A Hybrid Evolutionary Hyper-Heuristic Approach for Intercell Scheduling Considering Transportation Capacity

Dongni Li, Rongxin Zhan, Dan Zheng, Miao Li, and Ikou Kaku

***Abstract*—The problem of intercell scheduling considering transportation capacity with the objective of minimizing total weighted tardiness is addressed in this paper, which in nature is the coordination of production and transportation. Since it is a practical decision-making problem with high complexity and large problem instances, a hybrid evolutionary hyper-heuristic (HEH) approach, which combines heuristic generation and heuristic selection, is developed in this paper. In order to increase the diversity and effectiveness of heuristic rules, genetic programming is used to automatically generate new rules based on the attributes of parts, machines, and vehicles. The new rules are added to the candidate rule set, and a rule selection genetic algorithm is developed to choose appropriate rules for machines and vehicles. Finally, scheduling solutions are obtained using the selected rules. A comparative evaluation is conducted, with some state-of-the-art hyper-heuristic approaches which lack some of the strategies proposed in HEH, with a meta-heuristic approach that is suitable for large scale scheduling problems, and with adaptations of some well-known heuristic rules. Computational results show that the new rules generated in HEH have similarities to the best-performing human-made rules, but are more effective due to the evolutionary processes in HEH. Moreover, the HEH approach has advantages over other approaches in both computational efficiency and solution quality, and is especially suitable for problems with large instance sizes.**

***Note to Practitioners*—Our survey of the equipment manufacturing industry in China indicates that, for complex products like synthetic transmission devices, intercell transfers occur in the processing routes of more than 51% of parts. More than 47% of tardy parts are caused by inefficient intercell cooperation. Therefore, intercell transfers are inevitable and it is worth an effort to find out an effective approach to intercell scheduling. To solve intercell scheduling problems, two characteristics in industrial environments of complex products cannot be neglected. The first one is the large problem sizes, which involve up to hundreds of parts and thousands of operations; and the second one is the importance of transportation to intercell scheduling, which involves allocation and utilization of vehicles. However, sufficient transportation capacity is taken as a common assumption in most of research with respect to intercell scheduling, which shields the transportation dimension and hinders the application of these intercell scheduling approaches. Therefore, intercell scheduling with limited transportation capacity is considered, and a hybrid evolutionary hyper-heuristic is proposed in this paper. The advantages of this approach lie in that, (i) as a hyper-heuristic, it provides high computational efficiency, which is suitable for industrial environments with large problem sizes; and (ii) genetic programming is employed to generate problem-specific heuristic rules, which enhances the learning and searching ability of the approach. We compare the proposed approach with the man-made heuristic rules that are widely used in practice. Experimental results indicate that, for hundreds of parts and thousands of operations, given the same running time, our approach outperforms man-made rules with an average gap of 60.6% in minimizing total weighted tardiness. Therefore, our approach is advantageous in both computational efficiency and solution quality, and is especially suitable for the intercell scheduling problems in practice.**

***Index Terms*—Discrete event systems, manufacturing automation, scheduling, transportation.**

I. Introduction

CELLULAR manufacturing (CM) is an implementation of group technology (GT) in which parts or part families requiring similar production processes are grouped into distinct manufacturing cells. Ideally, all machines required for producing a part can be allocated to a cell.

However, because the parts for modern products are becoming more complicated, parts assigned to one cell may require a particular machine in another cell. Although this difficulty can be resolved by purchasing additional machines, this may not be economical. The parts requiring processing on machines in two or more cells are termed exceptional parts. Additionally, there exist special machines that have to be grouped in a specific cell because they are large and expensive and, hence, are usually required by various parts from different cells. These machines are called exceptional machines. Our survey for the equipment manufacturing industry in China indicates that, for complicated assemblies, such as synthetic transmission devices, intercell transfers occur in the processing routes of more than 51% of parts, and more than 47% of tardy parts are caused by inefficient intercell cooperation. Take Inner Mongolia First Machinery Group Corporation as an example, its direct economic losses in 2011 caused by the tardiness of exceptional parts were up to 1.3 billion RMB and the corresponding inventory costs were approaching 50 million RMB

Manuscript received January 07, 2015; revised April 25, 2015; accepted July 28, 2015. Date of publication September 16, 2015; date of current version April 05, 2016. This paper was recommended for publication by Associate Editor J. Sethuraman and Editor M. P. Fanti upon evaluation of the reviewers' comments. This work was supported in part by the Natural Science Foundation of China under Grant 71401014 and in part by the Natural Science Foundation of Beijing under Grant 4122069.

D. Li, R. Zhan, D. Zheng, and M. Li are with the Beijing Key Lab of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, China (e-mail: ldn@bit.edu.cn; zhan_rongxin@163.com; zhengdan04@163.com; shininglm@gmail.com).

I. Kaku is with the Department of Environmental and Information Studies, Tokyo City University, Yokohama, Japan (e-mail: kakuikou@tcu.ac.jp).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2015.2470080

[1]. Therefore, there is a need to study the scheduling problem in the context of multiple cells with intercell transfers.

It was reported that 72.9% of manufacturing firms with more than 100 employees have adopted manufacturing cells [2], the median level of intercell flow was 10% (with a mean value of approximately 20%) and only 10% of the surveyed shops processed parts completely within cells [3]. It was stated in [3] that because intercell transfers are inevitable, an ideal cellular manufacturing system (CMS) is difficult to implement, and the impact of intercell transfers on manufacturing systems should be quantitatively analyzed. However, there has been relatively little research into the scheduling problem considering intercell transfers.

Solimanpur, *et al.* [4] developed a two-stage heuristic approach in which the sequence of parts within a cell was determined in the first stage and sequence of cells in the second. The performance of the proposed approach was compared with LN-PT [5], which is the combination of heuristic rules LN [6] and PT [7]. They assumed that intercell transfers cannot happen until the destination cells have completed processing all parts originally assigned to them. However, intercell transfers may occur at anytime in practice. Tavakkoli-Moghaddam, *et al.* [8] proposed a multicriteria scatter search algorithm by simultaneously considering makespan, intracellular movement, tardiness, and sequence dependent setup costs. The problem considered consisted of two distinct sub-problems, i.e., intracell scheduling which determined the sequence of parts within cells, and intercell scheduling which determined the sequence of cells. However, due to the tight coupling among cells caused by intercell transfers, most intercell scheduling problems are too complex to be decomposed into distinct sub-problems. Golmohammadi, *et al.* [9] developed an improved electromagnetism-like (EM-like) approach to determine the sequence of parts within each part family and the sequence of part families in an integrated manner. However, they adopted the same assumptions as Solimanpur, *et al.* [4] that the intercell transfers cannot happen until the destination cells have completed processing all parts originally assigned to them. Tang, *et al.* [10] proposed a nonlinear programming model and developed a meta-heuristic based on scatter search to solve the problem. Based on the model presented by Tang, *et al.* and Elmi, *et al.* [11] proposed an integer linear programming model and developed a simulated annealing-based approach to solve the problem in which parts were allowed to visit machines more than once in a nonconsecutive manner. Considering a CMS consisting of multiple unit processing machines and one batch processing machine, Li, *et al.* [12] developed a combinatorial ant colony optimization approach for the intercell scheduling problem.

Because cells are usually located separately, the impact of intercell transfers should not be neglected. Some of the above studies [8], [10], [12] took the distance or time of intercell transfers into account, but there is an implicit assumption in these studies that the intercell transportation capacity is sufficient for parts to be delivered to the next cell (if any) as soon as they complete the operations in the current cell. Nevertheless, in practice, each cell has a limited number of vehicles for intercell transportation, and parts have to wait if the vehicles are unavailable. Furthermore, due to the possibility of overlapping among cells, a part may have alternative intercell processing routes, so a vehicle also needs to determine the destination cells for the parts. Therefore, intercell scheduling is essentially the coordination of parts scheduling for machines and batch formation for vehicles (in conjunction with flexible intercell routing).

On the other hand, as multiple cells are involved in intercell scheduling, the problem size is greatly increased. Therefore, both computational efficiency and solution quality are required for intercell scheduling approaches.

In industrial environments with large problem instances, heuristic rules are often preferable due to the simplicity and efficiency [4]. However, no single rule can outperform others in all circumstances, which means the performance of heuristic rules depends heavily on the scheduling environment and objectives. Therefore, some studies adopted combinatorial heuristic rules [13] in which the appropriate combinatorial rules are obtained by enumeration. However, for intercell scheduling problems, the enumeration strategy leads to unaffordable computational costs due to the enormous size of the search space. Therefore, there is a need for more effective strategies searching for appropriate combinatorial rules.

In this regard, meta-heuristics are used to search for suitable combinations of rules, which are then applied to generate scheduling solutions. These approaches are called hyper-heuristics (HH) [14].

Hyper-heuristics are usually classified into two main categories, heuristic selection (heuristics to choose heuristics) and heuristic generation (heuristics to generate heuristics) [14]. There are many applications of hyper-heuristics in the combinatorial optimization problems such as time-tabling, cutting and packing, production scheduling, constraint satisfaction, and vehicle routing. Several hyper heuristics have been proposed for production scheduling problems.

With respect to heuristic selection, the method in [15] integrated problem-specific heuristic rules with local search, in which the scheduling decisions were first divided into different time windows, and then a heuristic rule was selected for every time window and the scheduling solutions generated by applying the selected rule to the problem. Aytug, *et al.* [16] reviewed several types of machine learning for scheduling problems, which automated the acquisition of knowledge from experts. Herrmann, *et al.* [17] studied a job shop scheduling problem derived from the semiconductor industry. To meet the requirements of both due dates and throughput, a hyper-heuristic based on genetic algorithm (GA) was used to search for the appropriate combinations of rules and the scheduling solutions were then generated according to the selected rules. Vázquez-Rodríguez, *et al.* [18] studied a scheduling problem of an HFS and GA was adopted to search for suitable rules. Geiger, *et al.* [19] used a similar method for scheduling a single machine. Vázquez-Rodríguez and Petrovic [20] developed a hyper-heuristic, where a standard GA was employed to search the sequences of heuristic rules to solve a multimachine cardboard box shop scheduling problem. Based on the model presented by Vázquez-Rodríguez and Petrovic [20], Ochoa, *et al.* [21] did a further study and used the notion of fitness landscapes to analyze the search space of the heuristic rules.

