Highlights

Minimizing tardiness and makespan for distributed heterogeneous unrelated parallel machine scheduling by knowledge and Pareto-based memetic algorithm

Hua Wang, Rui Li, Wenyin Gong

- A DHUPMSP with minimizing both makespan and TTD is considered.
- Four problem features-based heuristic initialization rules are proposed.
- Four knowledge-based heuristic neighborhood structures are designed.
- KPMA gets better results for solving DHUPMSP than other methods.

Minimizing tardiness and makespan for distributed heterogeneous unrelated parallel machine scheduling by knowledge and Pareto-based memetic algorithm

Hua Wang[a], Rui Li[b], Wenyin Gong[b,*]

[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*

Abstract

This work aims to deal with the distributed heterogeneous unrelated parallel machine scheduling problem (DHUPMSP) with minimizing total tardiness (TDD) and makespan. To solve this complex combinatorial 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 population with great convergence and diversity; 2) four problem feature-based heuristic neighborhood structures 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 compared to five state-of-art and run on 20 instances with different scales. The results of numerical experiments 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.

Keywords: Distributed heterogeneous factory, unrelated parallel machine scheduling, memetic algorithm, knowledge-based heuristic strategies, multi-objective optimization.

1. Introduction

Nowadays, traditional manufacturing is emergent to update and transfer the artificial production scheduling method to intelligent advanced planning and scheduling (APS) system [1, 2]. The core of APS is mod-

*Corresponding author: Wenyin Gong
Email addresses: wanghua@njtech.edu.cn (Hua Wang), liruicug@163.com (Rui Li), wygong@cug.edu.cn (Wenyin Gong)

eling and simulating complex production systems and solving problems by intelligent optimization algorithms [3, 4, 5]. The parallel machine scheduling problem (PMSP) is one of the classical combinatorial optimization problems in APS which is applied in different areas of manufacturing systems including the electric wire-harness industry [6], freight system [7], and earth observation satellite scheduling [8]. 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 parallel 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 economic globalization and the growth of global trade volume, conventional single-factory UPMSP cannot satisfy the fast manufacturing requirement. The enterprise has to reduce the original 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]. 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 manufacturing [10]. Thus, the distributed heterogeneous shop scheduling problems start to be an emergency research topic such as distributed heterogeneous hybrid flow shop scheduling [10], job shop scheduling [11], flow shop scheduling [12, 13], and flexible job shop scheduling (FJSP) [14]. Due to its difficulty and complexity, there are a few works for distributed heterogeneous UPMSP (DHUPMSP) [15, 16]. 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 flexibility. Thus, how to efficiently solve DHUPMPS becomes an emerging topic.

The term memetic algorithms (MAs) is a combination of a population-based evolutionary algorithm and one or more local refinement strategies [17, 18, 19]. 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]. As for multi-objective shop scheduling, the MAs can be classified into two types which are the decomposition-based MAs [22] and Pareto-based MAs [23]. The decompos-ition-based MAs are based on the theory of MOEA/D [24] and the Pareto-based MAs are originated from the framework of NSGA-II [25]. 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 heterogeneous unrelated parallel machine scheduling problem with minimizing 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 proposed for DHUPMSP. The main contributions of this work are summarized below:

1) A DHUPMSP with minimizing makespan and TTD simultaneously 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. The problem description and MILP model of DHUPMSP are introduced in Section 3. Section 4 illustrates our approach KPMA. The results of detailed numerical experiments are demonstrated in Section 5. Finally, the conclusion of this study and some future directions are stated in Section 6.

2. Literature Review

2.1. 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]. Chen studied an extension model of UPMSP which considered the time-of-use electricity price [27]. Chen proposed its MILP model and applied CPLEX optimizer to solve it. Fang designed an adaptive large neighborhood search-based tabu search for UPMSP and combined a learning based automata to improve the efficiency of local searches [28]. Zheng proposed a collaborative fruit fly algorithm for UPMSP with resource constraints and obtained better results than compared algorithm [29]. Ding studied UPMSP with job deteriorating effects and designed a hybrid memetic algorithm for it [30]. Wang developed an evolutionary discrete particles swarm optimizer for UPMSP and the proposed local search strategies greatly improved the convergence of the algorithm [31]. An iterated greedy method was proposed in [32] 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]. Cao proposed a

Algorithm 1: The Framework of MA.

```
Input: Maximum number of function evaluations (MaxNFEs), population size (ps), probability of crossover (p_c), and probability of mutation (p_m)
Output: Non-dominated solutions PF
P ← Initial(ps).
F ← Fitness (P).
t = 1.
while NFEs ⩽ MaxNFEs do
    P_{t+1} ← GeneticOperator(P_t, p_c, p_m).
    while the stop critera are met do
        Apply local search to generate neighborhood solutions to update P_{t+1}.
    t = t + 1.
PF ← Get Pareto Front (P_t).
```

two-phase based memetic algorithm for stochastic UPMSP [34]. Chen studied UPMSP with dual resource constraints and proposed the genetic algorithm with a local search for it [35]. Wang researched UPMSP with controllable processing times and designed logic-based Benders decomposition for rapidly solving it [36].

2.2. Related Works of Distributed UPMSP

DUPMSP aims to parallelly produce all jobs in multiple distributed factories which is harder to solve than UPMSP. Nevertheless, there are few works for DUPMSP due to its complexity. Hatami proposed four fast and high-performing heuristics for DUPMSP [9]. In [37], 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]. Lei developed a division strategy based artificial bee colony algorithm for DHUPMSP with preventive maintenance and the division strategy can efficiently accelerate convergence [15]. Mnch studied DHUPMSP with the total weighted delivery time [39]. Pan designed a two-population co-evolution based on knowledge and feedback for green DHUPMSP [16].

2.3. 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 problems. Zhang proposed a MA for scheduling hybrid differentiation flowshop to enhance the convergence of co-evolution [40]. Ding applied a hybrid MA for PMSP and got a 100% success rate to update neighborhood solutions [30]. Lei developed a MA for hybrid flow shop scheduling (HFS) which can obtain solutions with great objectives values in short time [41]. Abedi designed a multi-population MA for job-shop scheduling and outperformed other algorithms [42]. Kurdi proposed a semi-constructive crossover based MA with mutation operators for flowshop scheduling and improved the metrics by 37.92% [43]. Shao designed a MA for distributed heterogeneous HFS and obtained the best results [44].

2.4. 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 previous 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 computation resources to converge.

3) The previous works lack elite strategy and the historical elite solutions are always abandoned due to the limited population 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.

3. Problem and Model Description

3.1. Description for DHUPMSP

As for DHUPMSP, each order is regarded as a job. A instance of DHUPMSP has n jobs and n_f heterogeneous factories. Every factory has m unrelated parallel machine and every job has only one stage. Each job's processing time $T_{f,i,k}$ 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 D_i. The main goal is to dispatch n job to n_f 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 (C_{max}).

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' processing times are certain; and v) one job can only be processed by one machine at the same time and cannot be interrupted during processing; Moreover, dynamic events such as machine breakdown, preventive maintenance are not considered.

3.2. MILP Model for DHUPSMP

The notations of DHUPMSP are introduced below:

Decision variables:

- $\mathbf{Y}_{i,f}$: The binary value is set to one when job I_i is allocated to factory f; Otherwise, the value equals zero;
- $\mathbf{X}_{f,i,k,t}$: The binary value is set to one when job I_i is dispatched to the position t of machine M_k in factory f; Otherwise, the value equals zero;
- $C_{f,k,t}$: the completion time of t_{th} position of machine M_k in factory f;
- $F_{f,i}$: the finishing time of job I_i in factory f;
- $B_{f,k,t}$: the beginning time of machine M_k at position t in factory f;
- $S_{f,i}$: the starting time of job I_i in factory f;

Parameters:

- n_f: the number of factories;
- F: set of factories and $F = \{1, 2, ..., n_f\}$;
- 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 $I = \{1, 2, ..., n\}$;
- n_t: the number of all positions;
- $Z_{f,k}$: positions set on machines M_K in factory f and $Z_{f,k} = \{1, 2, ..., n_t\}$;
- $Z'_{f,k}$: top $n_t - 1$ positions set on machines M_K in factory f and $Z_{f,k} = \{1, 2, ..., n_t - 1\}$;
- $T_{f,i,k}$: The processing time job I_i processed by machine M_k in factory f;

- D_i: The duedate of job I_i;

- 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;

The objectives of DHUPMSP in this work are TTD and C_{max}, which are elaborated as follows:

