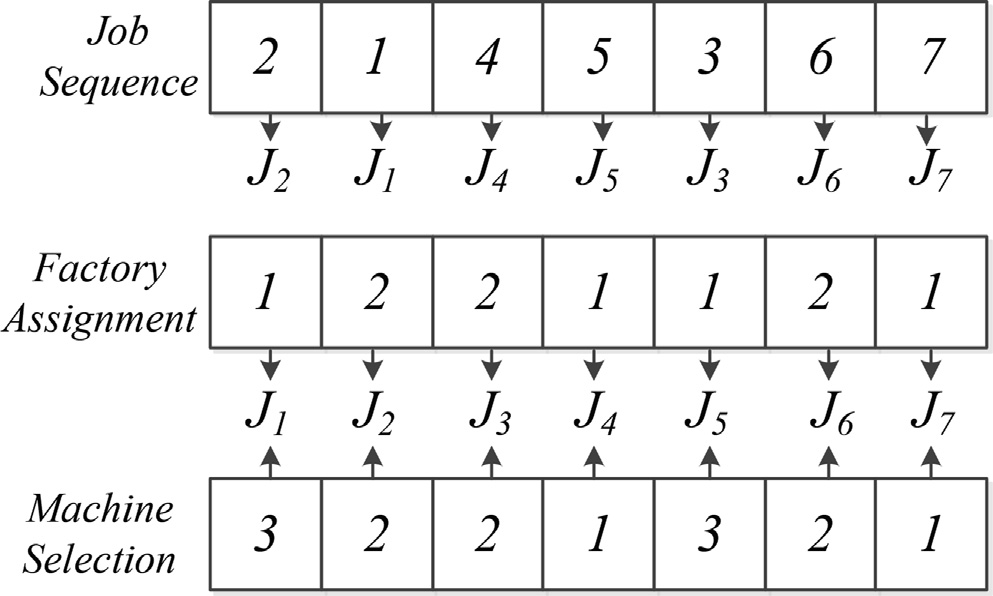


Fig. 1. An example for encoding schema for DHUPMSP.

*Hybrid heuristic initialization:* To keep great convergence and diversity simultaneously, the whole population is initialized by random rule first. Then, divide four sub-populations sizing *ps*/5. Then, the first sub-population executes SPT to rapidly get close

to the lower bound of the makespan. Next, the second sub- population adopts MFW to reduce makespan without large step converging. Then, the third sub-population applied the MFT rule to reduce the makespan. Moreover, the fourth sub-population used the EDD rule to get close to the objective space with low TDD. Finally, the rest of the population generated by random rule is evenly distributed in the target space to maintain diversity.

* 1. *Global search for producer population*

The objective of the global search is to sufficiently explore the decision space of DHUPMSP to keep great diversity. The global search is designed according to the Pareto domination based multi-objective evolutionary framework NSGA-II [[25]](#_bookmark33). First, the two-player tournament selection is applied to select the mating pool. Then, universal crossover (UX) [[45]](#_bookmark56) and precedence operation crossover (POX) [[14]](#_bookmark26) are adopted to generate offspring which are shown in [Fig. 2](#_bookmark12) and [Fig. 3](#_bookmark11). As for POX, the jobs set are randomly divided into two subsets *A* and *B* first. Then, the jobs from *A* in *JS*1 are copied to the same positions in *JS*3, and the jobs from *B* in *JS*2 are copied to the same positions in *JS*4. Next, the jobs in *B* are

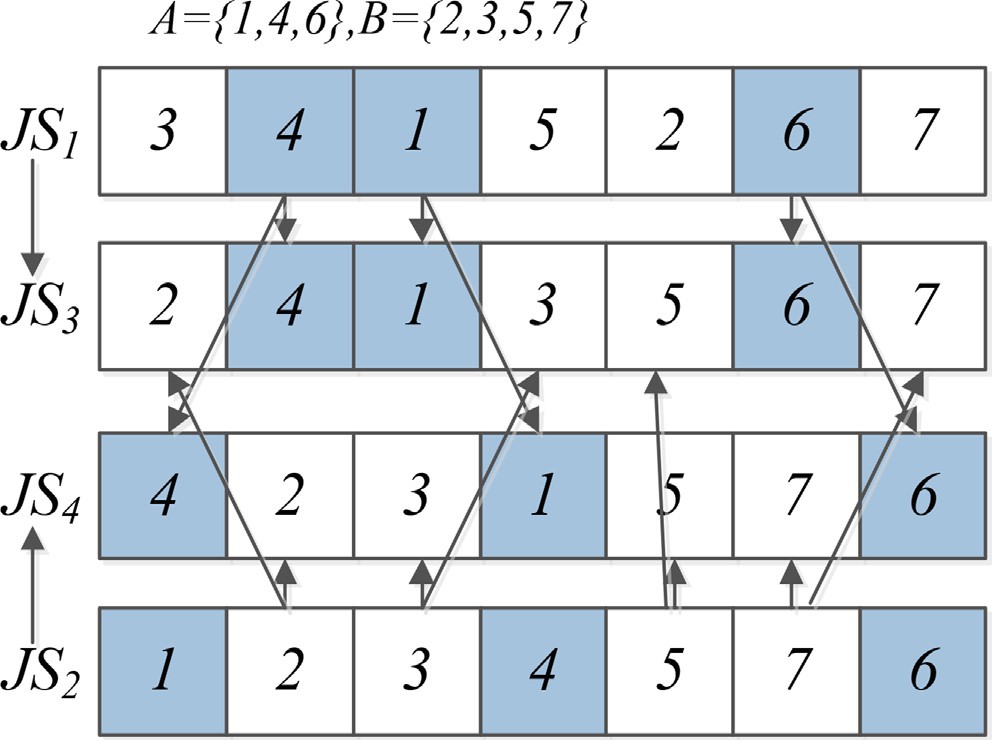
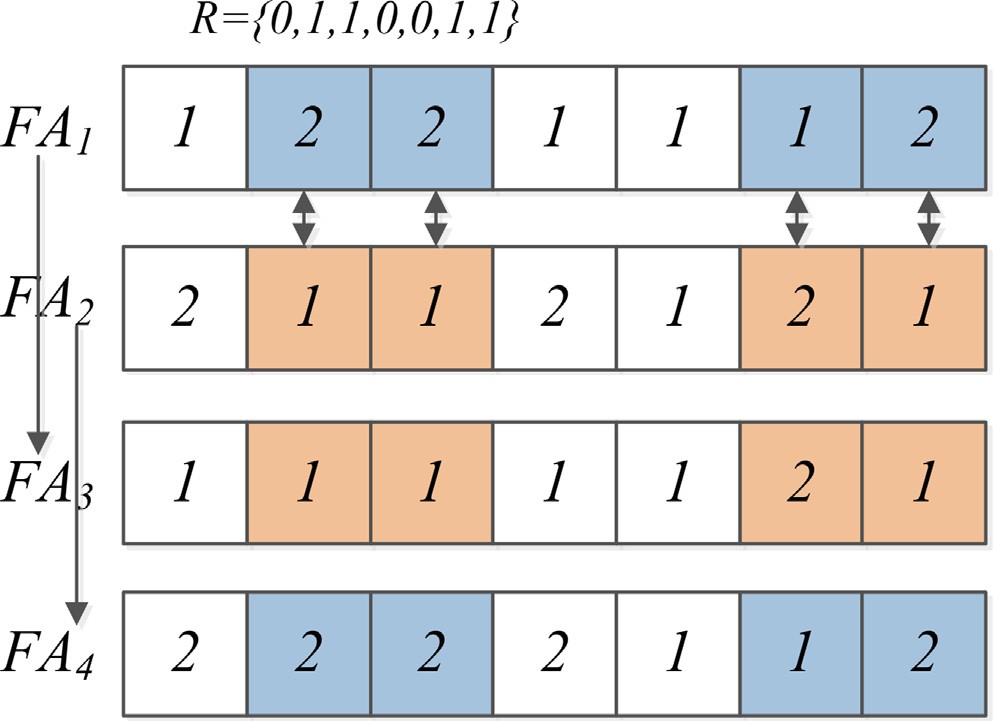


Fig. 2. POX for *JS*.