However, heuristic selection is a method of selecting from a set of existing heuristic rules [14], and the candidate heuristic rules are set in advance according to the researchers' domain knowledge. It is inefficient for a human analyst to specialize heuristics on a per-instance basis [14], and therefore, the heuristic selection may limit the diversity and effectiveness of the candidate rules due to the lack of exploration of more appropriate rules.

With respect to heuristic generation, Geiger and Uzsoy [22] applied a genetic programming (GP) hyper-heuristic to generating heuristic rules for a batch processing machine. Nguyen, *et al.* [23] investigated the use of GP for automatically discovering new dispatching rules for the job shop scheduling problem. These studies indicate that heuristic components can be combined and rearranged by heuristic generation to create heuristics superior to those created by humans.

However, heuristic generation is challenged by the size of the search space, which grows exponentially both in the number of components that define a rule and the number of rules to be created [14]. Recently, some studies started to consider the combination of heuristic generation and heuristic selection [24].

Motivated by the successful applications of hyper-heuristics, a hybrid evolutionary hyper-heuristic (HEH) approach is presented in this paper. In order to increase the diversity and effectiveness of the rules, GP is adopted to generate heuristic rules to extend the candidate rule set, based on which a rule selection genetic algorithm (RSGA) is developed to search appropriate rules for machines and vehicles. Finally, the obtained combinatorial rules are used to generate scheduling solutions. Some advanced techniques are included in HEH compared to the two-stage hyper-heuristic proposed in [24], such as a problem-specific encoding scheme for each machine and vehicle, a catastrophe operator to avoid falling into local optima, and a local search procedure to balance computational efficiency and solution quality. The analysis of the similarities between the new rules generated in HEH and those existing ones with the best performance is conducted, and the effectiveness of HEH is demonstrated by extensive comparisons with the combinatorial heuristic rules, with some state-of-the-art hyper-heuristics, and with tabu search (TS) that is suitable for large scale scheduling problems.

The rest of this paper is organized as follows: Section II presents the description of the addressed intercell scheduling problem. In Section III, the HEH approach is developed in detail. A series of experimental tests and results are reported in Section IV. Section V provides the concluding remarks.

II. Problem Description

A CMS consists of a number of job shop cells. A part family or families and the associated machine groups that constitute each cell have been identified. Machines have been laid out within each cell.

The intercell scheduling problem addressed in this paper is based on the following assumptions.

- The processing route of a part consists of multiple operations that have precedence constraints and require processing on machines in two or more cells.
- Due to the overlapping capabilities of machines, parts have alternative processing routes. However, for a given operation, there is at most one machine capable of processing it within each cell.
- The setup time is taken into account, and is known and independent of the sequence of operations.
- The intracell transfer time is neglected, while the intercell transfer time is taken into account.
- Each cell has one and only one vehicle, which is responsible for transporting parts to other cells, at least one part at a time.
- All of the vehicles are identical and have the same capacity constraint.
- The volumes for all of the parts are the same, so that the size of a batch on a vehicle is only determined by the number of parts in that batch. Parts cannot be added to, or removed from, a batch once transportation of the batch has started.
- Each batch has one and only one destination cell. After arriving at the destination cell and unloading the parts, the vehicle returns to its original cell without delay. The unloading time is neglected, and the time for the vehicle to return the original cell is the same as its travel time to the destination cell.
- A machine processes only one part at a time.
- Other conditions, such as preemption, machine breakdown, limited buffers, and the absence of operators, are not considered in this paper.

The objective of the addressed problem is to minimize total weighted tardiness (TWT), which is given in (1)

$$\min \sum_{i=1}^{N} w_i \max\{f_i - d_i, 0\} \tag{1}$$

where i represents the index of a part; N the total number of the parts; w_i the weight of part p_i; f_i the completion time of p_i; and d_i the due date of p_i.

For illustration, an instance is given in Table I. Assume that there are three cells, nine machines and three parts in a CMS. The part-machine load matrix is shown in Table I. As indicated in Table I, both Part 1 and Part 2 need to be transported from C_1 to C_2, and therefore, batch formation for the vehicle from C_1 to C_2 should be considered. Moreover, because either M_2, which is located in C_1, or M6, which is located in C_2, can process O_{33}, two alternative intercell routes for Part 3 are generated.

III. Proposed HEH Approach

The integration of part scheduling for machines and batch formation for vehicles greatly increases the size of the search space and made the online solution very difficult in practice. In the proposed HEH approach, GP is used to generate new heuristic rules offline and then GA is employed to select heuristic rules from the extended set of the candidate rules. Finally, the selected combinatorial rules are used to construct the scheduling solutions.

A. Framework of HEH

For the addressed intercell scheduling problem, in addition to the assignment sub-problem and the sequencing sub-problem

TABLE I
OPERATION-MACHINE LOAD MATRIX

Part	Operation	C1			C2			C3		
		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9
Part 1	O_{11}	-	4	-	-	-	-	-	-	-
	O_{12}	-	-	-	10	-	-	-	-	-
	O_{13}	-	-	-	-	4	-	-	-	-
Part 2	O_{21}	9	-	-	-	-	-	-	-	-
	O_{22}	-	-	-	-	3	-	-	-	-
	O_{23}	-	-	-	-	-	-	-	-	6
Part 3	O_{31}	-	-	5	-	-	-	-	-	-
	O_{32}	-	-	-	-	-	-	-	9	-
	O_{33}	-	2	-	-	-	3	-	-	-

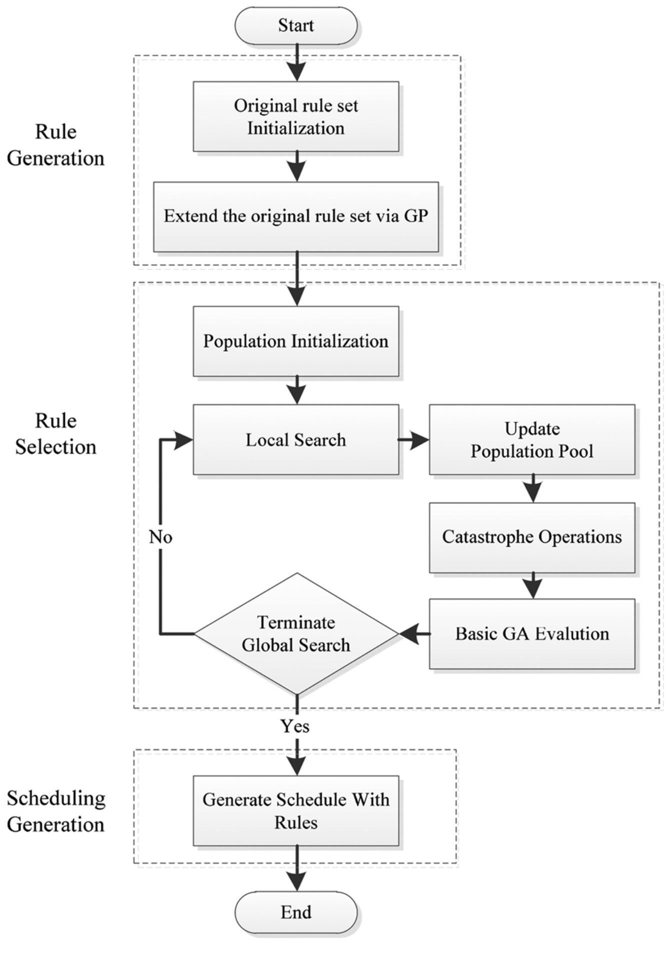


Fig. 1. Framework of the HEH approach.

caused by machine scheduling, the consideration of transportation capacity leads to the third sub-problem, i.e., batch formation. According to the assumptions, although parts have alternative processing routes, for a given operation, there is at most one machine capable of processing it within each cell, indicating the assignment sub-problem can be resolved in conjunction with the batch formation sub-problem. Therefore, the goal of HEH is to generate and search appropriate heuristic rules for the sequencing sub-problem and the batch formation sub-problem, and then apply the rules to obtaining the scheduling solutions.

The framework of the HEH approach is shown in Fig. 1. In the first stage, the original set of the candidate rules is initialized, and then extended via GP. In the second stage, using RSGA which differs from the basic GA through the presence of the catastrophe operator and the local search procedure, heuristic rules with good performance are selected from the extended rule set. Finally, in the last stage, the scheduling solutions are generated by the heuristic rules obtained in the second stage.

B. Rule Generation

In the first stage, heuristic rules are generated via GP to extend the original set of candidate rules.

1) Representation of Heuristic Rules: The heuristic rules used in the HEH approach incorporate different characteristics of machines, parts and vehicles. Because GP is appropriate for creating representations composed of attributes and operators using tree structures, it is adopted in the HEH approach for the generation of the combinatorial rules.

The terminals and functions for the GP hyper-heuristic contain predominantly the attributes that make up the rules with good performance [24]. The function set contains basic arithmetic and mathematical operations, including addition $(+)$, subtraction $(-)$, division $(/)$, multiplication $(*)$, and maximization (Max).

The terminals for the sequencing sub-problem are presented as follows:

r_i	release time of part i
d_i	due date of part i
cpt_i	processing time for the next operation of part i on the current machine
at_i	arrival time of part $i$at the current machine
op_i	the number of the operations yet to be scheduled for part i
rpt_i	the processing time for the operations yet to be scheduled for part i
w_i	weight of part i
q	a constant generated from the standard uniform distribution on (0,1]

The terminals for the batch formation sub-problem are presented as follows:

r_i	release time of part i
d_i	due date of part i
cpt_i	processing time for the next operation of part i on the current machine
ptt_i	intercell transfer time to the next cell for part i
q	a constant generated from the standard uniform distribution on (0,1]

2) Parameters for GP: Table II summarizes the parameter settings used for GP, which is based on ECJ20 library [25].