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

$$TTD = \sum_{i=1}^{n} \sum_{f=1}^{n_f} \max\left\{F_{f,i} \cdot \mathbf{Y}_{i,f} - D_i, 0\right\}. \tag{1}$$

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 makespan criteria are stated below:

$$C_{max} = \max\{F_{f,i}\}, \forall f \in F, i \in I. \tag{2}$$

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

$$\begin{cases} \min \ F_1 = TTD \\ \min \ F_2 = C_{max} \end{cases} \tag{3}$$

subject to:

$$\sum_{f \in F} \sum_{k \in M} \mathbf{X}_{f,i,k,t} \leqslant 1, \forall i \in I, t \in Z_{f,k} \tag{4}$$

$$\sum_{i \in I} \sum_{k \in M} \mathbf{X}_{f,i,k,t} \geq \sum_{i \in I} \sum_{k \in M} \mathbf{X}_{f,i,k,t+1}, \forall f \in F, t \in Z'_{f,k} \tag{5}$$

$$\sum_{f \in F} \mathbf{Y}_{i,f} = 1, \forall i \in I \tag{6}$$

$$S_{f,i} + \sum_{t \in Z_{f,k}} T_{f,i,k} \cdot \mathbf{X}_{f,i,k,t} \leqslant F_{f,i}, \forall i \in I, k \in M, f \in F \tag{7}$$

$$B_{f,k,t+1} - B_{f,k,t} \geq \sum_{i \in I} \sum_{k \in M} \mathbf{X}_{f,i,k,t} \cdot T_{f,i,k}, \forall f \in F, t \in Z'_{f,k} \tag{8}$$

$$B_{f,k,t} = S_{f,i} \cdot \mathbf{X}_{f,i,k,t}, \forall i \in I, k \in M, f \in F, t \in Z_{f,k} \tag{9}$$

$$0 \leqslant S_{f,i}, B_{f,k,t} \leqslant L, \forall i \in I, k \in M, f \in F, t \in Z_{f,k} \tag{10}$$

where Eq. (3) are objective functions which are TTD and C_{max}. Eq. (4) ensures that each job can not be processed on two different machines at the same time. Eq. (5) guarantees that each position of a machine is available only when its preceding position is selected. Eq. (6) makes sure a job only be dispatched to one factory. Eq. (7) states the relationship between the start time and finish time of a job. Eq. (8) guarantees the correction of the beginning time between two adjacent positions. Eq. (9) ensures the constrain between operation start time and machine start time. Eq.(10) is values' boundaries.

3.3. 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 $F_t = F_{t-1} + P_i$. The makespan depends on the completion time of the last finished job. Thus, $C_{max} = F_{last} = \sum_{t=1}^{n_t} P_i * \mathbf{X}_{f,i,k,t}, \forall i \in I, f \in F, k \in M$. Thus, the makespan 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 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 finish 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. $F_2 = F_1 + P_2$ and $F_1 = F_0 + P_1$. Moreover, $F_2 = F_0 + P_1 + P_2$. The tardiness of two jobs is $T_1 = F_1 - D_1$ and $T_2 = F_2 - D_2$.

The total tardiness 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 process I_2 ahead. The new finish 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 $F_3 = F_4 + P_1 = F_0 + P_1 + P_2$. Next, the new tardiness $T_3 = F_3 - D_1$ and $T_4 = F_4 - D_2$. The total tardiness $TDD' = T_0 + T_3 + T_4 = T_0 + 2 * F_0 + 2 * P_2 + P_1 - D1 - D2$. $\Delta TDD = TDD' - TDD = P_2 - P_1 < 0$. 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.

4. Our Approach: KPMA

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

4.1. Motivation

Based on the research gap mentioned in Section 2.4 and problem features stated in Section 3.3, this work proposed a knowledge 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 construct 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 computation 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.

4.2. Framework of KPMA

Algorithm 2 states the framework of KPMA. First, KPMA also initialized two swarms $\mathcal{P}$ and C. Meanwhile, $\mathcal{P}$ is initialized by a hybrid heuristic initialization to get great convergence and diversity simultaneously. Second, $\mathcal{P}$ will execute NSGA-II [25] for global search. Moreover, after the environmental selection, C obtains non-dominated solutions from $\mathcal{P}$. 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.

4.3. Encoding and Decoding

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

Algorithm 2: The Framework of KPMA.

Input: Maximum number of function evaluations ($MaxNFEs$), population size (ps), crossover rate (P_c), mutation rate (P_m), learning rate α, discount factor γ, greedy factor ϵ
Output: Non-dominated solutions PF
$\mathcal{P}_0 \leftarrow$ Heuristic Initial(ps). //Initial population
$C \leftarrow \varnothing$. //Initial elite archive
$F \leftarrow$ Decoding ($\mathcal{P}_0$). //Get fitness
$t = 1$.
while $NFEs \leqslant MaxNFEs$ **do**
    $Pool \leftarrow$ TournamentSelection($\mathcal{P}_t$).
    $\mathcal{H}_1 \leftarrow$ Fast non-dominated sort($\mathcal{P}$, $Pool$).
    $\mathcal{H}_2 \leftarrow$ CrowdingDistance($\mathcal{H}_1$).
    $NFEs \leftarrow NFEs + 2 * ps$.
    $Q_t \leftarrow \mathcal{H}_2 \cup \mathcal{P}_t$.
    $\mathcal{P}_{t+1} \leftarrow$ Environmental selection (Q_t, ps).
    $F_0 \leftarrow$ Get Pareto Front($\mathcal{P}_{t+1}$).
    $C_{t+1} \leftarrow C_t \cup F_0$.
    $C_{t+1} \leftarrow$ Delete duplicates (C_{t+1}).
    for $i = 1$ *to* ps **do**
        $T \leftarrow$LocalSearch ($\mathcal{P}_{t+1}(i)$).
        if $T \prec \mathcal{P}_{t+1}(i)$ **then**
            $\mathcal{P}_{t+1}(i) \leftarrow T$.
            $C_{t+1} \leftarrow C_{t+1} \cup T$
        else
            if $T \succ \mathcal{P}_{t+1}(i)$ **then**
                continue
            else
                $C_{t+1} \leftarrow C_{t+1} \cup T$
    $t = t + 1$.
$PF \leftarrow$ Get Pareto Front (C_t).

order is from J_1 to J_n 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 $T_{f,i,k}$ can be got. Then, the start and completion time of every job is calculated. Furthermore, the max finish time can be obtained and C_{max} is got. Finally, if a job's finish time is bigger than its due date D_i, the tardiness will be summarized and the TTD can be got.

4.4. Hybrid Heuristic Initialization

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

MFW rule: The MFW rule focuses to balance the factory workload. Due to problem features, reducing the job number gap can efficiently reduce the finish time gap between each factory. The MFW rule is described

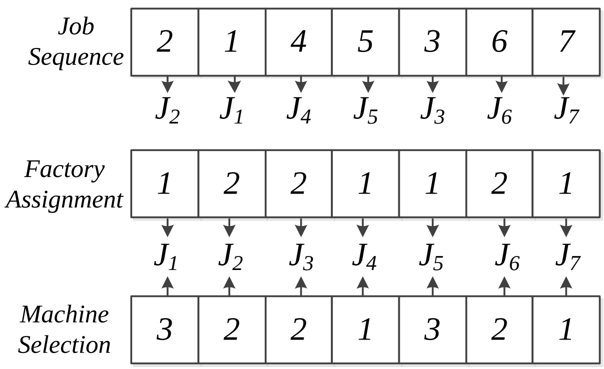


Figure 1: An example for encoding schema for DHUPMSP

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

Input: Job sequence (JS), factory assignment (FA), job number (n), processing time (T).

Output: Machine Selection MS

$MS \leftarrow$ Zeros(n).

for $i = 1$ *to* N **do**

  $J \leftarrow JS(i)$.

  $F \leftarrow FA(i)$.

  for $k = 1$ *to* m **do**

    $P_k \leftarrow T_{F,J,k}$

  $MS(i) \leftarrow$ machine index with min ($T_{F,J,k}$).

---

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 5. 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 finish 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 requirement to reduce tardiness. The EDD rule is stated in Algorithm 6. First, compare the due date of each job and process the job with the earlier due date ahead. Second, repeat the step until the job sequence is determined.