Fig. 3. UX for *FA* and *MS*.

copied to the empty space of *JS*3 with the same order in *JS*2 from left to right. The same operator is adopted to *JS*4. As for UX, a ran-

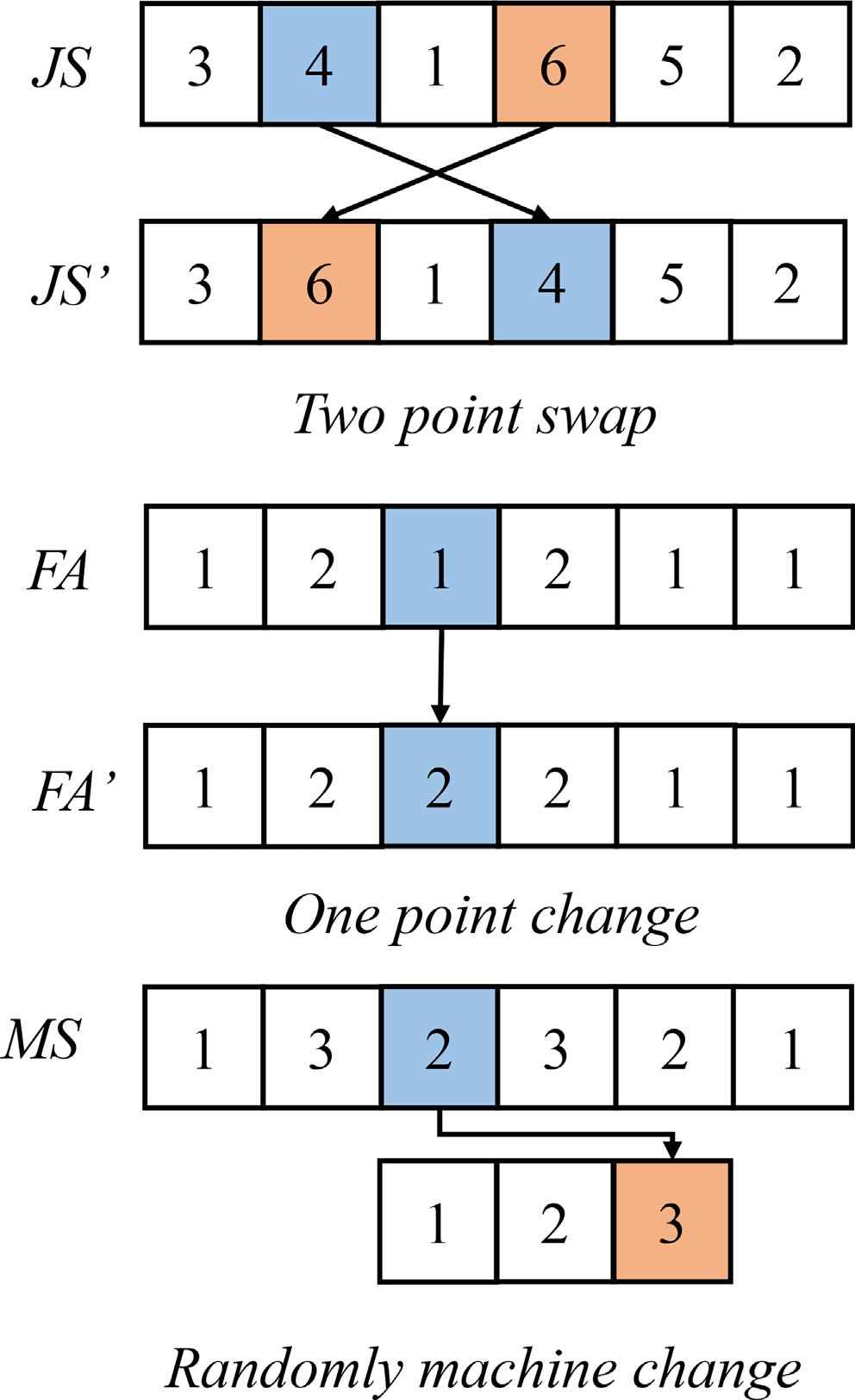


Fig. 4. Mutation for *JA*, *MS* and *FA*.

dom 0–1 vector *R* is generated first which size *n*. Then, traverse the parents from left to right. If the value of *R* is 1, exchange the gene of the parent.

Moreover, each offspring will adopt three mutation operators with probability Pm to enhance diversity. As for *JS* vector, ran- domly choose two jobs and exchange their positions. For *FA* vector, randomly choose a job and move it to another heterogeneous fac- tory. For *MS* vector, randomly choose a job and move it to another parallel machine. The mutation operators are shown in [Fig. 4](#_bookmark13). Finally, the child solutions obtained by the evolution operators are merged with the parent population *P*. The combined swarm

is chosen by the crowding distance strategy and fast non- dominated sorting to generate the population of next generation [[25]](#_bookmark33).

* 1. *Knowledge-based local reinforcement*

Designing problem features based local search strategies can greatly increase their efficiency for solving shop scheduling prob- lems. According to the problem features of DHUPMSP, four neigh- borhood structures are developed which are introduced below:

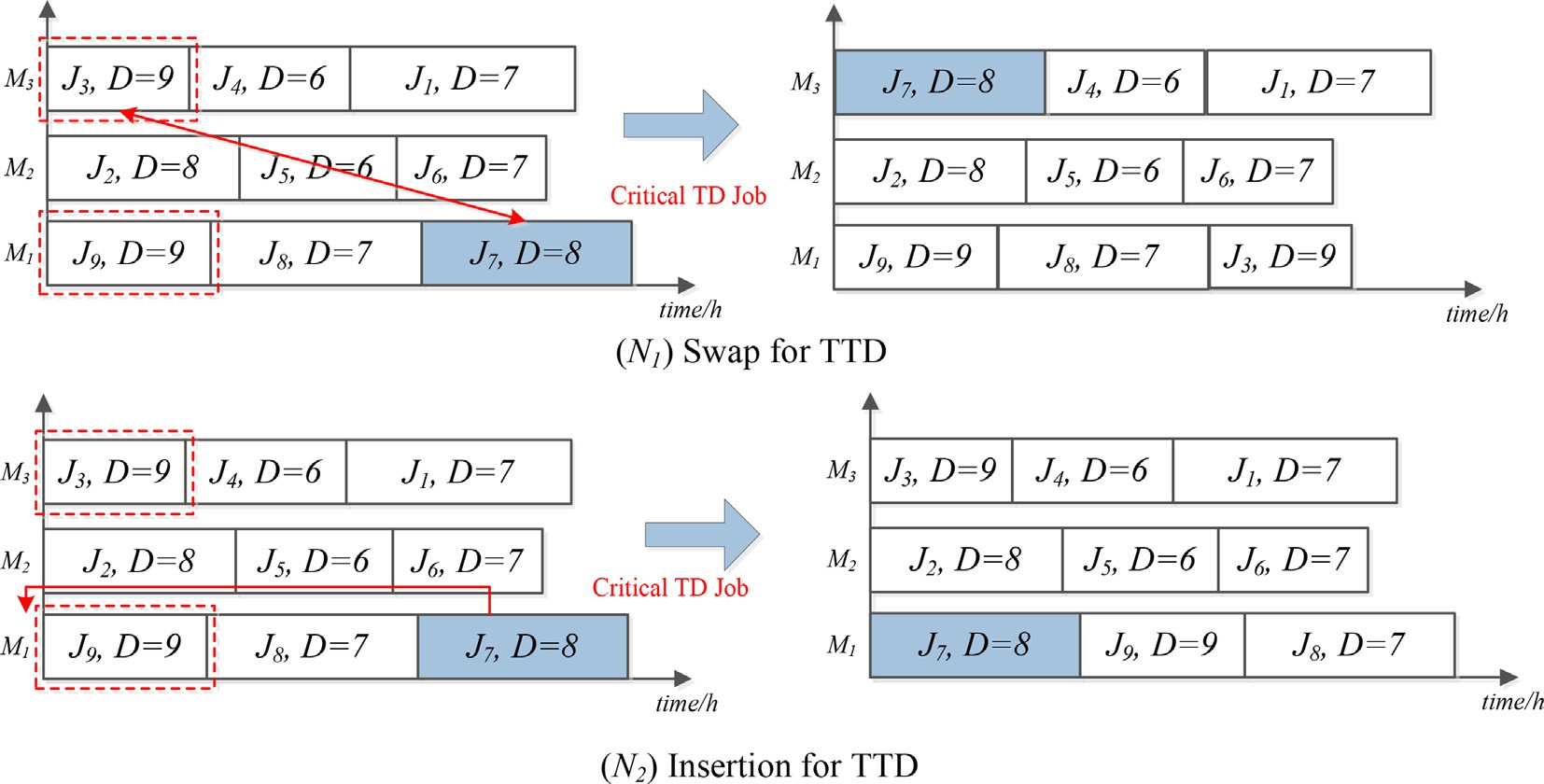


Fig. 5. An example for encoding schema for DHUPMSP.

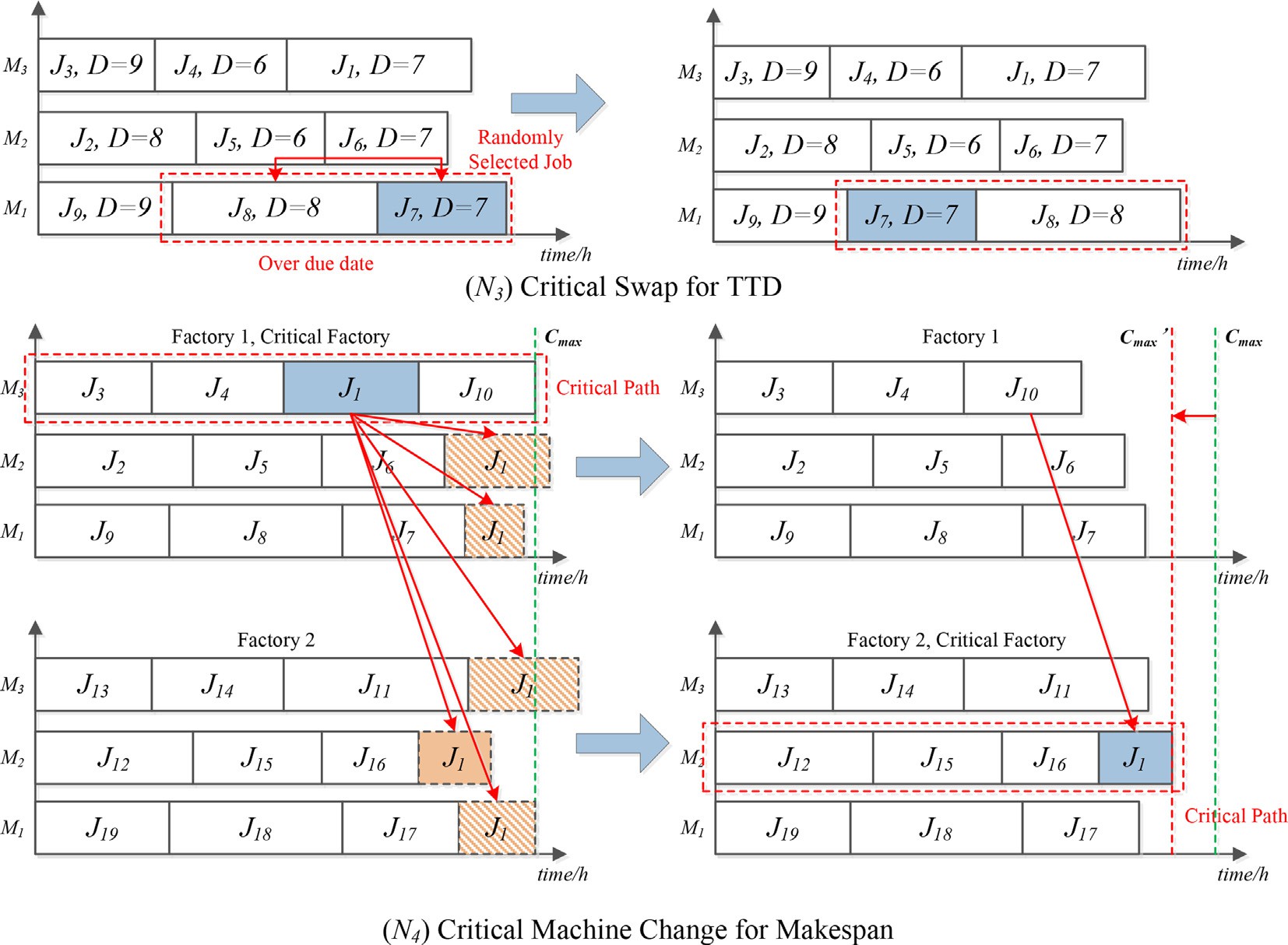


Fig. 6. An example for encoding schema for DHUPMSP.