A ramped half-and-half method is adopted to initialize the population, and the crossover operation exchanges sub-trees be-

TABLE II
PARAMETERS FOR GP

Parameter	Description
Population size	1024
Crossover probability	90%
Mutation probability	10%
Generations	50
Selection	Tournament (size 17)
Initial population	Ramped half-and-half (min depth 2, max depth 6)
Max depth for crossover	17
Fitness function	*fit*(Δ)(shown in Section III-B3)

tween the two parents to create new offspring. Sub-tree mutation is adopted in this paper, which randomly selects a mutation point in a tree and substitutes the sub-trees rooted there with randomly generated sub-trees. Examples of the crossover and mutation operations are shown in Fig. 2.

3) Fitness Evaluation: In general, objective function values are adopted to evaluate the fitness of a chromosome. However, it is difficult to use them directly, as the gap between different objective function values might be very large. Therefore, the deviation of the objective function value, which is calculated by (2), is used to evaluate the performance of the corresponding GP rule

$$dev(\mu, Ins) = \frac{obj(\mu, Ins) - Ref(Ins)}{Ref(Ins)} \tag{2}$$

where μ represents a GP generated rule; Ins an instance of the addressed problem; $obj(\mu, Ins)$ the objective function value obtained by μ through solving Ins; and $Ref(Ins)$ the reference value obtained via a basic GA by solving Ins.

Because no methods, such as local search and the catastrophe operator, are adopted in the basic GA for the improvement of its performance, its objective function value, $Ref(Ins)$, is larger than that of the proposed method, $obj(\mu, Ins)$.

Based on (2), the fitness of μ can be evaluated by (3)

$$fit(\mu) = dev_{avg}(\mu) = \frac{\sum_{i=1}^{I} dev(\mu, Ins_i)}{I} \tag{3}$$

where I represents number of instances.

4) Algorithm Description: Based on the analysis above, the algorithm for generating heuristic rules is presented as follows:

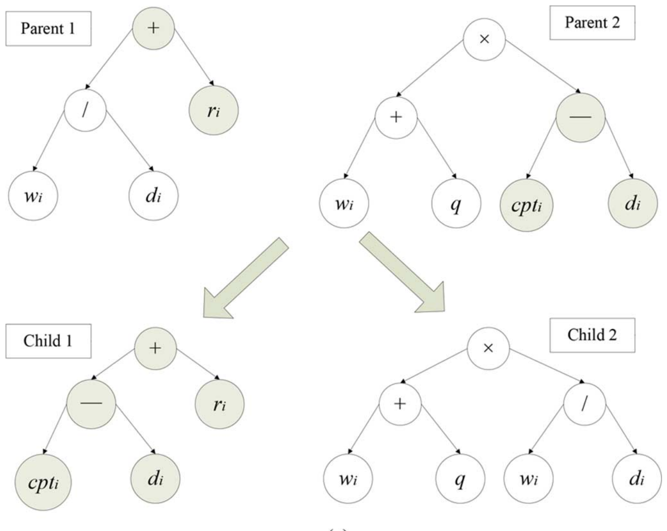
Heuristic Generation Algorithm (HGA)

Step 1. Initialization. Randomly generate a set of heuristic rules, which are represented by attributes and operators;

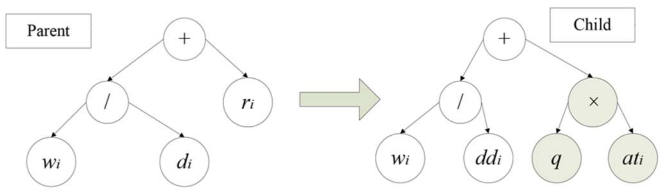
Step 2. Genetic selection. In each generation, calculate the fitness of each rule using (3) and select the good ones;

Step 3. Cross over and mutation. Generate a new rule by crossing over two parent rules and randomly mutate a rule;

Step 4. If there is no better rule for a given number of iterations, end; otherwise, go to Step 2.



(a)



(b)

Fig. 2. Examples of crossover and mutation on sub-trees for GP. (a) Crossover. (b) Mutation.

C. Rule Selection

In the second stage, appropriate combinations of heuristic rules are selected via RSGA from the candidate rule set that has been extended in the first stage. A rule is selected for each machine or vehicle, which remains the same for the entire planning horizon upon its assignment.

1) Encoding Scheme: An appropriate encoding scheme for the chromosome is essential for GA to represent feasible solutions. According to the addressed problem, the encoding scheme is designed for machines and vehicles, in which a chromosome consists of two segments, representing the sequencing rules and the batch formation rules, respectively.

Each gene in the chromosome, denoted by an index, represents the corresponding rule in the candidate rule set, as shown in Tables III and IV.

Among the sequencing rules listed in Table III, the first five rules are not due-date related, the next eight are due-date related, and the last three are the best ones generated by GP. Among the batch formation rules listed in Table IV, the first two rules are not due date related, the next two are due date related, and the last three are the best ones generated by GP.

For convenience of illustration, a possible chromosome of a simple instance including three cells, six machines and two parts is given in Fig. 3. The first position in the machine segment shows that the sequencing rule adopted by the first machine is SPT, which is the first one in the candidate sequencing rule set, as shown in Table III. Similarly, it is indicated in the batch formation segment that EDD is adopted by the first vehicle as the batch formation rule, which is the third one in the set of the candidate batch formation rules, as shown in Table IV.

TABLE III
CANDIDATE RULES FOR SEQUENCING

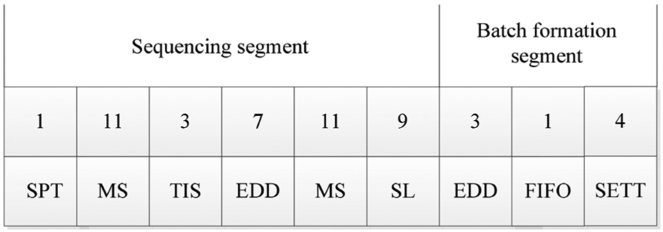
Index	Rules	Description	Formula
1	SPT	Select the part with the shortest processing time for its next operation.	$\min_i p_{ijm}$
2	WSPT	Select the part with the largest weighted processing time.	$\max_i w_i / p_{ijm}$
3	TIS	Select the part that has stayed in the system for the longest time.	$\max_i (t - at_i)$
4	SRPT	Select the part with the smallest processing time for the operations yet to be scheduled.	$\min_i rpt_i$
5	PT_TIS	Select the part with the smallest value of $p_{ijm} /(at_i - r_i)$.	$\min_i p_{ijm} /(at_i - r_i)$
6	AT_RPT	Select the part with the smallest value of $(at_i - r_i + d_i)$.	$\min_i (at_i - r_i + d_i)$
7	EDD	Select the part with the earliest due date.	$\min_i d_i$
8	WEDD	Select the part with the smallest weighted due date.	$\min_i d_i / w_i$
9	SL	Select the part with the smallest value of slack time.	$\min_i (d_i - r_i - rpt_i)$
10	ATC	Select the part with the smallest value of $\frac{w_i}{p_{ijm}} exp(-\frac{max(d_i - p_{ijm} - t,0)}{K\bar{p}})$.	$\min_i (\frac{w_i}{p_{ijm}} exp(-\frac{max(d_i - p_{ijm} - t,0)}{K\bar{p}}))$
11	MS	Select the part with the smallest value of $max(d_i - p_{ijm} - t,0)$.	$\min_i (max(d_i - p_{ijm} - t,0))$
12	COVERT	Select the part with the smallest value of $\frac{w_i}{p_{ijm}} max(1 - max(d_i - p_{ijm} - t,0),0)$.	$\min_i \frac{w_i}{p_{ijm}} max(1 - max(d_i - p_{ijm} - t,0),0)$
13	S_RPT	Select the part with the smallest value of $max(\frac{d_i - p_{ijm} - t}{p_{ijm}},0)$.	$\min_i (max(\frac{d_i - p_{ijm} - t}{p_{ijm}},0))$
14~16	SGP1 SGP2 SGP3	Select the part using the best three sequencing rules generated by GP, denoted by SGP1, SGP2, and SGP3, respectively.	

TABLE IV
CANDIDATE RULES FOR BATCH FORMATION

Index	Rules	Description
1	FIFO	Form a batch on the vehicle under the capacity constraint, with parts selected in ascending order of release times.
2	SPATT	Form a batch on the vehicle under the capacity constraint, with parts selected in ascending order of the sum of intercell transfer times and processing times.
3	EDD	Form a batch on the vehicle under the capacity constraint, with parts selected in ascending order of due dates.
4	SETT	Form a batch on the vehicle under the capacity constraint, with parts selected in ascending order of the sum of intercell transfer times and due dates.
5~7	TGP1 TGP2 TGP3	Transport the part using the best three batch formation rules generated by GP, denoted by TGP1, TGP2, and TGP3, respectively.

2) *Genetic Operators:* In RSGA, a roulette wheel method is adopted for the selection operator, and the probability of selecting chromosome g, denoted by $prob(g)$, is defined in (4)

$$prob(g) = \frac{fit(g)}{\sum_{g\in G} fit(g)} \tag{4}$$



Sequencing segment						Batch formation segment		
1	11	3	7	11	9	3	1	4
SPT	MS	TIS	EDD	MS	SL	EDD	FIFO	SETT

Fig. 3. Encoding scheme for a chromosome.

where G represents the number of chromosomes in the population; and $fit(g)$, defined in (5), represents the fitness of the chromosome

$$fit(g) = \frac{1}{obj(g) + 1} \tag{5}$$

where $obj(g)$ represents the objective function value of chromosome g.

In RSGA, two-point crossover operator and one-point mutation operator are adopted. Two segments of the chromosome perform these genetic operators respectively with the same crossover probability and mutation probability.