Hybrid heuristic initialization: To keep great convergence and diversity simultaneously, the whole pop-

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

Input: Machine selection (MS), job number (n), factory number (n_f), processing time (T).
Output: Factory assignment FA
$FA \leftarrow$ Zeros(N).
for $f = 1$ *to* n_f **do**
  for $i = 1$ *to* n **do**
    $ms \leftarrow MS(i)$.
    $R(f, i) \leftarrow T_{f,i,ms}$
$C \leftarrow$ zero(n_f).
for $i = 1$ *to* n **do**
  $r \leftarrow R(f, :)$.
  $r\_index \leftarrow sort(r)$.
  $AJ \leftarrow mean(C)$.
  for $f = 1$ *to* n_f **do**
    if $C(r(r\_index(f))) \leqslant AJ$ **then**
      $sf \leftarrow f$.
  $FA(i) \leftarrow sf$.
  $C(sf) \leftarrow C(sf) + 1$.

ulation 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.

4.5. 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]. First, the two-player tournament selection is applied to select the mating pool. Then, universal crossover (UX) [45] and precedence operation crossover (POX) [14] are adopted to generate offspring which are shown in Fig. 2 and Fig. 3. 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

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

Input: Job sequence (JS), factory assignment (FA), job number (n), machine number m, processing time (T).
Output: Machine selection MS
$MS \leftarrow$ Zeros(N).
for $i = 1$ *to* n **do**
    $P \leftarrow P \cup JS(i)$
for $f = 1$ *to* n_f **do**
    $N \leftarrow len(P)$.
    $F \leftarrow zeros(N)$.
    $S \leftarrow zeros(N)$.
    $C \leftarrow zeros(m)$.
    for $i = 1$ *to* N **do**
        $ms \leftarrow$ index with min C.
        $MS(P(i)) \leftarrow ms$.
        $S(i) \leftarrow C(ms)$.
        $F(i) \leftarrow T_{f,ms,P(i)} + S(i)$.
        $C(ms) \leftarrow F(i)$.

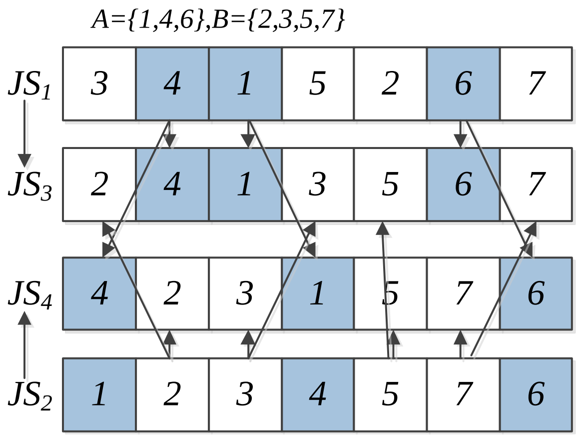


Figure 2: POX for JS

from B in JS_2 are copied to the same positions in JS_4. Next, the jobs in B are 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 random 0-1 vector R is generated first which size n. Then, traverse the parents from left to right. If the value of R is

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

Input: Due date (D), job number (n).
Output: Job sequence (JS)
$I \leftarrow (1 : n)$.
for $i = 1$ *to* n **do**
    for $j = 1$ *to* n **do**
        if $D_i > D_j$ **then**
            $t \leftarrow I_i$.
            $I_i \leftarrow I_j$.
            $I_j \leftarrow t$.
$JS \leftarrow I$

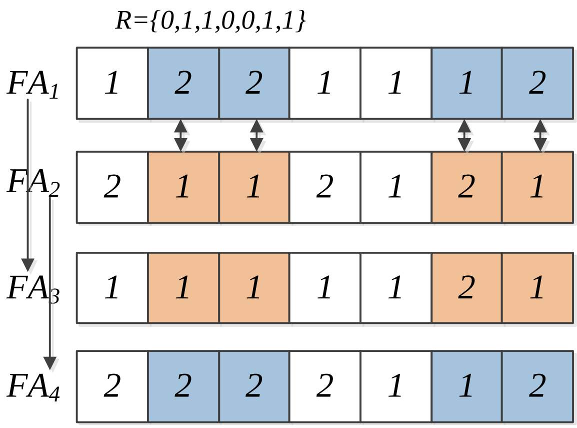


Figure 3: UX for FA and MS

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, randomly choose two jobs and exchange their positions. For FA vector, randomly choose a job and move it to another heterogeneous factory. For MS vector, randomly choose a job and move it to another parallel machine. The mutation operators are shown in Fig. 4. Finally, the child solutions obtained by the evolution operators are merged with the parent population $\mathcal{P}$. The combined swarm is chosen by the crowding distance strategy and fast non-dominated sorting to generate the population of next generation [25].

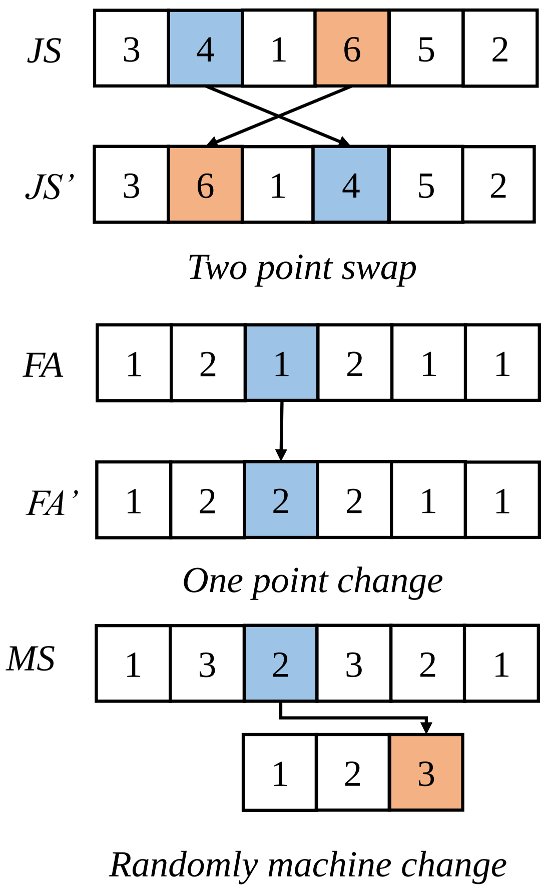


Figure 4: Mutation for JA, MS and FA

4.6. Knowledge-based Local Reinforcement

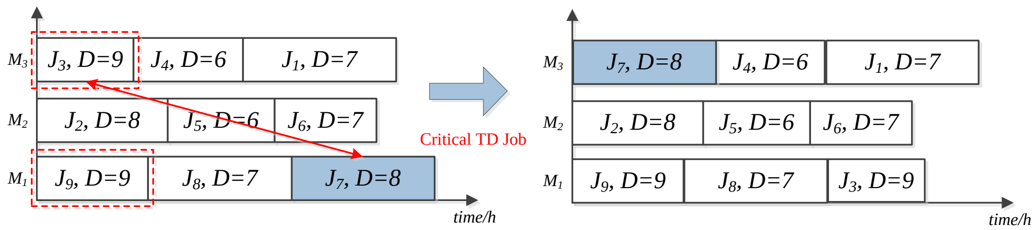
Designing problem features based local search strategies can greatly increase their efficiency for solving shop scheduling problems. According to the problem features of DHUPMSP, four neighborhood structures are developed which are introduced below:

4.6.1. $\mathcal{N}_1$ (Swap for TDD)

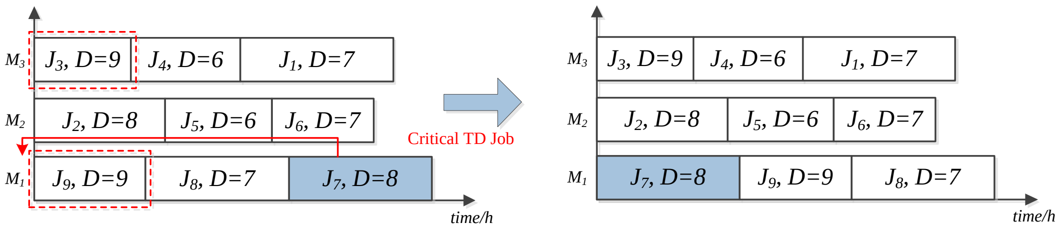
Fig 5 shows the procedure of N1. First, find the job with the max tardiness of the all factories. Second, traverse the jobs in front of the critical job. If there is a job that has a bigger due date, swap the machine selection and the job sequence of the critical job and selected job.

4.6.2. $\mathcal{N}_2$ (Insertion for TDD)

Fig 5 shows the procedure of $\mathcal{N}_2$. 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.