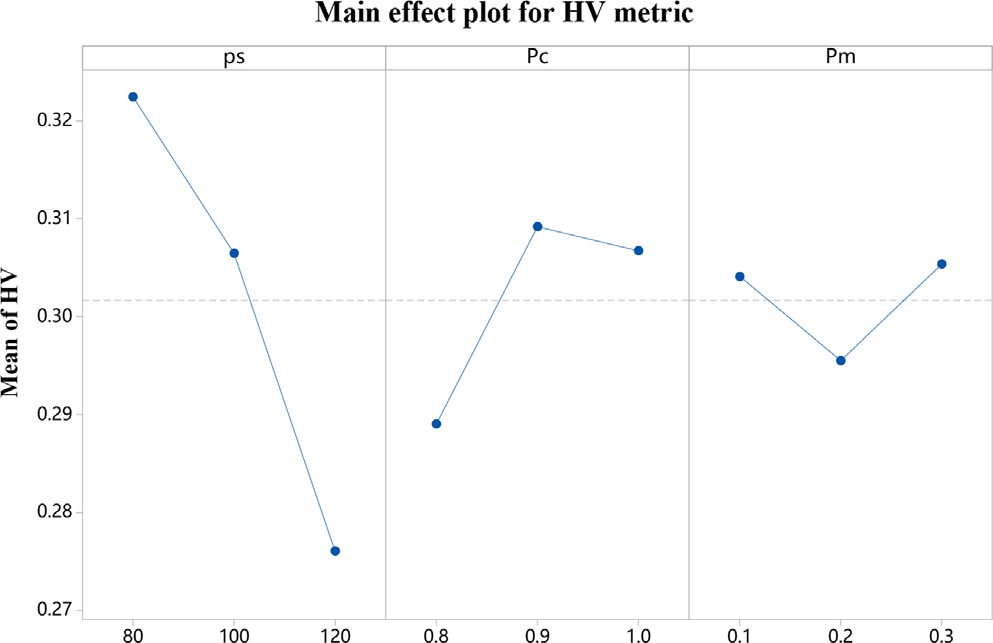


Fig. 7. Main effects plot of HV metric.

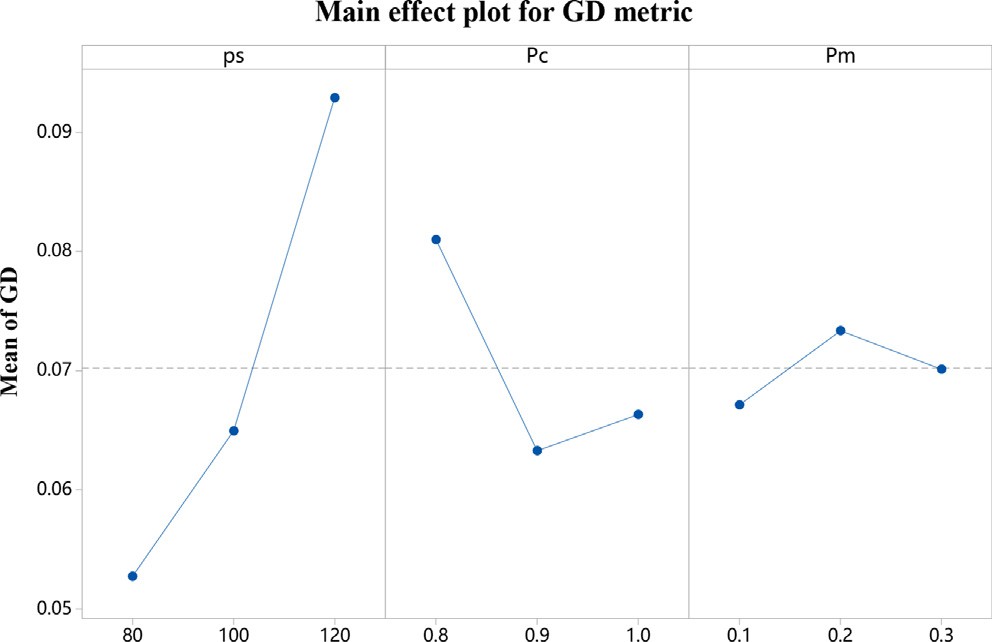


Fig. 8. Main effects plot of GD metric.

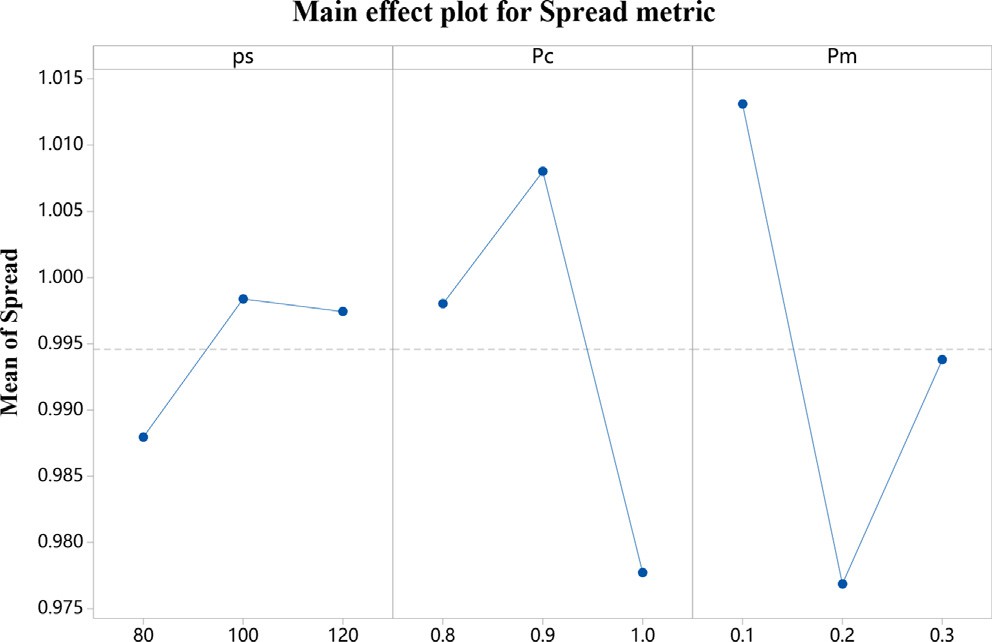


Fig. 9. Main effects plot of Spread metric.

* + 1. *N1 (Swap for TDD)*

[Fig. 5](#_bookmark14) shows the procedure of N1. First, find the job with the max tardiness of the all factories. Second, traverse the jobs in front of the critical job. If there is a job that has a bigger due date, swap the machine selection and the job sequence of the critical job and selected job.

* + 1. *N2 (Insertion for TDD)*

[Fig. 5](#_bookmark14) shows the procedure of *N*2. First, find the job with the critical job. Second, traverse the jobs in front of the critical job. If there is a job that has a bigger due date, insert the critical job in front of the selected job.

* + 1. *N3 (Critical swap for TDD)*

This neighborhood structure is designed based on problem *fea- ture 3*. [Fig. 6](#_bookmark15) shows the procedure of *N*3. First, randomly select a job. Then, record the tardiness of the selected job and the job front of it. Next, assume to swap these two jobs and calculate the new tardiness of them. Finally, if the new tardiness is reduced, swap the selected job and the job front of it.

* + 1. *N4 (Critical machine change for Cmax)*

This neighborhood structure is designed based on problem *fea- ture 1* and *feature 2*. [Fig. 6](#_bookmark15) shows the procedure of *N*4. First, the critical path is found which is the same as the machine with the max workload. Second, randomly choose a job on the critical path. Next, test the processing time on each machine of all factories. If there is a machine that can reduce the makespan, move the job to the machine which satisfies the criteria.

KPMA adopts a variable neighborhood search to increase the convergence and keep the diversity during evolution. The VNS in KPMA is executed by a random way.

* 1. *Elite strategy*

Based on the research gap in Section [2.4](#_bookmark5), this work proposed a simple elite strategy to increase the diversity. During the evolu- tionary stage and local search stage, store the non-dominated solu- tions to the elite archive. Then, the elite archive only keeps its non- dominated solutions at the end of every generation.