3) *Catastrophe Operator:* A basic GA easily falls into a local optimum, especially in the later period of the evolutionary

process. It is observed that if the best individual of the previous generation falls into a local optimum and remains unchanged for successive generations, a catastrophe operator is needed as a significant disturbance to the populations to start a new search. Therefore, the catastrophe operator is introduced to RSGA by increasing the mutation probability. The updated mutation probability, denoted by pm', is formulated by (6)

$$pm' = pm + (1 - pm) \times \tau \tag{6}$$

where pm is the original probability of mutation, τ is the catastrophe updating factor which determines the level of disturbance to the population and its value is restricted within the range $[\tau_{\min}, \tau_{\max}]$. The probability of GA falling into a local optimum increases as the evolutionary process goes on, and therefore, the value of τ should increase accordingly. In this regard, the evolutionary process is divided into three periods according to the ratio between the current generation, denoted by *it*, and the maximum generation, denoted by *gm*. The value of τ for each period is set as follows:

1) If $it/gm \in [k_1, k_2)$, where k_2 represents the terminal point of the middle period of the evolutionary process, then the value of τ is slightly increased and set to the medium value τ_{middle}, which is calculated using (7)

$$\tau_{\text{middle}} = \frac{\tau_{\min} + \tau_{\max}}{2}. \tag{7}$$

2) Otherwise, if $it/gm \in [k_2, 1)$, which indicates the late period of the evolutionary process, then the value of τ should be greatly increased according to (8)

$$\tau = \tau_{\text{middle}} + \frac{k_2(\tau_{\min} + \tau_{\max})}{2}. \tag{8}$$

4) Local Search: A GA has the ability to perform parallel search because it searches for a population of individuals rather than a single individual [18], while a local search procedure works on each single point of the search space, guiding the iteration from one point to another by locally improving the solution. Therefore, a hybrid approach combining GA with a local search procedure is more likely to provide better solution quality.

Moreover, because intercell scheduling approaches pay much attention to the computational efficiency in addition to the solution quality, the local search procedure should be simple but effective.

Based on the above considerations, a hill climbing (HC) algorithm is developed in RSGA due to its concise structure, which is presented as follows:

Hill Climbing (HC)

```
Step 1. Initialize the position index $pos = 0$ and the current chromosome $cur = g$, where $g$ is the initial chromosome;
Step 2. Let $next = NEIGHBOR(cur, pos)$, where next is the new chromosome generated by the neighborhood search algorithm, denoted $NEIGHBOR(cur, pos)$, for cur with a given position index pos;
Step 3. If cur performs better than next, end; otherwise, set $pos = pos + 1$, go to Step 2.
```

In the HC algorithm, the steps of $NEIGHBOR(cur, pos)$ are given as follows:

Neighborhood Search Algorithm (NSA)

```
Step 1. If position pos is in the sequencing segment, go to Step 2; otherwise, it is in the batch formation segment, go to Step 3;
Step 2. If position pos is not the end of sequencing segment, generate a new chromosome by swapping the values on positions pos and $pos + 1$ of cur; otherwise, generate a new chromosome by swapping the values on position pos and the beginning position in the sequencing segment of cur. End;
Step 3. If position pos is not the end of the batch formation segment, generate a new chromosome by swapping the values in positions pos and $pos + 1$ of cur; otherwise, generate a new chromosome by swapping the values in position pos and the beginning position in the batch formation segment of cur. End.
```

D. Schedule Generation

In the third stage, schedules are generated by decoding the chromosomes. The heuristic rules obtained from the second stage are used for part sequencing and batch formation. The decoding algorithm, part sequencing algorithm and batch formation algorithm are presented in this sub-section. The decoding algorithm is presented as follows, which shows how to obtain the scheduling solutions by calculating the values of the objective function.

Decoding Algorithm (DA)

```
Step 1. Simulation clock $t = 0$;
Step 2. For a given chromosome $g$, assign the sequencing rules to machines and the batch formation rules to vehicles;
Step 3. If all of the operations are completed, go to Step 8;
Step 4. If a machine, denoted by $m$, becomes idle, perform the part sequencing algorithm for $m$; else continue;
Step 5. If a vehicle, denoted by $c$, becomes idle, perform the batch formation algorithm for $c$; else continue;
Step 6. Record the start time and completion time of the current operation;
Step 7. Set $t = t + 1$, go to Step3;
Step 8. Calculate the values of the objective function with the information recorded in Step 6.
```

Note that in the above DA, each machine is assigned a sequencing rule which may be different from those assigned to other machines.

The part sequencing algorithm is presented as follows, which indicates how to sequence the parts for a given machine using the obtained sequencing rule that is particularly assigned to the machine.

Part Sequencing Algorithm (PSA)

Step 1. Update SPS_m, the schedulable part set of machine m;

Step 2. For all of the parts in SPS_m, choose the best part i_{best} using the assigned sequencing rule of m;

Step 3. Set $SPS_m = SPS_m - \{i_{best}\}$ and remove i_{best} from the buffer of machine m.

The batch formation algorithm is presented as follows, which shows the process of constructing a batch for a given vehicle using the obtained batch formation rule that is particularly assigned to the vehicle.

For illustration, two costs that are used to evaluate the effect of a batch formation scheme are defined below. During batch formation, a part is assigned a cost, denoted $cost_p$, according to the ranking criterion of the selected batch formation rule. For example, if EDD is selected as the batch formation rule, the value of the part's cost is set to its due date. Accordingly, the cost of a batch, denoted $cost_b$, is defined as the average cost of parts in that batch, which is given in (9)

$$cost_b = \frac{1}{N}\sum_{i=1}^{N} cost_{pi} \tag{9}$$

where i represents the index of a part in batch b; N represents the total number of parts in batch b; and p_i represents the i-th part in batch b.

Batch Formation Algorithm (BFA)

Step 1. For each cell c' in CR_c, sequence all of the parts in $TRS_{cc'}$ by $cost_p$, where CR_c represents the set of the candidate destination cells for vehicle c, and $TRS_{cc'}$ is the set of the parts that need to be transported to c'. The sorted parts are then selected to form a possible batch, of which the size is equal to $min\{|TRS_{cc'}|, v\}$, where v is the capacity of vehicle c;

Step 2. For vehicle c, a batch with the minimum $cost_b$, denoted b_{best}, is selected from all possible batches constructed at the first stage;

Step 3. Remove b_{best} from the buffer of c;

Step 4. Vehicle c starts transporting b_{best}.

IV. Computational Experiments and Results

To evaluate the performance of the HEH approach, a series of computational experiments are conducted. All approaches are coded in JAVA and implemented on a PC with 3.40 GHz i7–2600 CPU and 2 GB RAM.

A. Experiment Design

Because there are no benchmarks available for the addressed problem, 20 test problems are generated randomly, with the number of parts ranging from 5 to 450, the number of machines from 6 to 120, and the number of cells from 3 to 15. A test problem is denoted by $p_{n1}m_{n2}c_{n3}$ which represents that there are n_1 parts to be processed on n_2 machines located in n_3 cells. The attributes of the test problems are generated according to the settings in Table V, and meanwhile, 10 instances are generated randomly for each of the test problem configurations. Therefore, there are a total of 200 instances, each of which is tested by five independent replications and the average value of the objective function is calculated and presented in the following sub-sections.

TABLE V
PARAMETERS OF THE TEST PROBLEMS

Parameters	Distributions
v (capacity of each vehicle)	~U [2,10]
r_i (release time of part i)	~U [0,50]
w_i (weight of part i)	~U [0,1]
$tt_{cc'}$ (transfer time from cell c to c')	~U [6,50]
P_{ijm} (processing time for the j-th operation of part i on machine m)	~U [1,30]

As the tightness of the due dates greatly affects the quality of solutions in the tardiness related criterion, the due dates of parts are generated using (10)

$$d_i = r_i + dl\sum_{j=1}^{O_i}\sum_{m=1}^{M}(p_{ijm} + st_{im}) \tag{10}$$

where r_i represents the release time of part i, dl is a constant which reflects the tightness factor of the due dates and a smaller value of dl indicates a tighter due date of the part, and $\sum_{j=1}^{O_i}\sum_{m=1}^{M}(p_{ijm} + st_{im})$ represents the sum of the processing time for all operations of the part. Inspired by the work in [26], loose due dates have twice the dl values of tight ones. It is assumed that in loose due date settings, the phenomenon that no jobs are tardy exists in one to five problem scales. After preliminary experiments we found it is appropriate to set six/three times of the mean part processing time as relatively loose/tight due date settings.

To avoid over-fitting, two data sets are defined, a training set and a test set. In the training set, four different scales of test problems are defined, i.e., $p5m6c3$, $p40m13c5$, $p70m20c7$, and $p100m25c9$. Though the same scale of the test problems in the training set also exists in the corresponding test set, the instances used are different. The data in the training set is employed as the input of the factorial experiments and all of the comparison experiments are conducted based on the data in the test set.

B. Parameter Settings

Due to the important effects of parameters in algorithms on their performance, four groups of factorial experiments are conducted to determine the best parameter combinations, including the parameters in RSGA, catastrophe operator, ATC and TS. TS has been successfully applied to solving the large job shop scheduling problems with modest CPU time [27], [28], and in this paper, it is used as a meta-heuristic to compare with the proposed HEH approach.

RSGA plays an important role in the HEH approach, and relies heavily on the parameter settings. Therefore, a full factorial

TABLE VI
PARAMETERS OF RSGA

Factors	Levels
ps	6, 12, 24, 48
pc	0.05, 0.30, 0.60, 0.90
pm	0.00, 0.02,0.10, 0.18
gm	25, 50, 100

experiment is conducted for the HEH approach in both loose and tight due date settings. The related parameters include population size (*ps*), crossover probability (*pc*), mutation probability (*pm*) and maximum generation (*gm*). Each parameter is considered as a factor whose levels are shown in Table VI.

According to Table VI, 192 combinations ($4 \times 4 \times 4 \times 3$) of parameters are generated and analyzed by a multifactor analysis of variance (ANOVA) at 95% confidence level. Prior to performing ANOVA, it is essential to verify its assumptions. The Statistical Product and Service Solutions (SPSS) software packet is used to conduct the tests. The assumptions ANOVA makes about the probability of the response are: independence of observations; and homogeneity of variances.

For the first assumption, the Kaiser-Meyer-Olkin (KMO) measure of sampling adequacy provides an index of the proportion of variance among the variables that might be caused by underlying factors. The results show that the values in loose and tight due date settings are 0.519 and 0.505, respectively. Therefore, the results of the factor analysis will be useful.

Bartlett's test of sphericity tests the hypothesis of equal variance. If the Sig. value for the test is less that the alpha level, the null hypothesis is rejected at 5% significant level. It turns out that the Sig. values in both loose and tight due date settings are 0.000 leading us to reject the null hypothesis and conclude that there are correlations in the data set that are appropriate for the factor analysis.