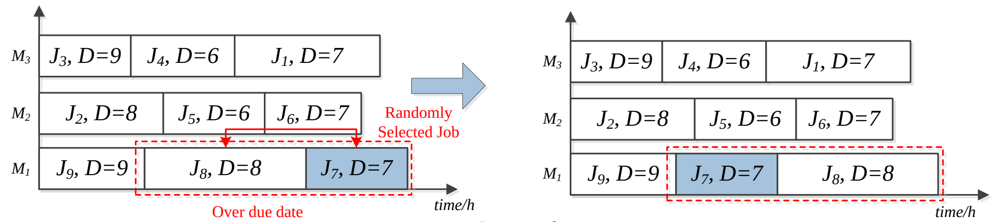


(N_1) Swap for TTD

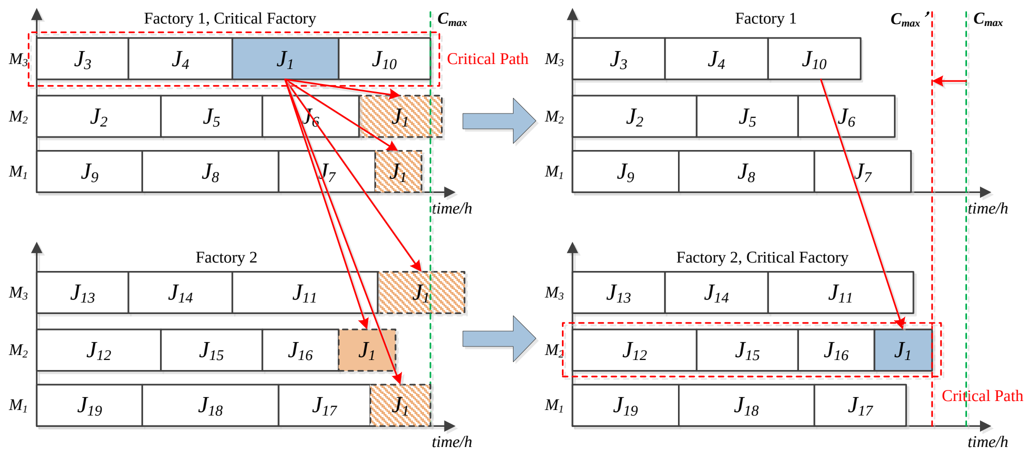


(N_2) Insertion for TTD

Figure 5: An example for encoding schema for DHUPMSP



(N_3) Critical Swap for TTD



(N_4) Critical Machine Change for Makespan

Figure 6: An example for encoding schema for DHUPMSP

4.6.3. $\mathcal{N}_3$ (Critical swap for TDD)

This neighborhood structure is designed based on problem *feature 3*. Fig 6 shows the procedure of $\mathcal{N}_3$. 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.

4.6.4. $\mathcal{N}_4$ (Critical machine change for C_{max})

This neighborhood structure is designed based on problem *feature 1* and *feature 2*. Fig 6 shows the procedure of $\mathcal{N}_4$. 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.

4.7. Elite Strategy

Based on the research gap in Section 2.4, this work proposed a simple elite strategy to increase the diversity. During the evolutionary stage and local search stage, store the non-dominated solutions to the elite archive. Then, the elite archive only keeps its non-dominated solutions at the end of every generation.

5. Results of Numerical Experiment

In Section 4, Our approach KPMA is illustrated detailedly. Moreover, 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 3.4GHz and 384G RAM. Moreover, the running environment is Pycharm2021 with python3.8.

5.1. 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 $n_f \in \{2, 3\}$ and the job number ranges from $n \in \{20, 40, 60, 80, 100\}$. The processing time $T_{f,i,k}$ is from $\{5, 95\}$ which is different in every heterogeneous factories and the machines amount n_m $= \in$ $\{4, 6\}$. . The duedate D_i ranges from average processing time added and minus 5 or 10. Finally, 20 test problems with several scales are created which are named as $20J4M2F$. The stop criteria is set to MaxNEFs=$400 * n \geqslant 2 * 10^4$.

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]

$$GD(P, P^*) = \frac{\sqrt{\sum_{y \in P} \min_{x \in P^*} d(x, y)^2}}{|P|} \tag{11}$$

In Eq. 11, 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 $y \in P^*$ and $x \in P$. Furthermore, an algorithm with better convergence has smaller GD metric value.

Diversity metric: Spread [46]

$$\begin{aligned} Spread(P, P^*) &= \frac{\sum_{i=1}^{|P^*|} d(P, P^*) + \sum_{X \in P} |d(X, P) - \bar{d}|}{\sum_{i=1}^{|P^*|} d(P, P^*) + (|P| - |P^*|)\bar{d}}, \\ d(X, P) &= min_{Y \in P, Y \neq X} \|F(X) - F(Y)\|, \\ \bar{d} &= \frac{1}{|P|} \sum_{X \in P} d(X, P) \end{aligned} \tag{12}$$

In Eq. 12, d represents the Euclidean distance of each Pareto solution and its adjacent point. Furthermore, an algorithm with better diversity has smaller Spread metric value.

Comprehensive metric: Hypervolume (HV) [47]

$$HV(P, r) = \bigcup_{\mathbf{x} \in P}^{P} v(\mathbf{x}, r). \tag{13}$$

In Eq. 13, the notation P is the non-dominated solutions set obtained by every algorithm. Meanwhile, notation $\mathbf{x}$ 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-dominated solutions. Furthermore, an algorithm with better convergence has smaller GD metric value.

5.2. Parameter Analysis Experiment

The parameter setting seriously affects an algorithm's performance for solving DHUPMSP. The KPMA has three parameters including population size ps, crossover rate P_c, mutation rate P_m. To simplify the parameter experiment, a Taguchi method [48] is applied. Moreover, the parameters' levels are designed following: $ps = \{80, 100, 120\}$; $P_c = \{0.8, 0.9, 1.0\}$; $P_m = \{0.1, 0.2, 0.3\}$. An orthogonal design $L_9(3^3)$ 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 independent run are recorded. Figs. 7, 8, 9 show three main effects plots of all parameters. According to three plots, the optimal parameter configuration is that $ps = 80$, $P_c = 0.9$, and $P_m = 0.1$.

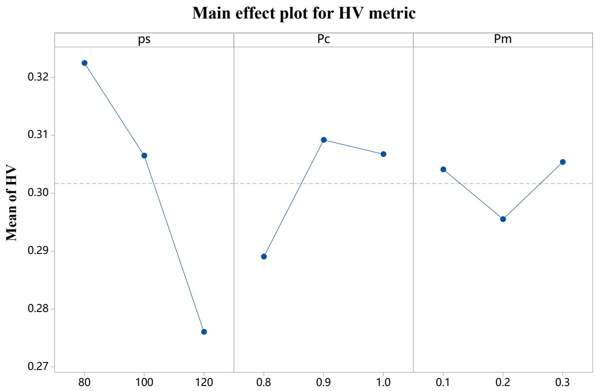


Figure 7: Main effects plot of HV metric

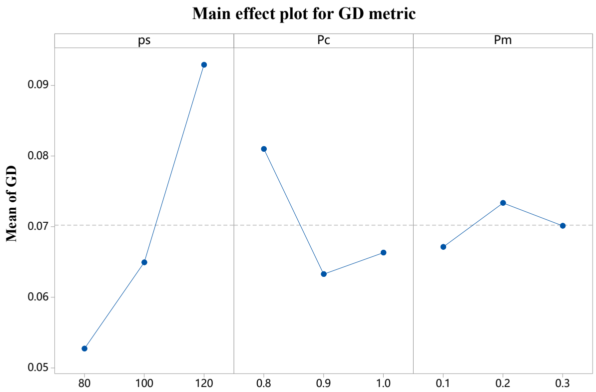


Figure 8: Main effects plot of GD metric

5.3. 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=$400*n \geqslant 2*10^4$.

Tables 2, 3, and 4 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 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-value$\leqslant 0.05$ represents that KPMA is significantly superior to all variant algorithms.