1. Results of numerical experiment

In Section [4](#_bookmark9), Our approach KPMA is illustrated detailedly. More- over, numerical experiments are executed to test the effectiveness of KPMA in this section. All variant and comparison algorithms are coded in python. The hardware environment is on a CPU of Intel(R) Xeon(R) Gold 6246R with 3.4 GHz and 384G RAM. Moreover, the running environment is Pycharm2021 with python3.8.

* 1. *Test problems and evaluation metrics*

To verify the effective of KPMA, 20 test problems with different scales for DHUPMSP are created. The factories amount belongs to

Table 1

The Friedman run-and-sum test results for all variant algorithms of KPMA (significant level *a* = 0.05).

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MOEAs |  | HV |  |  |  | GD |  |  |  | Spread |  |  |
|  | rank |  | *p*-value |  | rank |  | *p*-value |  | rank |  | *p*-value |
| KPMA-L | 5.20 |  | 5.34E-11 |  | 5.20 |  | 1.46E-11 |  | 5.15 |  | 2.07E-01 |  |
| KPMA-E | 3.10 |  |  |  | 3.30 |  |  |  | 5.00 |  |  |  |
| KPMA-I | 3.35 |  |  |  | 2.85 |  |  |  | 4.15 |  |  |  |
| KPMA | 1.20 |  |  |  | 1.15 |  |  |  | 3.60 |  |  |  |

Table 2

Statistical results of HV (max) metric of all KPMA variants.





Table 3

Statistical results of GD (min) metric of all KPMA variants.





*nf* ∈ {2, 3} and the job number ranges from *n* ∈ {20, 40, 60, 80, 100}. The processing time *Tf* ,*i*,*k* is from {5, 95} which is different in every heterogeneous factories and the machi- nes amount *nm* =∈ {4, 6}.. The duedate *Di* ranges from average pro- cessing time added and minus 5 or 10. Finally, 20 test problems

and *x* ∈ *P*. Furthermore, an algorithm with better convergence has smaller GD metric value.

*Diversity metric:* Spread [[46]](#_bookmark56)

|*P*\*|

X X

*d*(*P*,*P*\*)+ |*d*(*X*,*P*)—*d*¯|

with several scales are created which are named as 20*J*4*M*2*F*. The

stop criteria is set to MaxNEFs=400 \* *n* P 2 \* 104.

*Spread*(*P*, *P*\*)= *i*=1

X*d P*,*P*\*

|*P*\*|

*X*∈*P*

*P*

,

*P*\* *d*¯

Three metrics usually applied to multi-objective optimization algorithms (MOEAs) are adopted to represent the performance of all MOEAs and the equations are defined as follows:

*Convergence metric:* Generation distance (GD) [[25]](#_bookmark33)

( )+(| |—| |)

*i*=1

*d*(*X*, *P*)= *minY*∈*P*,*Y*–*X* ||*F*(*X*)— *F*(*Y*)||,

*d*¯ =  1 X*d*(*X*, *P*)

|*P*|

# (12)

rXﬃﬃﬃﬃﬃﬃﬃmﬃﬃﬃﬃﬃiﬃnﬃﬃﬃ*d*ﬃﬃﬃ(ﬃﬃ*x*ﬃﬃ,ﬃﬃ*y*ﬃﬃﬃ)ﬃﬃ2ﬃ

*X*∈*P*

In Eq. [12](#_bookmark21), *d* represents the Euclidean distance of each Pareto solu-

*GD*(*P*, *P*\*)=

*y*∈*P*

*x*∈*P*\*

|*P*|

# (11)

tion and its adjacent point. Furthermore, an algorithm with better diversity has smaller Spread metric value.

In Eq. [11](#_bookmark20), the notation *P* is the non-dominated solutions set obtained by every algorithm and *P*\* represents the optimal Pareto reference solutions set calculated by all MOEAs. Moreover, *d*(*x*, *y*) represents the second-order Euclidean distance between *y* ∈ *P*\*

*Comprehensive metric:* Hypervolume (HV) [[47]](#_bookmark56)

*P*

[

*HV* (*P*, *r*)= *v*(x, *r*). (13)

x∈*P*

Table 4

Statistical results of Spread (min) metric of all KPMA variants.





Table 5

The Friedman run-and-sum test results for all comparison algorithms and KPMA (significant level *a* = 0.05).

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MOEAs |  | HV |  |  |  | GD |  |  |  | Spread |  |  |
|  | rank |  | *p*-value |  | rank |  | *p*-value |  | rank |  | *p*-value |
| NSGA-II | 4.65 |  | 1.73E-12 |  | 4.70 |  | 8.19E-12 |  | 2.90 |  | 1.44E-02 |  |
| MOEA/D | 3.40 |  |  |  | 2.80 |  |  |  | 4.65 |  |  |  |
| DABC | 2.60 |  |  |  | 3.30 |  |  |  | 3.10 |  |  |  |
| KTPO | 4.85 |  |  |  | 4.65 |  |  |  | 3.75 |  |  |  |
| VICA | 4.50 |  |  |  | 4.55 |  |  |  | 3.80 |  |  |  |
| KPMA | 1.00 |  |  |  | 1.00 |  |  |  | 2.80 |  |  |  |

Table 6

Results obtained by the Wilcoxon test for algorithm KPMA.

HV

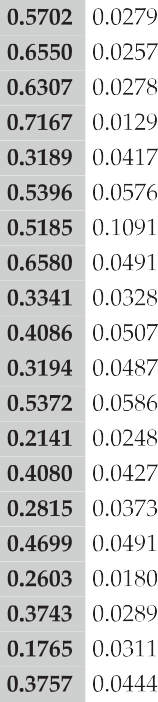
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | VS | *R*+ *R*— Exact P-value Asymptotic P-value | | | | |  |
| NSGA-II | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |
| MOEA/D | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |
| DABC | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |
| KTPO | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |
| VICA | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |
|  | |  |  | GD |  |  |
| VS | | *R*+ *R*— Exact P-value Asymptotic P-value | | | | | |
| NSGA-II | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |  |
| MOEA/D | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |  |
| DABC | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |  |
| KTPO | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |  |
| VICA | | 210.0 | 0.0 |  | 1.9074E-6 | 0.000082 |  |
|  | |  |  | Spread |  |  |  |
| VS | | *R*+ *R*— Exact P-value Asymptotic P-value | | | | | |
| NSGA-II | | 105.0 | 105.0 | P0.2 | | 0.985107 | |
| MOEA/D | | 155.0 | 55.0 | 0.06372 | | 0.059389 | |
| DABC | | 120.0 | 90.0 | P0.2 | | 0.562821 | |
| KTPO | | 133.0 | 77.0 | P0.2 | | 0.287337 | |
| VICA | | 139.0 | 71.0 | P0.2 | | 0.197754 | |

In Eq. [13](#_bookmark21), the notation *P* is the non-dominated solutions set obtained by every algorithm. Meanwhile, notation x represents a normalized non-dominated solution from each algorithm and *r* is

the reference point in normalization objective space and *r* is usually set to (1.1, 1.1) to calculate the boundary points. Moreover, notation *v* is the hypercube volume value constructed by each non-

Table 7

Statistical results of HV (max) metrics of all comparison algorithms.



dominated solutions. Furthermore, an algorithm with better con- vergence has smaller GD metric value.