In ANOVA, each combination of the parameters is tested on four instances defined in the training set. Therefore, there are a total of 768 instances. Each instance is solved by five independent replications of the algorithm and the average value is taken as the performance measure.

The effects of all of the main factors are shown in Table VII, including ps, pc, pm, gm, and the potential two-factor interactions among them. In Table VII, the values in the rightmost column indicate the probabilities that the corresponding sources in the leftmost column have no effect.

For loose due dates, as shown in Table VII (a), the factors like ps, gm and pm*gm are important.

In order to identify the best combination of the four pairs of factors, it is important to find out how the two-factor interaction affects the response variable. It is observed that the value of the response variable reaches its best value when $ps = 48$ and $gm = 100$.

Similarly, for tight due dates, as shown in Table VII (b), the factors like ps, pm, gm and pc are of statistical significance, and so are ps*gm and pc*gm. It is also observed that when $ps = 48$ and $gm = 100$ or $ps = 48$ and $pm = 0.18$, the value of the response variable takes its best value.

In summary, in both loose and tight due date settings, when $ps = 48$, $pc = 0.90$, $pm = 0.18$, and $gm = 100$, the best performance is achieved, which also indicates the stability of the HEH approach. All of these values are set as default in the following experiments.

TABLE VII
ANOVA TABLE OF RSGA

(a) loose due date						
Source	Df	Seq SS	Adj SS	Adj MS	F	P
ps	3	5.94E+12	6.13E+12	2.04E+12	24.64	0.000
pc	3	2.81E+11	3.11E+11	1.04E+11	1.25	0.295
pm	3	3.93E+11	3.86E+11	1.29E+11	1.55	0.205
gm	2	2.96E+12	3.04E+12	1.52E+12	18.30	0.000
*ps*pc*	9	6.81E+11	7.02E+11	7.80E+10	0.94	0.493
*ps*pm*	9	1.04E+12	1.04E+12	1.16E+11	1.40	0.196
*ps*gm*	6	3.62E+11	3.82E+11	6.37E+10	0.77	0.596
*pc*pm*	9	8.77E+11	8.83E+11	9.81E+10	1.18	0.311
*pc*gm*	6	2.97E+11	2.86E+11	4.77E+10	0.57	0.750
*pm*gm*	6	1.47E+12	1.47E+12	2.45E+11	2.96	0.010
Deviation	134	1.11E+13	1.11E+13	8.30E+10		
Total	190	2.54E+13				
(b) tight due date						
Source	Df	Seq SS	Adj SS	Adj MS	F	P
ps	3	1.51E+13	1.51E+13	5.04E+12	88.58	0.000
pc	3	5.21E+11	5.22E+11	1.74E+11	3.05	0.031
pm	3	7.05E+11	7.01E+11	2.34E+11	4.10	0.008
gm	2	8.09E+12	7.97E+12	3.98E+12	69.98	0.000
*ps*pc*	9	4.85E+11	4.99E+11	5.54E+10	0.97	0.465
*ps*pm*	9	7.52E+12	7.58E+11	8.42E+10	1.48	0.162
*ps*gm*	6	1.10E+12	1.10E+12	1.83E+11	3.21	0.006
*pc*pm*	9	7.55E+11	7.56E+11	8.40E+10	1.48	0.163
*pc*gm*	6	1.02E+12	1.02E+12	1.70E+11	2.99	0.009
*pm*gm*	6	2.24E+11	2.24E+11	3.74E+10	0.66	0.685
Deviation	134	7.63E+12	7.63E+12	5.69E+10		
Total	190	3.64E+13				

As for the catastrophe operator, two parameters, i.e., the terminal point of the early period of the evolutionary process, denoted by k_1, and that of the middle period of the evolutionary process, denoted by k_2, need to be specified before the experiments begin. Therefore, the values of k_1 and $k_2 (k_2 > k_1)$ are selected from three levels respectively, i.e., $\{1/5gm, 2/5gm, 3/5gm\}$ and $\{2/5gm, 3/5gm, 4/5gm\}$. After running on the training set in both loose and tight due date settings, k_1 and k_2 are respectively set 0.2 and 0.8 in loose due date settings, and 0.2 and 0.4 in tight due date settings. Besides, the lower limit of the catastrophe updating factor $\tau_{\min}$ is set 0 and its upper limit $\tau_{\max}$ is set 1.

A parameter denoted by K in ATC also needs to be set in advance. As stated in [26], $K$is a look-ahead parameter that scales the slack (measured in units of average processing time) according to the expected number of competing parts. The suitable value of K is selected from $\{2, 2.5, 3, 3.5, 4\}$. By running on the training set, 3 is selected as the default value in the following experiments.

There are three parameters in TS that need to be set beforehand, i.e., the termination condition (TC), the length of the tabu list (LTL), and the number of candidate solutions in the subset of the unselected neighbors (NCS). All these parameters are set after running on the training set. TC is adapted with the variation of the problem scales. Out of the four levels of LTL, i.e., 3, 5, 8 and 10, 3 is finally selected. As for NCS, out of the four candidates, i.e., 5, 10, 15, and 20, 10 is selected, which means that 10 candidate solutions are searched in the neighborhood of the current solution.

C. Comparison Experiments

To evaluate the solution quality and computational efficiency of the HEH approach, three hyper-heuristics, i.e., GA-based HH, GP-based HH and 2-stage HH, which are originally designed for other scheduling problems, are introduced here. In GA-based HH, GA is adopted to search for appropriate combinations of the heuristic rules in the given candidate rule set, and the selected rules are then used for the generation of the scheduling solutions. In GP-based HH, suitable heuristic rules are designed through exploiting the structure of the addressed problem and combining good heuristic components, and the schedules are determined according to the generated rules. 2-stage HH is derived from [24], which is the integration of GP-based HH and GA-based HH. HEH is highly correlated to the above approaches and it deviates from 2-stage HH in the adoption of the catastrophe operator and the hill climbing algorithm.

Six groups of experiments are designed in this sub-section: i) to compare HEH with combinatorial rules; ii) to evaluate the effect of GP by comparing 2-stage HH with GA-based HH; iii) to analyze the structure of the GP-generated rules by comparing them with the best-performing human-made rules; iv) to evaluate the effect of GA by comparing 2-stage HH with GP-based HH; v) to compare HEH with 2-stage HH; and vi) to compare HEH with TS.

1) Comparison With Combinatorial Rules: To compare with HEH, 52 combinatorial rules are designed. Each sub-problem, i.e., part sequencing for machines and batch formation for vehicles, is assigned a heuristic rule. By illustration, only three combinatorial heuristic rules with the best performance, are listed in this sub-section and presented in Table VIII.

The gap which shows the percentage deviation between the best three combinatorial rules and the HEH approach is defined in (11)

$$Gap_{Comb} = \frac{score_{rules} - score_{HEH}}{score_{HEH}} \tag{11}$$

where $score_{rules}$ and $score_{HEH}$ represent the average objective function values obtained by the best or worst three combinatorial heuristic rules and that obtained by HEH, respectively.

As shown in Table IX, in loose due date settings, for the best cases of different test problems, the gap ranges from 16.9% to 1201.2%, and HEH outperforms the best three combinatorial rules in all test problems with an average gap of 176.7%, as shown in Table IX (a). While for the worst cases, the gap ranges from 0.4% to 122.7%, and the average gap is 20.4%, indicating HEH outperforms the worst three combinatorial rules in all test problems.

TABLE VIII
COMBINATORIAL RULES FOR COMPARISON WITH HEH

(a) loose due date		
Sequencing rule	Batch formation rule	Notation
SRPT	EDD	CombL1
ATC	SETT	CombL2
WEDD	EDD	CombL3
(b) tight due date		
Sequencing rule	Batch formation rule	Notation
WEDD	EDD	CombT1
ATC	SETT	CombT2
WSPT	SETT	CombT3

Similarly, in tight due date settings, for the best cases, HEH outperforms the best three combinatorial rules in all test problems with an average gap of 35.6%. While for the worst cases, HEH outperforms the worst three combinatorial rules in all test problems and the average gap is 9.8%, as shown in Table IX (b).

The reason for the superiority of HEH lies in that the combinatorial rules are given in advance and rely on the expertise of human experience. In contrast, in HEH, GP is used for the automatic design of heuristic rules which exploits the structure of the addressed problem and GA is used to select suitable rules for machines and vehicles. Therefore, the rules in HEH are more effective in solving the addressed problem.

Moreover, it is observed in Table IX that, in loose due date settings, two out of three best sequencing rules and all of the best three batch formation rules are due date related. Such an observation is more obvious in the tight due date settings, where all of the best six heuristic rules that constitute the combinatorial rules are due date related. As the objective is to minimize TWT, it is not surprising that due date related rules are preferable.

To analyze the variation of the performance for the combinatorial rules as the scale of test problems increases, the box plots are given in Fig. 4. The vertical axis in Fig. 4 represents the percentage deviation between each of the combinatorial rules and the average level of all combinatorial rules, which is defined in (12)

$$Gap_{rule} = \frac{score_{rule} - score_{avg}}{score_{avg}} \tag{12}$$

where $score_{rule}$represents the objective function value obtained by each of the combinatorial rules and $score_{avg}$ represents the average level of the objective function values obtained by all of the combinatorial rules, respectively.

Among all of the test problems in the test set, three are selected as the representatives of different scales, i.e., $p50m15c5$ for the small scale, denoted S, $p100m25c9$ for the moderate scale, denoted M, and $p200m50c15$ for the large scale, denoted L.

In Fig. 4, the lower quartile, the median quartile, and the upper quartile are depicted on a box and the lines extending from each end of the box show the range of the remaining data.