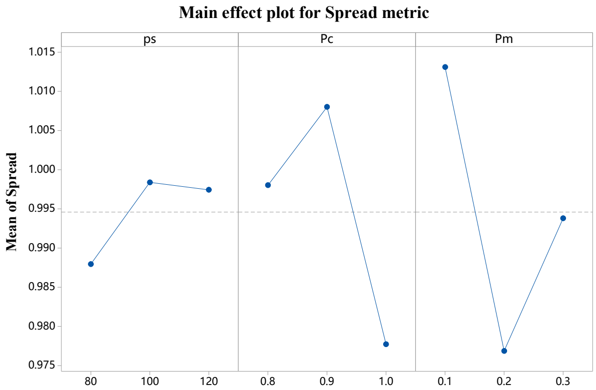


Figure 9: Main effects plot of Spread metric

Table 1: The Friedman run-and-sum test results for all variant algorithms of KPMA (significant level α=0.05)

<table>
<tr><th rowspan="2">MOEAs</th><th colspan="2">HV</th><th colspan="2">GD</th><th colspan="2">Spread</th></tr>
<tr><th>rank</th><th>p-value</th><th>rank</th><th>p-value</th><th>rank</th><th>p-value</th></tr>
<tr><td>KPMA-L</td><td>5.20</td><td rowspan="4">5.34E-11</td><td>5.20</td><td rowspan="4">1.46E-11</td><td>5.15</td><td rowspan="4">2.07E-01</td></tr>
<tr><td>KPMA-E</td><td>3.10</td><td>3.30</td><td>5.00</td></tr>
<tr><td>KPMA-I</td><td>3.35</td><td>2.85</td><td>4.15</td></tr>
<tr><td>KPMA</td><td>1.20</td><td>1.15</td><td>3.60</td></tr>
</table>

Table 2: Statistical results of HV (max) metric of all KPMA variants.

Instances	HV							
	KPMA-L		KPMA-E		KPMA-I		KPMA	
	mean	std	mean	std	mean	std	mean	std
20J4M2F	0.2202	0.0628	0.3770	0.0296	0.3447	0.0453	**0.4127**	0.0333
20J4M3F	0.2957	0.0585	0.5285	0.0784	0.5368	0.0588	**0.6451**	0.0269
20J6M2F	0.2715	0.0487	0.5662	0.0537	0.5457	0.0609	**0.6298**	0.0285
20J6M3F	0.4291	0.0828	0.6541	0.0368	0.6623	0.0213	**0.6856**	0.0140
40J4M2F	0.1233	0.0239	0.2111	0.0391	0.2169	0.0387	**0.2438**	0.0405
40J4M3F	0.2025	0.0732	0.4643	0.0687	0.4153	0.0742	**0.5277**	0.0566
40J6M2F	0.1924	0.0679	0.4466	0.0745	0.4713	0.0560	**0.4848**	0.1163
40J6M3F	0.2506	0.0689	0.4339	0.0447	0.4224	0.0760	**0.5779**	0.0642
60J4M2F	0.1450	0.0207	0.2379	0.0359	0.2269	0.0459	**0.3000**	0.0357
60J4M3F	0.1311	0.0499	0.2436	0.0599	0.2535	0.0430	**0.4146**	0.0506
60J6M2F	0.1837	0.0309	0.2355	0.0388	0.2481	0.0310	**0.2962**	0.0493
60J6M3F	0.1982	0.0458	0.2721	0.0918	0.2595	0.0653	**0.5394**	0.0578
80J4M2F	0.1202	0.0273	0.1681	0.0277	0.1758	0.0236	**0.1872**	0.0252
80J4M3F	0.0958	0.0340	0.1727	0.0345	0.1853	0.0359	**0.3261**	0.0468
80J6M2F	0.1438	0.0279	0.1579	0.0324	0.1442	0.0314	**0.2347**	0.0403
80J6M3F	0.1258	0.0433	0.2112	0.0368	0.1988	0.0760	**0.3778**	0.0617
100J4M2F	0.0705	0.0161	0.1255	0.0160	0.1155	0.0114	**0.1729**	0.0187
100J4M3F	0.1014	0.0326	0.1890	0.0339	0.1880	0.0402	**0.2725**	0.0338
100J6M2F	0.1113	0.0284	0.1714	0.0278	0.1656	0.0249	**0.1655**	0.0298
100J6M3F	0.1248	0.0357	0.1987	0.0357	0.1763	0.0536	**0.3097**	0.0554

Table 3: Statistical results of GD (min) metric of all KPMA variants.

Instances	GD							
	KPMA-L		KPMA-E		KPMA-I		KPMA	
	mean	std	mean	std	mean	std	mean	std
20J4M2F	0.2217	0.1076	0.0533	0.0338	0.0759	0.0308	**0.0480**	0.0435
20J4M3F	0.2555	0.0593	0.0689	0.0468	0.0782	0.0308	**0.0291**	0.0171
20J6M2F	0.3421	0.0655	0.0707	0.0540	0.0841	0.0527	**0.0453**	0.0296
20J6M3F	0.2221	0.0769	0.0648	0.0451	0.0525	0.0279	**0.0402**	0.0361
40J4M2F	0.1347	0.0317	0.0559	0.0300	0.0442	0.0155	**0.0354**	0.0258
40J4M3F	0.3374	0.0776	0.1432	0.0725	0.1486	0.0644	**0.0804**	0.0384
40J6M2F	0.3371	0.0913	0.0979	0.0705	**0.0806**	0.0372	0.0887	0.0854
40J6M3F	0.3333	0.1073	0.1382	0.0382	0.1286	0.0517	**0.0476**	0.0250
60J4M2F	0.1549	0.0358	0.0822	0.0279	0.0759	0.0374	**0.0307**	0.0220
60J4M3F	0.3776	0.1050	0.2024	0.0597	0.1970	0.0554	**0.0664**	0.0351
60J6M2F	0.1689	0.0571	0.1094	0.0397	0.0949	0.0309	**0.0658**	0.0458
60J6M3F	0.2937	0.0732	0.2491	0.1259	0.2183	0.0497	**0.0461**	0.0327
80J4M2F	0.0973	0.0277	0.0388	0.0223	0.0329	0.0191	**0.0219**	0.0153
80J4M3F	0.2766	0.0513	0.1607	0.0376	0.1392	0.0289	**0.0482**	0.0302
80J6M2F	0.1145	0.0287	0.1047	0.0370	0.0964	0.0327	**0.0275**	0.0277
80J6M3F	0.2860	0.0976	0.2073	0.0489	0.2084	0.0786	**0.0645**	0.0639
100J4M2F	0.1495	0.0372	0.0780	0.0182	0.0734	0.0178	**0.0219**	0.0138
100J4M3F	0.1812	0.0606	0.0856	0.0319	0.0820	0.0344	**0.0254**	0.0159
100J6M2F	0.0771	0.0316	0.0348	0.0230	0.0342	0.0186	**0.0301**	0.0168
100J6M3F	0.1989	0.0595	0.1110	0.0532	0.1168	0.0548	**0.0323**	0.0214

Table 4: Statistical results of Spread (min) metric of all KPMA variants.

	Spread							
Instances	KPMA-L		KPMA-E		KPMA-I		KPMA	
	mean	std	mean	std	mean	std	mean	std
20J4M2F	1.0072	0.0160	1.2012	0.4205	0.9759	0.2706	**0.9506**	0.1880
20J4M3F	1.0528	0.1899	1.0363	0.1444	**0.9365**	0.1620	0.9824	0.2502
20J6M2F	1.0553	0.1236	**0.9627**	0.0914	1.0136	0.1254	1.1207	0.2787
20J6M3F	1.0474	0.2331	0.9985	0.1261	**0.9906**	0.0302	1.0168	0.2957
40J4M2F	**0.9990**	0.0328	1.0740	0.2589	1.0528	0.1689	1.0696	0.3491
40J4M3F	**0.9844**	0.0383	1.0531	0.1259	0.9993	0.0464	1.1584	0.4498
40J6M2F	0.9827	0.0229	0.9830	0.0469	1.0036	0.1424	**0.9697**	0.2531
40J6M3F	**1.0000**	0.0175	1.0368	0.1044	1.0301	0.1961	1.0044	0.3186
60J4M2F	1.0003	0.0565	0.9909	0.0328	0.9534	0.0399	**0.8904**	0.1694
60J4M3F	1.0034	0.0143	0.9983	0.0624	0.9922	0.0332	**0.9924**	0.0429
60J6M2F	1.0133	0.0537	**0.9793**	0.0985	1.0321	0.1369	1.0085	0.1544
60J6M3F	1.0009	0.0338	**0.9944**	0.0273	1.0152	0.0390	1.0238	0.1169
80J4M2F	0.9830	0.0238	1.0263	0.0971	1.0345	0.1203	**0.8980**	0.1052
80J4M3F	1.0071	0.0340	0.9827	0.0361	**0.9819**	0.0403	0.9966	0.1522
80J6M2F	0.9766	0.0435	1.0120	0.0639	1.0151	0.0707	**0.9705**	0.1998
80J6M3F	0.9930	0.0178	0.9983	0.0248	**0.9433**	0.0705	0.9976	0.0890
100J4M2F	0.9916	0.0355	1.0248	0.0510	0.9740	0.0251	**0.9217**	0.1674
100J4M3F	0.9954	0.0142	1.0222	0.0796	0.9844	0.0385	**0.9722**	0.1091
100J6M2F	1.0127	0.0658	0.9777	0.1040	**0.9638**	0.1116	1.0465	0.2095
100J6M3F	1.0063	0.0309	1.0113	0.0535	0.9667	0.0573	**0.9456**	0.0844