* 1. *Parameter analysis experiment*

The parameter setting seriously affects an algorithm’s perfor- mance for solving DHUPMSP. The KPMA has three parameters including population size *ps*, crossover rate *Pc*, mutation rate *Pm*. To simplify the parameter experiment, a Taguchi method [[48]](#_bookmark56) is applied. Moreover, the parameters’ levels are designed following:

*ps* = {80, 100, 120}; *Pc* = {0.8, 0.9, 1.0}; *Pm* = {0.1, 0.2, 0.3}. An

orthogonal design *L*9(33) is used for parameter experiment. For a fair comparison, every parameter setting independently executes ten times and the stop criteria are MaxNFEs=400 \* *n*. Moreover, the average values of HV, GD and Spread metrics of each indepen- dent run are recorded. [Figs. 7–9](#_bookmark17) show three main effects plots of all parameters. According to three plots, the optimal parameter con-

figuration is that *ps* = 80, *Pc* = 0.9, and *Pm* = 0.1.

* 1. *Effectiveness of all components of KPMA*

To evaluate the effectiveness of every improvement proposed in this work, three variant algorithms are generated which are. i) KPMA-L is the KPMA without knowledge-based VNS; ii) KMPA-E is KPMS without the elite strategy; iii) KPMA-I is the KPMA without hybrid heuristic initialization. For a fair comparison, all algorithms independently run 10 times on 20 test problems. The stop criteria

are MaxNFEs=400 \* *n* P 2 \* 104.

[Tables 2–4](#_bookmark19) state the statistical results of HV, GD, and Spread metrics of all variant algorithms. In each table, all optimal values

of every metric are marked by bold. Furthermore, [Table 1](#_bookmark18) shows the results of Friedman rank-and-sum test. Several conclusions are obtained following: i) Comparing KPMA and variant algorithm KPMA-L can evaluate the performance of the designed knowledge- based VNS. ii) The comparison results of KPMA and KPMA-E shows the effectiveness of the elite strategy. iii) Comparing KPMA and KPMA-I can evaluate the effectiveness of the developed hybrid

heuristic initialization. iv) The *p*-value6 0.05 represents that KPMA

is significantly superior to all variant algorithms.

* 1. *Comparison experiment and discussions*

In this section, KPMA is compared with NSGA-II [[25]](#_bookmark33) and MOEA/D [[24]](#_bookmark29). Additionally, three state-of-art algorithms for DHUPMSP called DABC [[15]](#_bookmark26), KTPO [[16]](#_bookmark26) and VICA [[38]](#_bookmark50) are com- pared. The parameters of each comparison algorithms are set with the best configuration according to their references. The mutation probability *pm* = 0.2, crossover probability *pc* = 0.9 and population size *ps* = 100 for DABC, KTPO, NSGA-II and MOEA/D. The population sizes *ps* = 80 for KPMA. The neighborhoods updating range *T* = 10 for MOEA/D. Assimilation probability *Pa*

= 0.4, total imperialist countries *Nimp* = 10 and revolutionary probability *Pr* = 0.2 for VICA. To permit the fairness, all algo- rithms have the same stop criteria

(MaxNFEs=400 \* *n* P 2 \* 104). Due to the complexity of DHUPMSP, each comparison algorithm independent runs twenty times in 20 test problems. [Tables 7–9](#_bookmark24) show the statistical results (mean and standard deviation values) of all MOEAs for HV, GD, and Spread metrics in 20 test problems. Furthermore, the nota- tion ‘‘—” and ‘‘+” represents that the comparison algorithm is sig-

Table 8

Statistical results of GD (min) metrics of all comparison algorithms.

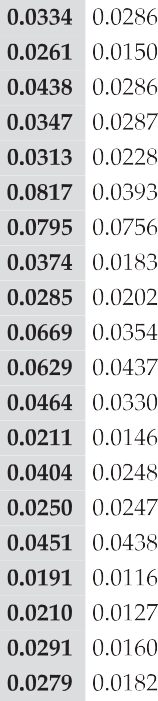
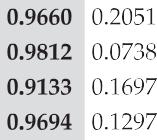
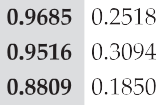


Table 9

Statistical results of Spread (min) metrics of all comparison algorithms.



nificantly worse and better than KPMA, and ‘‘=” states there is no significant difference. In addition, the optimal values are marked in bold. As [Tables 7–9](#_bookmark24) show, as for HV and GD metrics, KPMA is significantly superior to all compared MOEAs, which represents that KPMA has better convergence and comprehen- sive performance than comparison algorithms. As for the Spread metric, KPMA has no significant difference from other algo- rithms. [Table 5](#_bookmark22) shows the results of the Friedman rank-and-

sum test for all MOEAs in 20 test problems. KPMA has the best rank for all indicators, where the *p*-value6 0.05 states that KPMA is significantly superior to the compared MOEAs. [Table 6](#_bookmark23) records the results of Wilcoxon test for all metrics. The notation ”*R*+/*R*— ” means the degree that KPMA is significantly better/worse than

the compared algorithm. The gap between *R*+ and *R*— is larger the significance is stronger. As shown in [Table 6](#_bookmark23), the KPMA is

significantly better than all compared algorithms on HV and GD metrics where the *p*-value< 0.05. This evaluates the effec- tiveness of the proposed knowledge-based strategies. As for

Spread metric, the KPMA has no significant difference because the feature of the instances which makes the non-dominated solutions of each algorithm are close to each other. In summary, the KPMA has better comprehensive performance than compared algorithms.

The success of KPMA relies on its design. First, the proposed hybrid heuristic initialization provides an initial population with great convergence and diversity which let KPMA be far ahead before starting evolution. Second, the problem features-based vari- able neighborhood makes KPMA efficiently converge and success- fully improves the efficiency of local search. Finally, the elite