In loose due date settings, as shown in Fig. 4(a), a large deviation exists between the lower and upper quartiles in the small

TABLE IX
COMPARISON BETWEEN THE COMBINATORIAL RULES AND HEH

Problem scale	(a) loose due date						(b) tight due date					
	3-Combs		HEH		Gap_{Comb} (%)		3-Combs		HEH		Gap_{Comb} (%)	
	min	max	min	max	min	max	min	max	min	max	min	max
*p*5*m*6*c*3	0.00E+00	2.77E+02	0.00E+00	0.00E+00	-	-	1.24E+03	1.95E+03	1.16E+03	1.71E+03	7.0	13.7
*p*15*m*8*c*3	0.00E+00	4.67E+03	0.00E+00	2.10E+03	-	122.7	3.04E+04	4.41E+04	1.58E+04	4.21E+04	92.3	4.7
*p*20*m*11*c*3	0.00E+00	9.25E+03	0.00E+00	7.07E+03	-	30.9	3.26E+04	5.24E+04	1.65E+04	4.90E+04	97.6	6.8
*p*40*m*13*c*5	3.84E+04	4.96E+05	0.00E+00	2.30E+05	-	115.3	6.62E+05	1.34E+06	4.26E+05	9.69E+05	55.1	38.3
*p*50*m*15*c*5	2.80E+05	1.41E+06	4.28E+04	1.28E+06	553.3	10.2	1.44E+06	3.28E+06	1.05E+06	3.05E+06	36.4	7.8
*p*60*m*16*c*5	6.82E+05	1.75E+06	5.25E+04	1.39E+06	1201.2	25.9	3.11E+06	4.64E+06	1.98E+06	4.10E+06	57.1	13.4
*p*70*m*20*c*7	1.13E+06	3.63E+06	1.58E+05	2.62E+06	610.9	38.6	6.42E+06	9.53E+06	4.30E+06	8.28E+06	49.4	15.0
*p*80*m*21*c*7	1.16E+07	1.52E+07	6.70E+06	1.44E+07	73.0	5.6	2.22E+07	2.72E+07	1.70E+07	2.55E+07	30.2	6.5
*p*90m21*c*7	1.61E+07	1.86E+07	8.22E+06	1.79E+07	96.1	3.8	2.97E+07	3.53E+07	2.15E+07	3.35E+07	38.2	5.3
*p*100*m*25*c*9	2.29E+07	2.85E+07	1.59E+07	2.84E+07	44.2	0.4	4.11E+07	4.77E+07	3.23E+07	4.60E+07	27.3	3.9
*p*120*m*30*c*9	7.89E+07	8.75E+07	5.66E+07	8.58E+07	39.5	2.0	1.08E+08	1.49E+08	9.01E+07	1.22E+08	19.7	22.3
*p*140*m*35*c*11	1.05E+08	1.23E+08	7.84E+07	1.17E+08	33.3	5.0	1.73E+08	1.96E+08	1.36E+08	1.80E+08	27.2	9.2
*p*160*m*40*c*11	1.73E+08	1.97E+08	1.37E+08	1.90E+08	26.3	3.8	2.71E+08	2.97E+08	2.17E+08	2.89E+08	24.8	2.9
*p*180*m*45*c*13	3.84E+08	4.30E+08	2.94E+08	4.24E+08	30.3	1.3	5.18E+08	5.91E+08	4.17E+08	5.41E+08	24.2	9.3
*p*200*m*50*c*15	4.81E+08	5.27E+08	3.93E+08	5.15E+08	22.4	2.4	6.75E+08	7.06E+08	5.32E+08	6.93E+08	26.8	1.8
*p*250*m*65*c*15	1.27E+09	1.37E+09	1.08E+09	1.34E+09	16.9	2.5	1.68E+09	1.77E+09	1.42E+09	1.72E+09	17.8	3.1
*p*300*m*75*c*15	2.31E+09	2.50E+09	1.92E+09	2.39E+09	20.4	4.3	2.92E+09	3.08E+09	2.37E+09	2.88E+09	23.0	6.9
*p*350*m*90*c*15	3.63E+09	3.94E+09	3.06E+09	3.73E+09	18.6	5.7	4.62E+09	4.72E+09	3.81E+09	4.57E+09	21.1	3.3
*p*400*m*100*c*15	8.80E+09	9.28E+09	7.15E+09	8.93E+09	23.0	4.0	1.01E+10	1.05E+10	8.46E+09	9.25E+09	18.9	13.8
*p*450*m*120*c*15	1.08E+10	1.15E+10	9.07E+09	1.11E+10	18.5	3.4	1.27E+10	1.33E+10	1.07E+10	1.22E+10	18.3	8.2
Average					176.7	20.4					35.6	9.8

scale, indicating a big fluctuation in the performance of the combinatorial rules. As the problem scale increases, the range between the upper quartile and lower quartile narrows down gradually until all of the solutions have the similar performance and converge to the median quartile.

Similarly, in tight due date settings, as shown in Fig. 4(b), a decline in the range between the upper quartile and lower quartile is also observed as the problem scale grows larger, and the performance of the combinatorial rules gradually stabilizes and a similar performance is finally achieved.

The reason for the above observations is mainly due to the greedy nature of the heuristic rules, and the disadvantage due to the lack of solution quality gets even more obvious as the size increases.

2) Effects of GP: 2-stage HH is an extension of the GA-based HH approach, which additionally takes GP into consideration. To evaluate the performance of GP in generating heuristic rules, a comparison experiment between 2-stage HH and GA-based HH is conducted. The gap which shows the percentage deviation between GA-based HH and 2-stage HH is defined in (13)

$$Gap_{GA} = \frac{score_{GA-based\ HH} - score_{2-stage\ HH}}{score_{2-stage\ HH}} \quad (13)$$

where $score_{GA-based\ HH}$ and $score_{2-stage\ HH}$ represent the average objective function values obtained by GA-based HH and 2-stage HH, respectively.

In loose due date settings, for the best cases of different test problems, the gap ranges from 0.6% to 8.2%, and 2-stage HH outperforms GA-based HH in all test problems with an average gap of 4.4%, as shown in Table X (a). While for the worst cases, the gap ranges from 0.3% to 38.7%, and the average gap is 10.0%, indicating 2-stage HH outperforms GA-based HH in all test problems.

Similarly, in tight due date settings, for the best cases, 2-stage HH outperforms GA-based HH in 19 out of 20 test problems with an average gap of 5.1%. While for the worst cases, 2-stage HH outperforms GA-based HH in 18 out of 20 test problems and the average gap is 4.0%, as shown in Table X (b).

The data presented in Table X indicates the effectiveness and stability of 2-stage HH in both loose and tight due date settings, and the advantage of 2-stage HH is more obvious in loose due date settings. This is because that, in tight due date settings, most parts cannot be completed on time, which makes it not that important what the rule is like.

Moreover, the evolutionary processes of 2-stage HH, GA-based HH, GP-based HH, and HEH are given in Fig. 5 for illustration, and in this sub-section, the lines of 2-stage HH and GA-based HH are mainly considered and analyzed.

It is observed from Fig. 5 that in both loose and tight due date settings, 2-stage HH provides much better initial solutions than GA-based HH, and converges to better final solutions in the end. This is because the initial solution in 2-stage HH is obtained from the GP-generated rule, which has been evolved according

TABLE X
COMPARISON BETWEEN GA-BASED HH AND 2-STAGE HH

Problem scale	(a) loose due date						(b) tight due date					
	GA-based HH		2-stage HH		Gap_{GA} (%)		GA-based HH		2-stage HH		Gap_{GA} (%)	
	min	max	min	max	min	max	min	max	min	max	min	max
*p*5*m*6*c*3	0.00E+00	2.72E+02	0.00E+00	2.20E+02	-	23.6	1.16E+03	2.09E+03	1.16E+03	2.10E+03	0.0	-0.5
*p*15*m*8*c*3	0.00E+00	6.37E+03	0.00E+00	4.59E+03	-	38.7	1.92E+04	4.70E+04	1.66E+04	4.60E+04	15.4	2.2
*p*20*m*11*c*3	0.00E+00	1.04E+04	0.00E+00	8.81E+03	-	18.1	2.07E+04	5.65E+04	1.73E+04	5.24E+04	20.0	7.9
*p*40*m*13*c*5	0.00E+00	4.20E+05	0.00E+00	3.59E+05	-	16.8	4.82E+05	1.12E+06	4.54E+05	1.07E+06	6.2	5.5
*p*50*m*15*c*5	5.97E+04	1.46E+06	5.52E+04	1.28E+06	8.2	13.8	1.12E+06	3.37E+06	1.11E+06	3.20E+06	0.7	5.4
*p*60*m*16*c*5	9.82E+04	1.96E+06	9.29E+04	1.47E+06	5.8	33.3	2.23E+06	4.77E+06	2.10E+06	4.63E+06	6.2	3.1
*p*70*m*20*c*7	4.85E+05	3.24E+06	4.69E+05	3.15E+06	3.3	3.0	4.80E+06	9.51E+06	4.56E+06	9.04E+06	5.2	5.2
*p*80*m*21*c*7	8.09E+06	1.57E+07	7.88E+06	1.53E+07	2.7	2.8	1.85E+07	2.68E+07	1.79E+07	2.65E+07	3.4	1.1
*p*90m21*c*7	9.93E+06	2.03E+07	9.88E+06	2.00E+07	0.6	1.2	2.47E+07	3.78E+07	2.40E+07	3.56E+07	2.9	6.4
*p*100*m*25*c*9	1.86E+07	3.18E+07	1.80E+07	3.05E+07	3.5	4.5	3.57E+07	5.42E+07	3.56E+07	4.94E+07	0.2	9.8
*p*120*m*30*c*9	6.48E+07	9.91E+07	6.10E+07	9.47E+07	6.2	4.6	1.00E+08	1.29E+08	9.54E+07	1.27E+08	5.0	1.6
*p*140*m*35*c*11	8.78E+07	1.29E+08	8.32E+07	1.25E+08	5.6	2.9	1.55E+08	1.96E+08	1.46E+08	1.90E+08	6.3	3.0
*p*160*m*40*c*11	1.54E+08	2.08E+08	1.47E+08	2.07E+08	4.4	0.8	2.47E+08	3.11E+08	2.37E+08	3.03E+08	4.5	2.4
*p*180*m*45*c*13	3.31E+08	4.87E+08	3.11E+08	4.43E+08	6.4	10.0	4.71E+08	5.70E+08	4.60E+08	5.72E+08	2.4	-0.4
*p*200*m*50*c*15	4.42E+08	5.46E+08	4.17E+08	5.44E+08	6.0	0.3	6.25E+08	7.37E+08	6.00E+08	7.27E+08	4.3	1.4
*p*250*m*65*c*15	1.20E+09	1.50E+09	1.15E+09	1.41E+09	4.4	6.4	1.64E+09	1.88E+09	1.53E+09	1.78E+09	7.1	5.4
*p*300*m*75*c*15	2.09E+09	2.74E+09	2.04E+09	2.55E+09	2.6	7.6	2.67E+09	3.26E+09	2.61E+09	3.16E+09	2.2	3.2
*p*350*m*90*c*15	3.39E+09	3.99E+09	3.28E+09	3.89E+09	3.4	2.6	4.40E+09	4.93E+09	4.19E+09	4.87E+09	5.1	1.0
*p*400*m*100*c*15	7.81E+09	9.79E+09	7.52E+09	9.49E+09	3.9	3.2	9.41E+09	1.21E+10	9.18E+09	1.10E+10	2.5	9.7
*p*450*m*120*c*15	9.98E+09	1.24E+10	9.63E+09	1.17E+10	3.6	6.0	1.22E+10	1.48E+10	1.18E+10	1.38E+10	2.9	7.1
Average					4.4	10.0					5.1	4.0