5.4. Comparison Experiment and Discussions

In this section, KPMA is compared with NSGA-II [25] and MOEA/D [24]. Additionally, three state-of-art algorithms for DHUPMSP called DABC [15], KTPO [16] and VICA [38] are compared. The parameters of each comparison algorithms are set with the best configuration according to their references. The mutation probability p_m=0.2, crossover probability p_c=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 P_a=0.4, total imperialist countries N_{imp}=10 and revolutionary probability P_r=0.2 for VICA. To permit the fairness, all algorithms have the same stop criteria (MaxNFEs=$400 * n \geqslant 2 * 10^4$). Due to the complexity of DHUPMSP, each comparison algorithm independent runs twenty times in 20 test problems. Tables 7, 8, and 9 show the statistical results (mean and standard deviation values) of all MOEAs for HV, GD, and Spread metrics in 20 test problems. Furthermore, the notation "-" and "+" represents that the comparison algorithm is significantly worse and better than KPMA, and "=" states there is no significant difference. In addition, the optimal values are marked in **bold**. As Tables 7, 8, and 9 show, as for HV and GD metrics, KPMA is significantly superior to all compared MOEAs, which represents that KPMA has better convergence and comprehensive performance than comparison algorithms. As for the Spread metric, KPMA has no significant difference from other algorithms. Table 5 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-value$\leqslant$ 0.05 states that KPMA is significantly superior to the compared MOEAs. Table 6 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, the KPMA is significantly better than all compared algorithms on HV and GD metrics where the $p-$value$<$ 0.05. This evaluates the effectiveness 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.

Table 5: The Friedman run-and-sum test results for all comparison algorithms and KPMA (significant level α=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**	

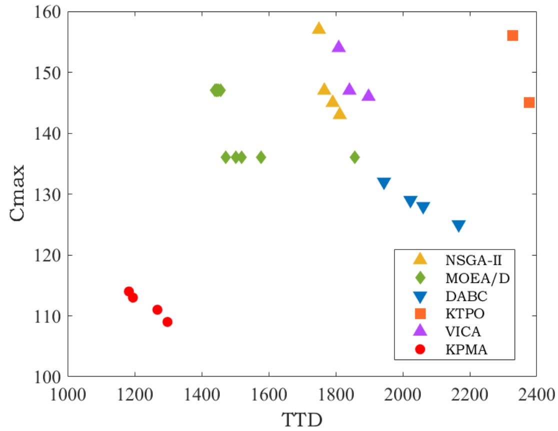


Figure 10: PF comparison results of all algorithms on 100J6M3F

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	≥ 0.2	0.985107
MOEA/D	155.0	55.0	0.06372	0.059389
DABC	120.0	90.0	≥ 0.2	0.562821
KTPO	133.0	77.0	≥ 0.2	0.287337
VICA	139.0	71.0	≥ 0.2	0.197754

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 variable neighborhood makes KPMA efficiently converge and successfully improves the efficiency of local search. Finally, the elite strategy improves the usage of historical elite solutions to increase the diversity of final Pareto solutions. Furthermore, Fig. 10 displays the Pareto Front results of all algorithms on instance 100J6M3F which are selected with the best HV metric from 20 runs. Observing the diversity and convergence of each PF, KPMA can obtain better 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. Therefore, KPMA is capable of solving DHUPMSP well.

Table 7: Statistical results of HV (max) metrics of all comparison algorithms.

	HV											
Instances	NSGA-II		MOEA/D		DABC		KTPO		VICA		KPMA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
20J4M2F	0.3577-	0.0978	0.3077-	0.1268	0.4063-	0.0533	0.4215-	0.0699	0.3204-	0.0998	**0.5702**	0.0279
20J4M3F	0.321-	0.0546	0.3657-	0.0534	0.4469-	0.0579	0.4348-	0.0751	0.292-	0.0838	**0.6550**	0.0257
20J6M2F	0.2742-	0.0508	0.2996-	0.0940	0.3791-	0.0558	0.3933-	0.1189	0.2803-	0.0560	**0.6307**	0.0278
20J6M3F	0.4409-	0.0801	0.5589-	0.1140	0.5833-	0.0206	0.5541-	0.0891	0.4465-	0.0830	**0.7167**	0.0129
40J4M2F	0.2052-	0.0381	0.2278-	0.0379	0.2535-	0.0274	0.1875-	0.0287	0.2004-	0.0248	**0.3189**	0.0417
40J4M3F	0.2435-	0.0925	0.3888-	0.0895	0.3502-	0.0786	0.3028-	0.0312	0.2123-	0.0657	**0.5396**	0.0576
40J6M2F	0.2372-	0.0401	0.2585-	0.0612	0.4069=	0.1097	0.3677-	0.0712	0.1908-	0.0466	**0.5185**	0.1091
40J6M3F	0.4246-	0.0779	0.3677-	0.0823	0.3319-	0.0810	0.3711-	0.1049	0.4093-	0.0774	**0.6580**	0.0491
60J4M2F	0.1859-	0.0392	0.2144-	0.0230	0.2562-	0.0144	0.18-	0.0267	0.1824-	0.0271	**0.3341**	0.0328
60J4M3F	0.1262-	0.0383	0.2482-	0.0288	0.227-	0.0394	0.1536-	0.0389	0.1207-	0.0323	**0.4086**	0.0507
60J6M2F	0.2012-	0.0403	0.1839-	0.0449	0.264-	0.0409	0.1859-	0.0385	0.2048-	0.0227	**0.3194**	0.0487
60J6M3F	0.1923-	0.0722	0.3011-	0.0831	0.1959-	0.0678	0.1726-	0.0606	0.1972-	0.0463	**0.5372**	0.0586
80J4M2F	0.1295-	0.0211	0.1689-	0.0213	0.1863-	0.0140	0.1172-	0.0317	0.1374-	0.0095	**0.2141**	0.0248
80J4M3F	0.1556-	0.0237	0.3016-	0.0334	0.2422-	0.0452	0.1288-	0.0278	0.1617-	0.0265	**0.4080**	0.0427
80J6M2F	0.1642-	0.0235	0.1969-	0.0273	0.2211-	0.0296	0.1195-	0.0363	0.1797-	0.0271	**0.2815**	0.0373
80J6M3F	0.2131-	0.0311	0.2948-	0.0659	0.2742-	0.0344	0.1846-	0.0691	0.2338-	0.0445	**0.4699**	0.0491
100J4M2F	0.1274-	0.0168	0.1726-	0.0201	0.1759-	0.0153	0.0795-	0.0123	0.1391-	0.0159	**0.2603**	0.0180
100J4M3F	0.1674-	0.0317	0.2421-	0.0354	0.2463-	0.0393	0.1514-	0.0252	0.1703-	0.0340	**0.3743**	0.0289
100J6M2F	0.1455=	0.0374	0.105-	0.0282	0.1471-	0.0179	0.0875-	0.0133	0.1452-	0.0267	**0.1765**	0.0311
100J6M3F	0.1772-	0.0362	0.2456-	0.0405	0.2484-	0.0450	0.1398-	0.0344	0.1893-	0.0327	**0.3757**	0.0444
-/=/+	19/1/0		20/0/0		19/1/0		19/1/0		20/0/0			

6. 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

Table 8: Statistical results of GD (min) metrics of all comparison algorithms.