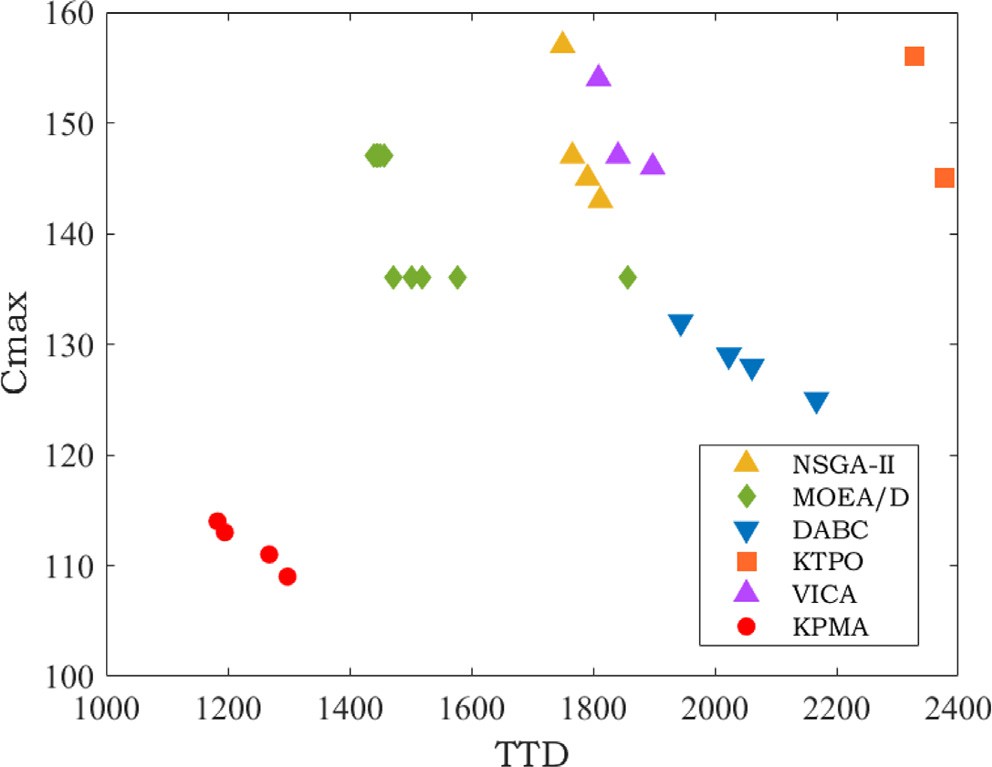


Fig. 10. PF comparison results of all algorithms on 100J6M3F.

strategy improves the usage of historical elite solutions to increase the diversity of final Pareto solutions. Furthermore, [Fig. 10](#_bookmark25) displays the Pareto Front results of all algorithms on instance 100J6M3F which are selected with the best HV metric from 20 runs. Observ- ing the diversity and convergence of each PF, KPMA can obtain bet- ter Pareto solutions on two sides than all comparison algorithms, which shows that KPMA can get solutions having lower objective values and get closer approximations towards practical PF. There- fore, KPMA is capable of solving DHUPMSP well.

1. Conclusion

This work put forward a knowledge and Pareto-based memetic algorithm for the bi-objective distributed heterogeneous unrelated machine scheduling problem. First, three problem features of DHUPMSP are analyzed and two key conclusions for optimizing DHUPMSP are obtained. Second, a hybrid heuristic initialization fixing four heuristic rules is designed to provide an initial popula- tion with great convergence and diversity simultaneously. Next, four problem feature-based neighborhood structures are proposed to efficiently improve the success rate of local searches to increase convergence. Then, an elite strategy is proposed to improve the usage of historical elite solutions to enhance diversity. Finally, the results of numerical experiments show that KPMA is signifi- cantly superior to the five comparison algorithms in terms of obtaining the Pareto solutions with better convergence and diversity.

Some future tasks are discussed following: i) adopt a learning schema to KPMA to increase its intelligence; ii) consider a multi- population co-evolution framework to increase convergence; and

iii) consider multiple operations to extend DHUPMSP.

CRediT authorship contribution statement

Hua Wang: Resources, Project administration, Software, Dat- acuration, Writing - original draft, Writing - review & editing. Rui Li: Resources, Software, Data curation, Writing - original draft, Writing - review & editing. Wenyin Gong: Funding acquisition, Conceptualization, Methodology, Writing - review & editing.

Acknowledgement

This work was partly supported by the National Key Research and Development Program of China under Grant No. 2020YFB1712102.