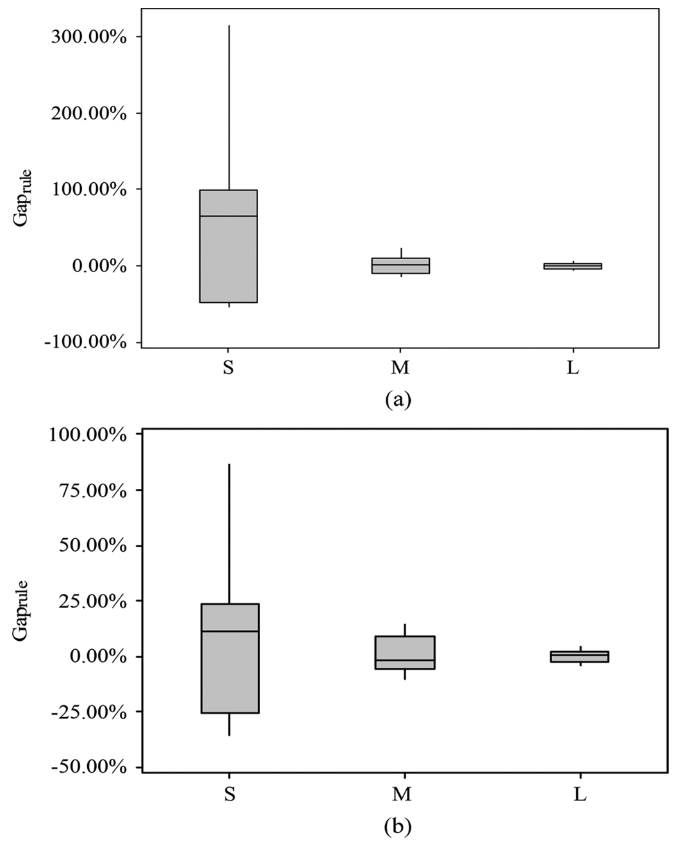


Fig. 4. Box plots of the combinatorial rules. (a) Loose due date. (b) Tight due date.

to the characteristics of the addressed problem. GP is able to exploit the structure of the addressed problem with the given attributes of the machines and vehicles, which is fundamental in the evolution of a more suitable heuristic rule. Through the extension of the candidate rule set, GP enables the hyper-heuristic to search in a wider exploration scope. Due to the importance of the candidate rules, it is beneficial to make GP a heuristic generation methodology in the hyper-heuristic.

3) Analysis of the Structure of GP-Generated Rules: It was reported in [26] that for the job shop scheduling problem with due date related objectives, some heuristic rules like ATC and WEDD have outperformed other competing rules. Therefore, to evaluate the effectiveness of the GP-generated rules, they are compared with ATC and WEDD in this paper.

Because two sub-problems, i.e., part sequencing for machines and batch formation for vehicles, are considered, heuristic rules are designed for them, respectively. Owing to the similarity in the generation of the sequencing rules and batch formation rules, the experiments in this sub-section mainly focus on the sequencing rules. In order to analyze the sequencing rules without being influenced by the batch formation dimension, the comparison experiments are conducted based on the premise that the batch formation rules for all vehicles are set to be the same in advance.

The GP-generated sequencing rule is compared with ATC, which is the best heuristic rule for the weighted tardiness criterion found so far [26]. The gap which shows the percentage

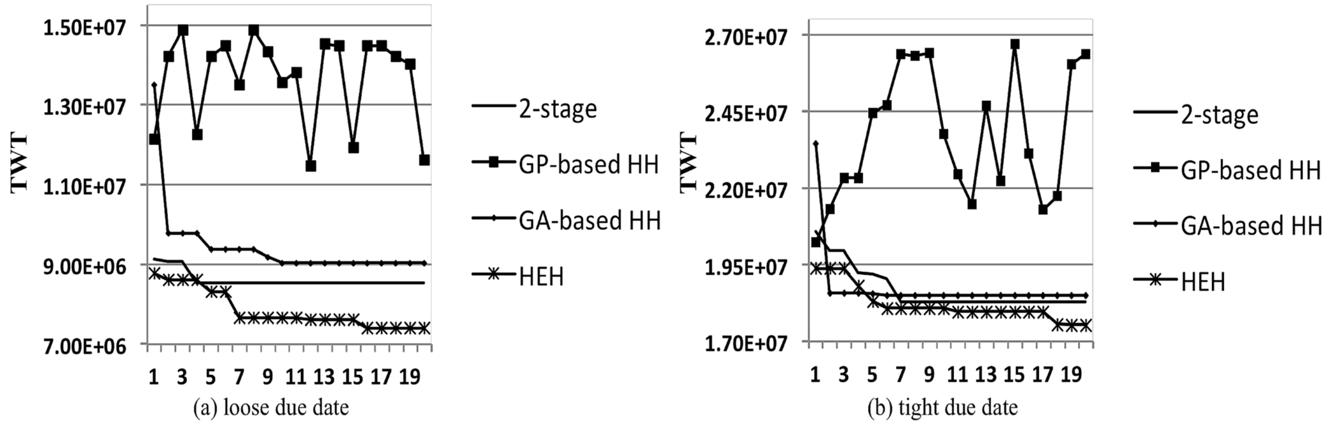


Fig. 5. Evolutionary processes of GP-based HH, GA-based HH, 2-stage HH and HEH.

deviation between the GP-generated rule and ATC is defined in (14)

$$Gap_{SGP} = \frac{score_{ATC} - score_{SGP}}{score_{SGP}} \quad (14)$$

where $score_{SGP}$ and $score_{ATC}$ represent the objective function values obtained by the GP-generated rule and ATC, respectively.

As shown in Table XI, Gap_{SGP1}, Gap_{SGP2} and Gap_{SGP3} represent the gaps between the best three GP-generated sequencing rules, i.e., SGP1, SGP2 and SGP3 in Table III, and ATC, respectively.

It is observed from Table XI that, for loose due dates, all of the best three GP-generated rules dominate ATC in most of the test problems, with average gaps of 8.6%, 6.7% and 7.9%, respectively. It is similar for tight due dates, where the overall performance of the best three GP-generated rules is better than that of ATC, with the average gaps of 7.6%, 11.9% and 10.8%, respectively.

To observe what lies behind the above results, it is essential to study the structure of the GP-generated rules.

Though the rules generated by GP seem complicated, it is found that several sub-trees with various replications in the rules can be simplified, i.e., sub-trees like Div(X, X) and Sub(X, X) can be reduced to a constant, and sub-tree Sub($\mathrm{X}+\mathrm{Y}$, X) can be reduced to Y. By illustration, an example of the simplification of a GP-generated rule is given in Fig. 6. It is observed that the sub-trees like Max(X, X) can be reduced to X. In this way, the GP-generated rules are simplified considerably.

After the above simplification, the structure of the simplified rules is further studied. Similarities with ATC and WEDD are found in two out of the best three GP-generated rules, as shown in Fig. 7.

In the first GP-generated rule, two sub-trees, i.e., Div(W, PT) and Div(DD, W), are repeated three times. The replications indicate the contribution of these sub-trees to the effectiveness of the generated rule. It should be noted that Div (W, PT) is a function representation of WSPT, which is also a special case of ATC, and Div(DD, W) corresponds to WEDD. That is, the generated rule is essentially a ranking expression, which combines several human-made heuristic rules with other heuristic components.

For example, sub-tree Div(W, PT) in the first GP-generated rule is one of the extreme case of ATC which can be reduced to WSPT when the look-ahead parameter K is large enough. Sub-tree Max(Sub(Sub(DD, PT), Max(RPT, W)), 0.1489462) in the second GP-generated rule has obtained by reducing $K$to a certain value.

In summary, the heuristic rules designed by GP are of similar structure to the best-performing human-made rules for weighted tardiness criterion, which indicates the effectiveness of the GP-generated rules. In addition, more problem-specific rules can be generated by GP due to the appropriate selection of heuristic components and suitable algebraic operations on these components, and therefore, the GP-generated rules can achieve better performance than the best performing human-made rules.

4) Effects of GA: The 2-stage HH approach is also an extended version of the GP-based HH approach, which additionally takes GA into consideration. A comparison experiment between GP-based HH and 2-stage HH is conducted to evaluate the effects of GA in the selection of combinations of rules. The gap which shows the percentage deviation between GP-based HH and 2-stage HH is defined in (15)

$$Gap_{GP} = \frac{score_{GP-based\ HH} - score_{2-stage\ HH}}{score_{2-stage\ HH}} \quad (15)$$

where $score_{GP-based\ HH}$ and $score_{2-stage\ HH}$ represent the average objective function values obtained by GP-based HH and 2-stage HH, respectively.

In loose due date settings, for the best cases, the gap ranges from 2.7% to 92.4%, and 2-stage HH outperforms GP-based HH in all test problems with an average gap of 30.3%, as shown in Table XII (a). While for the worst cases, the gap ranges from 23.2% to 205.3%, and the average gap is 78.7%, indicating 2-stage HH outperforms GP-based HH in all test problems.