Instances	GD											
	NSGA-II		MOEA/D		DABC		KTPO		VICA		KPMA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
20J4M2F	0.1996-	0.1107	0.2574-	0.1491	0.1353-	0.0550	0.1109-	0.0768	0.2149-	0.1012	**0.0334**	0.0286
20J4M3F	0.2213-	0.0725	0.2003-	0.0660	0.1636-	0.0844	0.1182-	0.0576	0.2736-	0.0934	**0.0261**	0.0150
20J6M2F	0.3313-	0.0898	0.2906-	0.0977	0.2524-	0.0838	0.1983-	0.0876	0.312-	0.0878	**0.0438**	0.0286
20J6M3F	0.2508-	0.1105	0.1242-	0.0758	0.1582-	0.0508	0.1465-	0.0663	0.224-	0.0625	**0.0347**	0.0287
40J4M2F	0.116-	0.0471	0.0669-	0.0295	0.0721-	0.0153	0.1117-	0.0336	0.1145-	0.0348	**0.0313**	0.0228
40J4M3F	0.3603-	0.1151	0.1519-	0.0632	0.2345-	0.0892	0.3067-	0.0682	0.4102-	0.0806	**0.0817**	0.0393
40J6M2F	0.3183-	0.0899	0.2599-	0.1181	0.1663=	0.0866	0.1732-	0.0653	0.4037-	0.1074	**0.0795**	0.0756
40J6M3F	0.2059-	0.0742	0.208-	0.0870	0.3151-	0.1253	0.2393-	0.0880	0.2094-	0.0570	**0.0374**	0.0183
60J4M2F	0.1487-	0.0587	0.0938-	0.0415	0.0823-	0.0188	0.1416-	0.0232	0.1605-	0.0388	**0.0285**	0.0202
60J4M3F	0.3768-	0.0918	0.1446-	0.0536	0.2224-	0.0576	0.3557-	0.1167	0.3573-	0.1205	**0.0669**	0.0354
60J6M2F	0.1572-	0.0568	0.1343-	0.0787	0.0933=	0.0286	0.1903-	0.0506	0.142-	0.0351	**0.0629**	0.0437
60J6M3F	0.327-	0.0771	0.1399-	0.0652	0.3396-	0.1326	0.3534-	0.0991	0.3182-	0.1029	**0.0464**	0.0330
80J4M2F	0.0956-	0.0327	0.0377-	0.0122	0.0482-	0.0218	0.1273-	0.0448	0.0765-	0.0133	**0.0211**	0.0146
80J4M3F	0.2275-	0.0497	0.0819-	0.0284	0.1491-	0.0531	0.3122-	0.0559	0.2044-	0.0415	**0.0404**	0.0248
80J6M2F	0.1295-	0.0214	0.061-	0.0249	0.0717-	0.0287	0.1835-	0.0318	0.1175-	0.0395	**0.0250**	0.0247
80J6M3F	0.2081-	0.0466	0.0887=	0.0348	0.2098-	0.0584	0.3055-	0.1647	0.1964-	0.0339	**0.0451**	0.0438
100J4M2F	0.145-	0.0342	0.0618-	0.0191	0.0916-	0.0217	0.2338-	0.0708	0.1335-	0.0267	**0.0191**	0.0116
100J4M3F	0.1596-	0.0500	0.0636-	0.0329	0.1072-	0.0379	0.2161-	0.0795	0.1375-	0.0351	**0.0210**	0.0127
100J6M2F	0.0671-	0.0352	0.0787-	0.0292	0.0732-	0.0210	0.1177-	0.0242	0.0684-	0.0228	**0.0291**	0.0160
100J6M3F	0.1695-	0.0316	0.0858-	0.0460	0.1227-	0.0594	0.2653-	0.0561	0.177-	0.0396	**0.0279**	0.0182
-/=/+	20/0/0		19/1/0		18/2/0		20/0/0		20/0/0			

Table 9: Statistical results of Spread (min) metrics of all comparison algorithms.

Instances	Spread											
	NSGA-II		MOEA/D		DABC		KTPO		VICA		KPMA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
20J4M2F	1.0164=	0.0945	1.0072=	0.0492	1.0578=	0.2790	1.0023=	0.1344	1.0151=	0.0408	**0.9241**	0.2050
20J4M3F	0.9143=	0.0550	1.0003=	0.1277	**0.9183=**	0.1208	0.9864=	0.1057	1.0993=	0.2000	0.9823	0.2423
20J6M2F	0.9853=	0.0359	1.0133=	0.0550	**0.9712=**	0.0562	1.0175=	0.1153	1.0389=	0.0960	1.1191	0.2771
20J6M3F	**1.0161=**	0.1036	1.0459=	0.1298	1.1268=	0.2287	1.0327=	0.0800	1.0582=	0.1426	1.0462	0.3097
40J4M2F	0.9973=	0.0212	1.0129=	0.0417	**0.9534=**	0.1236	0.9866=	0.1015	0.9622=	0.0548	1.0697	0.3488
40J4M3F	0.9977=	0.0206	0.9897=	0.0375	0.9913=	0.0515	**0.9755=**	0.0500	1.0012=	0.0035	1.1638	0.4589
40J6M2F	0.9710=	0.0308	0.9855=	0.0406	1.0465=	0.1462	0.9805=	0.0543	1.0008=	0.0164	**0.9685**	0.2518
40J6M3F	0.9989=	0.0118	1.0367=	0.0864	1.0232=	0.0522	1.0413=	0.0538	1.0606=	0.1439	**0.9516**	0.3094
60J4M2F	1.0008=	0.0228	1.0431=	0.0561	1.0056=	0.0729	0.9824=	0.0610	1.0042=	0.0364	**0.8809**	0.1850
60J4M3F	**0.9884=**	0.0262	1.0128=	0.0492	1.011=	0.0193	1.0143=	0.0518	0.999=	0.0163	0.9923	0.0429
60J6M2F	**0.9801=**	0.0363	1.0157=	0.0588	0.9874=	0.0541	1.0279=	0.0701	1.0272=	0.0643	1.0084	0.1547
60J6M3F	1.0125=	0.0372	0.9931=	0.0713	1.0106=	0.0284	1.0106=	0.0430	**0.993=**	0.0218	1.0249	0.1174
80J4M2F	0.9643=	0.0468	1.0332=	0.0796	0.9376=	0.1437	1.0011=	0.0148	0.9904-	0.0258	**0.9265**	0.0707
80J4M3F	0.9848=	0.0278	1.0315=	0.0530	0.9783=	0.0532	1.0035=	0.0160	**0.9791=**	0.0249	0.9918	0.1462
80J6M2F	0.9835=	0.0429	1.0488=	0.0330	0.9876=	0.0696	0.9791=	0.0529	0.967=	0.0600	**0.9660**	0.2051
80J6M3F	0.9954=	0.0276	1.0544=	0.0907	0.9831=	0.0332	0.9851=	0.0462	0.9877=	0.0147	**0.9812**	0.0738
100J4M2F	0.9887=	0.0218	1.0123=	0.0163	0.9954=	0.0540	1.0054=	0.0138	0.9961=	0.0222	**0.9133**	0.1697
100J4M3F	1.0265=	0.0534	1.0558=	0.0623	0.9875=	0.0439	1.0338=	0.0894	1.0049=	0.0235	**0.9694**	0.1297
100J6M2F	**0.9455=**	0.0815	1.039=	0.0569	1.0038=	0.0351	1.0053=	0.0558	0.9871=	0.0516	1.0439	0.1769
100J6M3F	1.0141-	0.0313	1.0564=	0.0592	0.9629=	0.0763	0.9939=	0.0170	0.9972=	0.0244	**0.9308**	0.0994
-/=/+	1/19/0		0/20/0		0/20/0		0/20/0		1/19/0			

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 population 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 significantly 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.

References

[1] L.-C. Wang, C.-C. Chen, J.-L. Liu, P.-C. Chu, Framework and deployment of a cloud-based advanced planning and scheduling system, Robotics and Computer-Integrated Manufacturing 70 (2021) 102088.

[2] S. Elsherbiny, E. Eldaydamony, M. Alrahmawy, A. E. Reyad, An extended intelligent water drops algorithm for workflow scheduling in cloud computing environment, Egyptian Informatics Journal 19 (2018) 33–55.

[3] R. Li, W. Gong, C. Lu, L. Wang, A learning-based memetic algorithm for energy-efficient flexible job shop scheduling with type-2 fuzzy processing time, IEEE Transactions on Evolutionary Computation (2022) 1–1.

[4] R. Li, W. Gong, C. Lu, Self-adaptive multi-objective evolutionary algorithm for flexible job shop scheduling with fuzzy processing time, Computers & Industrial Engineering 168 (2022) 108099.

[5] K. Huang, R. Li, W. Gong, R. Wang, H. Wei, Brce: bi-roles co-evolution for energy-efficient distributed heterogeneous permutation flow shop scheduling with flexible machine speed, Complex & Intelligent Systems (2023).

[6] E. Cevikcan, M. B. Durmusoglu, An integrated job release and scheduling approach on parallel machines: An application in electric wire-harness industry, Computers & Industrial Engineering 76 (2014) 318–332.