References

1. [Wang L-C, Chen C-C, Liu J-L, Chu P-C. Framework and deployment of a cloud-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0005) [based advanced planning and scheduling system. Robot Comput-Integrated](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0005) [Manuf 2021;70: 102088](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0005).
2. [Elsherbiny S, Eldaydamony E, Alrahmawy M, Reyad AE. An extended intelligent](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0010) [water drops algorithm for workflow scheduling in cloud computing](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0010) [environment. Egypt Inform J 2018;19:33–55](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0010).
3. [Li R, Gong W, Lu C, Wang L. A learning-based memetic algorithm for energy-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0015) [efficient flexible job shop scheduling with type-2 fuzzy processing time. IEEE](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0015) [Trans Evol Comput 2022. 1–1](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0015).
4. [Li R, Gong W, Lu C. Self-adaptive multi-objective evolutionary algorithm for](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0020) [flexible job shop scheduling with fuzzy processing time. Comput Ind Eng](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0020) [2022;168: 108099](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0020).
5. [Huang K, Li R, Gong W, Wang R, Wei H. Brce: bi-roles co-evolution for energy-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0025) [efficient distributed heterogeneous permutation flow shop scheduling with](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0025) [flexible machine speed. Complex Intell Syst 2023](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0025).
6. [Cevikcan E, Durmusoglu MB. An integrated job release and scheduling](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0030) [approach on parallel machines: An application in electric wire-harness](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0030) [industry. Comput Ind Eng 2014;76:318–32](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0030).
7. [Rivera G, Porras R, Sanchez-Solis JP, Florencia R, García V. Outranking-based](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0035) [multi-objective pso for scheduling unrelated parallel machines with a freight](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0035) [industry-oriented application. Eng Appl Artif Intell 2022;108: 104556](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0035).
8. [Wang J, Song G, Liang Z, Demeulemeester E, Hu X, Liu J. Unrelated parallel](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0040) [machine scheduling with multiple time windows: An application to earth](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0040) [observation satellite scheduling. Comput Oper Res 2023;149: 106010](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0040).
9. [Hatami S, Ruiz R, Andrés-Romano C. Heuristics for a distributed parallel](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0045) [machine assembly scheduling problem with eligibility constraints, in.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0045) [International Conference on Industrial Engineering and Systems](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0045) [Management (IESM) 2015;2015:145–53](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0045).
10. [Lu C, Gao L, Yi J, Li X. Energy-efficient scheduling of distributed flow shop with](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0050) [heterogeneous factories: A real-world case from automobile industry in China.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0050) [IEEE Trans Industr Inf 2021;17:6687–96](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0050).
11. [Lu C, Zhang B, Gao L, Yi J, Mou J. A knowledge-based multiobjective memetic](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0055) [algorithm for green job shop scheduling with variable machining speeds. IEEE](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0055) [Syst J 2022;16:844–55](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0055).
12. [Zhao F, Ma R, Wang L. A self-learning discrete jaya algorithm for](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0060) [multiobjective energy-efficient distributed no-idle flow-shop scheduling](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0060) [problem in heterogeneous factory system. IEEE Trans Cybern 2021:1–12](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0060).
13. [Zhao F, He X, Wang L. A two-stage cooperative evolutionary algorithm with](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0065) [problem-specific knowledge for energy-efficient scheduling of no-wait flow-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0065) [shop problem. IEEE Trans Cybern 2021;51:5291–303](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0065).
14. [Li R, Gong W, Wang L, Lu C, Jiang S. Two-stage knowledge-driven evolutionary](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0070) [algorithm for distributed green flexible job shop scheduling with type-2 fuzzy](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0070) [processing time. Swarm Evolut Comput 2022;101139](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0070).
15. [Lei D, Liu M. An artificial bee colony with division for distributed unrelated](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0075) [parallel machine scheduling with preventive maintenance. Comput Ind Eng](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0075) [2020;141: 106320](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0075).
16. [Pan Z, Lei D, Wang L. A knowledge-based two-population optimization](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0080) [algorithm for distributed energy-efficient parallel machines scheduling. IEEE](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0080) [Trans Cybern 2022;52:5051–63](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0080).
17. [Chen X, Ong Y-S, Lim M-H, Tan KC. A multi-facet survey on memetic](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0085) [computation. IEEE Trans Evol Comput 2011;15:591–607](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0085).
18. [Selvi S, Manimegalai D. Multiobjective variable neighborhood search](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0090) [algorithm for scheduling independent jobs on computational grid. Egypt](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0090) [Inform J 2015;16:199–212](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0090).
19. [Su Y, Li Y, Xuan S. Prediction of complex public opinion evolution based on](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0095) [improved multi-objective grey wolf optimizer. Egypt Inform J](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0095) [2023;24:149–60](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0095).
20. [Kalra M, Singh S. A review of metaheuristic scheduling techniques in cloud](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0100) [computing. Egypt Inform J 2015;16:275–95](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0100).
21. [Zhang G, Liu B, Wang L, Yu D, Xing K. Distributed co-evolutionary memetic](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0105) [algorithm for distributed hybrid differentiation flowshop scheduling problem.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0105) [IEEE Trans Evol Comput 2022;26:1043–57](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0105).
22. [Wang J-J, Wang L. A cooperative memetic algorithm with learning-based agent](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0110) [for energy-aware distributed hybrid flow-shop scheduling. IEEE Trans Evol](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0110) [Comput 2022;26:461–75](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0110).
23. [Lu C, Huang Y, Meng L, Gao L, Zhang B, Zhou J. A pareto-based collaborative](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0115) [multi-objective optimization algorithm for energy-efficient scheduling of](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0115) [distributed permutation flow-shop with limited buffers. Robot Comput-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0115) [Integrated Manuf 2022;74: 102277](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0115).
24. [Zhang Q, Hui L. MOEA/D: A multiobjective evolutionary algorithm based on](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0120) [decomposition. IEEE Trans Evol Comput 2007;11:712–31](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0120).
25. [Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0125) [genetic algorithm: Nsga-ii. IEEE Trans Evol Comput 2002;6:182–97](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0125).
26. [Pei Z, Wan M, Jiang ZZ, Wang Z, Dai X. An approximation algorithm for](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0130) [unrelated parallel machine scheduling under tou electricity tariffs. IEEE Trans](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0130) [Autom Sci Eng 2021;18:743–56](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0130).
27. [Cheng J, Chu F, Zhou M. An improved model for parallel machine scheduling](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0135) [under time-of-use electricity price. IEEE Trans Autom Sci Eng 2018;15:896–9](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0135).
28. [Fang W, Zhu H, Mei Y. Hybrid meta-heuristics for the unrelated parallel](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0140) [machine scheduling problem with setup times. Knowl-Based Syst 2022;241:](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0140) [108193](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0140).
29. [Zheng XL, Wang L. A collaborative multiobjective fruit fly optimization](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0145) [algorithm for the resource constrained unrelated parallel machine green](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0145) [scheduling problem. IEEE Trans Syst, Man, Cybern: Syst 2018;48:790–800](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0145).
30. [Ding J, Shen L, Lü Z, Xu L, Benlic U. A hybrid memetic algorithm for the parallel](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0150) [machine scheduling problem with job deteriorating effects. IEEE Trans](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0150) [Emerging Topics Comput Intell 2020;4:385–97](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0150).
31. [Wang MZ, Zhang LL, Choi TM. Bi-objective optimal scheduling with raw](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0155) [material’s shelf-life constraints in unrelated parallel machines production.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0155) [IEEE Trans Syst, Man, Cybern: Syst 2020;50:4598–610](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0155).
32. [Mecler D, Abu-Marrul V, Martinelli R, Hoff A. Iterated greedy algorithms for a](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0160) [complex parallel machine scheduling problem. Eur J Oper Res](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0160) [2022;300:545–60](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0160).
33. [Wang H, Alidaee B. Unrelated parallel machine selection and job scheduling](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0165) [with the objective of minimizing total workload and machine fixed costs. IEEE](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0165) [Trans Autom Sci Eng 2018;15:1955–63](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0165).
34. [Cao Z, Lin C, Zhou M, Zhou C, Sedraoui K. Two-stage genetic algorithm for](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0170) [scheduling stochastic unrelated parallel machines in a just-in-time](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0170) [manufacturing context. IEEE Trans Autom Sci Eng 2022:1–14](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0170).
35. [Chen H, Guo P, Jimenez J, Dong ZS, Cheng W. Unrelated parallel machine](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0175) [photolithography scheduling problem with dual resource constraints. IEEE](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0175) [Trans Semicond Manuf 2023;36:100–12](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0175).
36. [Wang S, Wu R, Chu F, Yu J. Unrelated parallel machine scheduling problem](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0180) [with special controllable processing times and setups. Comput Oper Res](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0180) [2022;148: 105990](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0180).
37. Y.Y. Huang, B. Qian, R. Hu, Z.Q. Zhang, X.H. Zhu, Hybrid eda for solving distributed heterogeneous parallel machine scheduling problem, in: 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC), 2018, pp. 830–834.
38. [Zhou T, Zhang Q, Wang X, Ren X. Imperialist competitive algorithm based on](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0190) [vnsobl optimization for distributed parallel machine scheduling problem.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0190) [Chinese Automation Congress (CAC) 2019;2019:5717–23](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0190).
39. [Münch L, Shen L. Parallel machine scheduling with the total weighted delivery](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0195) [time performance measure in distributed manufacturing. Comput Oper Res](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0195) [2021;127: 105126](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0195).
40. [Zhang G, Liu B, Wang L, Xing K. Distributed heterogeneous co-evolutionary](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0200) [algorithm for scheduling a multistage fine-manufacturing system with setup](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0200) [constraints. IEEE Trans Cybern 2022:1–14](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0200).
41. [Lei C, Zhao N, Ye S, Wu X. Memetic algorithm for solving flexible flow-shop](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0205) [scheduling problems with dynamic transport waiting times. Comput Ind Eng](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0205) [2020;139: 105984](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0205).
42. [Abedi M, Chiong R, Noman N, Zhang R. A multi-population, multi-objective](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0210) [memetic algorithm for energy-efficient job-shop scheduling with](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0210) [deteriorating machines. Expert Syst Appl 2020;157: 113348](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0210).
43. [Kurdi M. A memetic algorithm with novel semi-constructive evolution](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0215) [operators for permutation flowshop scheduling problem. Appl Soft Comput](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0215) [2020;94: 106458](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0215).
44. [Shao W, Shao Z, Pi D. A network memetic algorithm for energy and labor-](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0220) [aware distributed heterogeneous hybrid flow shop scheduling problem.](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0220) [Swarm Evolut Comput 2022;75: 101190](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0220).
45. [Li R, Gong W, Lu C. A reinforcement learning based rmoea/d for bi-objective](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0225) [fuzzy flexible job shop scheduling. Expert Syst Appl 2022;203: 117380](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0225).
46. [Wang Y-N, Wu L-H, Yuan X-F. Multi-objective self-adaptive differential](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0230) [evolution with elitist archive and crowding entropy-based diversity](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0230) [measure. Soft Comput 2010;14:193–209](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0230).
47. [While L, Hingston P, Barone L, Huband S. A faster algorithm for calculating](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0235) [hypervolume. IEEE Trans Evol Comput 2006;10:29–38](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0235).
48. [Van Nostrand RC. Design of experiments using the taguchi approach: 16 steps](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0240) [to product and process improvement. Technometrics 2002;44. 289–289](http://refhub.elsevier.com/S1110-8665(23)00031-2/h0240).