[7] G. Rivera, R. Porras, J. P. Sanchez-Solis, R. Florencia, V. GarcÃŋa, Outranking-based multi-objective pso for scheduling unrelated parallel machines with a freight industry-oriented application, Engineering Applications of Artificial Intelligence 108 (2022) 104556.

[8] J. Wang, G. Song, Z. Liang, E. Demeulemeester, X. Hu, J. Liu, Unrelated parallel machine scheduling with multiple time windows: An application to earth observation satellite scheduling, Computers & Operations Research 149 (2023) 106010.

[9] S. Hatami, R. Ruiz, C. AndrÃĺs-Romano, Heuristics for a distributed parallel machine assembly scheduling problem with eligibility constraints, in: 2015 International Conference on Industrial Engineering and Systems Management (IESM), 2015, pp. 145–153.

[10] C. Lu, L. Gao, J. Yi, X. Li, Energy-efficient scheduling of distributed flow shop with heterogeneous factories: A real-world case from automobile industry in china, IEEE Transactions on Industrial Informatics 17 (2021) 6687–6696.

[11] C. Lu, B. Zhang, L. Gao, J. Yi, J. Mou, A knowledge-based multiobjective memetic algorithm for green job shop scheduling with variable machining speeds, IEEE Systems Journal 16 (2022) 844 – 855.

[12] F. Zhao, R. Ma, L. Wang, A self-learning discrete jaya algorithm for multiobjective energy-efficient distributed no-idle flow-shop scheduling problem in heterogeneous factory system, IEEE Transactions on Cybernetics (2021) 1–12.

[13] F. Zhao, X. He, L. Wang, A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem, IEEE Transactions on Cybernetics 51 (2021) 5291–5303.

[14] R. Li, W. Gong, L. Wang, C. Lu, S. Jiang, Two-stage knowledge-driven evolutionary algorithm for distributed green flexible job shop scheduling with type-2 fuzzy processing time, Swarm and Evolutionary Computation (2022) 101139.

[15] D. Lei, M. Liu, An artificial bee colony with division for distributed unrelated parallel machine scheduling with preventive maintenance, Computers & Industrial Engineering 141 (2020) 106320.

[16] Z. Pan, D. Lei, L. Wang, A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling, IEEE Transactions on Cybernetics 52 (2022) 5051 – 5063.

[17] X. Chen, Y.-S. Ong, M.-H. Lim, K. C. Tan, A multi-facet survey on memetic computation, IEEE Transactions on Evolutionary Computation 15 (2011) 591–607.

[18] S. Selvi, D. Manimegalai, Multiobjective variable neighborhood search algorithm for scheduling independent jobs on computational grid, Egyptian Informatics Journal 16 (2015) 199–212.

[19] Y. Su, Y. Li, S. Xuan, Prediction of complex public opinion evolution based on improved multi-objective grey wolf optimizer, Egyptian Informatics Journal 24 (2023) 149–160.

[20] M. Kalra, S. Singh, A review of metaheuristic scheduling techniques in cloud computing, Egyptian Informatics Journal 16 (2015) 275–295.

[21] G. Zhang, B. Liu, L. Wang, D. Yu, K. Xing, Distributed co-evolutionary memetic algorithm for distributed hybrid differentiation flowshop scheduling problem, IEEE Transactions on Evolutionary Computation 26 (2022) 1043–1057.

[22] J.-J. Wang, L. Wang, A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling, IEEE Transactions on Evolutionary Computation 26 (2022) 461–475.

[23] C. Lu, Y. Huang, L. Meng, L. Gao, B. Zhang, J. Zhou, A pareto-based collaborative multi-objective optimization algorithm for energy-efficient scheduling of distributed permutation flow-shop with limited buffers, Robotics and Computer-Integrated Manufacturing 74 (2022) 102277.

[24] Q. Zhang, L. Hui, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (2007) 712–731.

[25] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Transactions on Evolutionary Computation 6 (2002) 182–197.

[26] Z. Pei, M. Wan, Z. Z. Jiang, Z. Wang, X. Dai, An approximation algorithm for unrelated parallel machine scheduling under tou electricity tariffs, IEEE Transactions on Automation Science and Engineering 18 (2021) 743–756.

[27] J. Cheng, F. Chu, M. Zhou, An improved model for parallel machine scheduling under time-of-use electricity price, IEEE Transactions on Automation Science and Engineering 15 (2018) 896–899.

[28] W. Fang, H. Zhu, Y. Mei, Hybrid meta-heuristics for the unrelated parallel machine scheduling problem with setup times, Knowledge-Based Systems 241 (2022) 108193.

[29] X. L. Zheng, L. Wang, A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem, IEEE Transactions on Systems, Man, and Cybernetics: Systems 48 (2018) 790–800.

[30] J. Ding, L. Shen, Z. LÃij, L. Xu, U. Benlic, A hybrid memetic algorithm for the parallel machine scheduling problem with job deteriorating effects, IEEE Transactions on Emerging Topics in Computational Intelligence 4 (2020) 385–397.

[31] M. Z. Wang, L. L. Zhang, T. M. Choi, Bi-objective optimal scheduling with raw materialâĂŹs shelf-life constraints in unrelated parallel machines production, IEEE Transactions on Systems, Man, and Cybernetics: Systems 50 (2020) 4598–4610.

[32] D. Mecler, V. Abu-Marrul, R. Martinelli, A. Hoff, Iterated greedy algorithms for a complex parallel machine scheduling problem, European Journal of Operational Research 300 (2022) 545–560.

[33] H. Wang, B. Alidaee, Unrelated parallel machine selection and job scheduling with the objective of minimizing total workload and machine fixed costs, IEEE Transactions on Automation Science and Engineering 15 (2018) 1955–1963.

[34] Z. Cao, C. Lin, M. Zhou, C. Zhou, K. Sedraoui, Two-stage genetic algorithm for scheduling stochastic unrelated parallel machines in a just-in-time manufacturing context, IEEE Transactions on Automation Science and Engineering (2022) 1–14.

[35] H. Chen, P. Guo, J. Jimenez, Z. S. Dong, W. Cheng, Unrelated parallel machine photolithography scheduling problem with dual resource constraints, IEEE Transactions on Semiconductor Manufacturing 36 (2023) 100–112.

[36] S. Wang, R. Wu, F. Chu, J. Yu, Unrelated parallel machine scheduling problem with special controllable processing times and setups, Computers & Operations Research 148 (2022) 105990.

[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] T. Zhou, Q. Zhang, X. Wang, X. Ren, Imperialist competitive algorithm based on vnsobl optimization for distributed parallel machine scheduling problem, in: 2019 Chinese Automation Congress (CAC), 2019, pp. 5717–5723.

[39] L. MÃűnch, L. Shen, Parallel machine scheduling with the total weighted delivery time performance measure in distributed manufacturing, Computers & Operations Research 127 (2021) 105126.

[40] G. Zhang, B. Liu, L. Wang, K. Xing, Distributed heterogeneous co-evolutionary algorithm for scheduling a multistage fine-manufacturing system with setup constraints, IEEE Transactions on Cybernetics (2022) 1–14.

[41] C. Lei, N. Zhao, S. Ye, X. Wu, Memetic algorithm for solving flexible flow-shop scheduling problems with dynamic transport waiting times, Computers & Industrial Engineering 139 (2020) 105984.

[42] M. Abedi, R. Chiong, N. Noman, R. Zhang, A multi-population, multi-objective memetic algorithm for energy-efficient job-shop scheduling with deteriorating machines, Expert Systems with Applications 157 (2020) 113348.

[43] M. Kurdi, A memetic algorithm with novel semi-constructive evolution operators for permutation flowshop scheduling problem, Applied Soft Computing 94 (2020) 106458.

[44] W. Shao, Z. Shao, D. Pi, A network memetic algorithm for energy and labor-aware distributed heterogeneous hybrid flow shop scheduling problem, Swarm and Evolutionary Computation 75 (2022) 101190.

[45] R. Li, W. Gong, C. Lu, A reinforcement learning based rmoea/d for bi-objective fuzzy flexible job shop scheduling, Expert Systems with Applications 203 (2022) 117380.

[46] Y.-N. Wang, L.-H. Wu, X.-F. Yuan, Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure, Soft Computing 14 (2010) 193–209.

[47] L. While, P. Hingston, L. Barone, S. Huband, A faster algorithm for calculating hypervolume, IEEE Transactions on Evolutionary Computation 10 (2006) 29–38.

[48] R. C. Van Nostrand, Design of experiments using the taguchi approach: 16 steps to product and process improvement, Technometrics 44 (2002) 289–289.