Similarly, in tight due date settings, for the best cases, 2-stage HH outperforms GP-based HH in all test problems with an average gap of 24.1%. While for the worst cases, 2-stage HH outperforms GP-based HH in all test problems and the average gap is 53.1%, as shown in Table XII (b).

The result in Table XII also reflects the effectiveness and stability of 2-stage HH, and the advantage of 2-stage HH is more obvious in loose due date settings. The reason is the same as that illustrated in Section IV-C2. That is, owing to the tight due

TABLE XI
COMPARISON BETWEEN THE BEST THREE GP-GENERATED RULES AND ATC

(a) loose due date

Problem scale	SGP1	SGP2	SGP3	ATC	Gap_{SGP1} (%)	Gap_{SGP2} (%)	Gap_{SGP3} (%)
p5m6c3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-
p15m8c3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-
p20m11c3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-
p40m13c5	0.00E+00	0.00E+00	0.00E+00	4.85E+04	-	-	-
p50m15c5	3.22E+05	2.92E+05	3.11E+05	3.12E+05	-3.2	6.7	0.3
p60m16c5	6.18E+05	6.07E+05	6.12E+05	6.64E+05	7.3	9.3	8.4
p70m20c7	8.29E+05	9.25E+05	7.90E+05	1.12E+06	35.1	21.1	41.9
p80m21c7	1.11E+07	1.05E+07	1.16E+07	1.16E+07	4.9	10.9	0.2
p90m21c7	1.19E+07	1.51E+07	1.08E+07	1.63E+07	36.7	7.9	51.6
p100m25c9	2.16E+07	2.43E+07	2.47E+07	2.36E+07	9.2	-3.0	-4.5
p120m30c9	6.97E+07	7.42E+07	7.10E+07	7.43E+07	6.5	0.2	4.6
p140m35c11	1.06E+08	1.03E+08	1.05E+08	1.05E+08	-0.8	2.7	0.2
p160m40c11	1.54E+08	1.46E+08	1.62E+08	1.67E+08	8.2	14.5	2.8
p180m45c13	3.19E+08	3.60E+08	3.75E+08	3.98E+08	24.8	10.5	6.2
p200m50c15	4.66E+08	4.75E+08	4.70E+08	4.81E+08	3.2	1.1	2.4
p250m65c15	1.31E+09	1.22E+09	1.26E+09	1.25E+09	-5.0	2.5	-0.8
p300m75c15	2.33E+09	2.32E+09	2.35E+09	2.37E+09	1.8	2.0	0.8
p350m90c15	3.48E+09	3.52E+09	3.40E+09	3.59E+09	3.0	1.9	5.4
p400m100c15	8.35E+09	7.99E+09	8.16E+09	8.42E+09	0.8	5.4	3.1
p450m120c15	1.05E+10	9.72E+09	1.07E+10	1.10E+10	5.2	13.3	2.9
Average					8.6	6.7	7.9

(a) loose due date

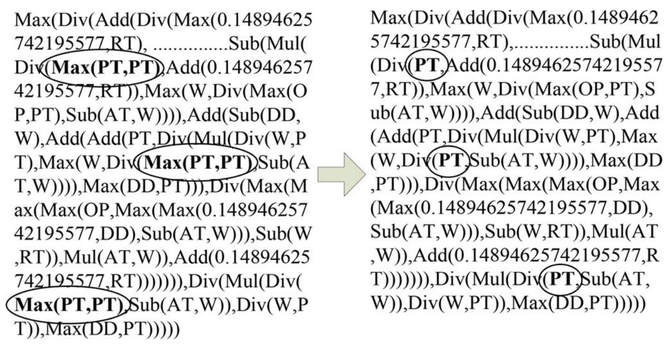
Problem scale	SGP1	SGP2	SGP3	ATC	Gap_{SGP1} (%)	Gap_{SGP2} (%)	Gap_{SGP3} (%)
p5m6c3	1.20E+03	1.24E+03	1.20E+03	1.16E+03	-3.0	-5.9	-3.0
p15m8c3	2.49E+04	2.10E+04	2.45E+04	3.35E+04	34.4	59.1	36.7
p20m11c3	2.75E+04	2.51E+04	2.87E+04	4.18E+04	51.7	66.3	45.7
p40m13c5	7.02E+05	6.13E+05	6.13E+05	6.33E+05	-9.8	3.3	3.2
p50m15c5	1.26E+06	1.18E+06	1.14E+06	1.86E+06	47.8	57.7	63.8
p60m16c5	3.00E+06	2.78E+06	2.82E+06	3.29E+06	9.8	18.1	16.8
p70m20c7	6.62E+06	6.47E+06	6.18E+06	6.71E+06	1.4	3.7	8.6
p80m21c7	2.30E+07	2.22E+07	2.17E+07	2.19E+07	-4.7	-1.2	0.9
p90m21c7	2.83E+07	2.69E+07	2.75E+07	2.93E+07	3.4	8.6	6.6
p100m25c9	4.19E+07	4.47E+07	4.26E+07	4.54E+07	8.4	1.7	6.8
p120m30c9	1.09E+08	1.06E+08	1.07E+08	1.03E+08	-5.1	-2.4	-3.6
p140m35c11	1.69E+08	1.62E+08	1.66E+08	1.77E+08	4.8	9.3	6.4
p160m40c11	2.74E+08	2.70E+08	2.62E+08	2.74E+08	-0.2	1.5	4.6
p180m45c13	4.93E+08	4.83E+08	4.98E+08	5.13E+08	4.1	6.3	3.1
p200m50c15	6.71E+08	6.63E+08	6.24E+08	6.75E+08	0.6	1.9	8.2
p250m65c15	1.71E+09	1.72E+09	1.71E+09	1.72E+09	0.2	0.1	0.3
p300m75c15	2.87E+09	2.89E+09	2.87E+09	2.90E+09	1.1	0.4	1.1
p350m90c15	4.60E+09	4.60E+09	4.54E+09	4.53E+09	-1.5	-1.5	0.0
p400m100c15	9.76E+09	9.82E+09	9.82E+09	9.89E+09	1.3	0.7	0.7
p450m120c15	1.19E+10	1.17E+10	1.17E+10	1.28E+10	7.4	9.5	9.0
Average					7.6	11.9	10.8

dates, it is difficult for parts to complete processing no matter what rule is adopted.

Moreover, the evolutionary processes of 2-stage HH and GP-based HH are also given in Fig. 5.

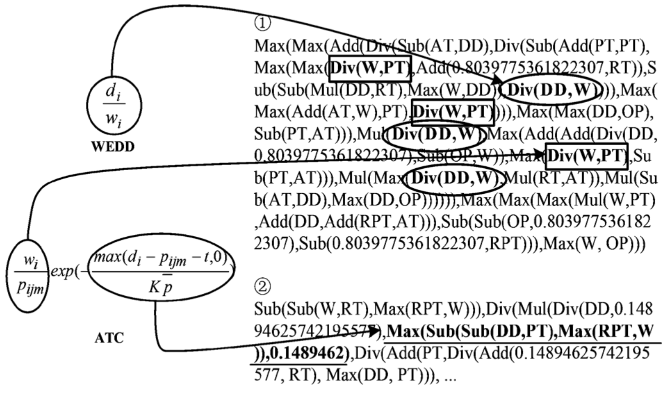
TABLE XII
COMPARISON BETWEEN GP-BASED HH AND 2-STAGE HH

Problem scale	(a) loose due date						(b) tight due date					
	GP-based HH		2-stage HH		*Gap*$_{GP}$(%)		GP-based HH		2-stage HH		*Gap*$_{GP}$(%)	
	min	max	min	max	min	max	min	max	min	max	min	max
*p*5*m*6*c*3	0.00E+00	3.12E+02	0.00E+00	2.20E+02	-	41.8	1.20E+03	3.24E+03	1.16E+03	2.10E+03	3.1	53.9
*p*15*m*8*c*3	0.00E+00	7.56E+03	0.00E+00	4.59E+03	-	64.6	2.49E+04	8.43E+04	1.66E+04	4.60E+04	50.0	83.1
*p*20*m*11*c*3	0.00E+00	2.28E+04	0.00E+00	8.81E+03	-	159.2	2.75E+04	9.40E+04	1.73E+04	5.24E+04	59.6	79.5
*p*40*m*13*c*5	7.73E+03	6.92E+05	0.00E+00	3.59E+05	-	92.6	7.02E+05	1.68E+06	4.54E+05	1.07E+06	54.5	57.7
*p*50*m*15*c*5	9.89E+04	3.39E+06	5.52E+04	1.28E+06	79.1	164.8	1.26E+06	5.49E+06	1.11E+06	3.20E+06	13.8	71.7
*p*60*m*16*c*5	1.79E+05	4.49E+06	9.29E+04	1.47E+06	92.4	205.3	3.00E+06	5.96E+06	2.10E+06	4.63E+06	42.9	28.8
*p*70*m*20*c*7	5.65E+05	5.27E+06	4.69E+05	3.15E+06	20.3	67.4	6.62E+06	1.44E+07	4.56E+06	9.04E+06	45.2	59.3
*p*80*m*21*c*7	1.11E+07	3.14E+07	7.88E+06	1.53E+07	40.2	105.7	2.58E+07	3.55E+07	1.79E+07	2.65E+07	44.1	34.1
*p*90m21*c*7	1.53E+07	2.94E+07	9.88E+06	2.00E+07	54.9	47.1	2.93E+07	4.94E+07	2.40E+07	3.56E+07	22.1	39.0
*p*100*m*25*c*9	2.52E+07	3.75E+07	1.80E+07	3.05E+07	40.1	23.2	4.19E+07	6.93E+07	3.56E+07	4.94E+07	17.7	40.3
*p*120*m*30*c*9	7.73E+07	1.58E+08	6.10E+07	9.47E+07	26.7	66.4	1.13E+08	2.12E+08	9.54E+07	1.27E+08	18.1	66.7
*p*140*m*35*c*11	1.18E+08	1.96E+08	8.32E+07	1.25E+08	41.4	56.9	1.69E+08	2.81E+08	1.46E+08	1.90E+08	15.8	48.0
*p*160*m*40*c*11	1.54E+08	2.98E+08	1.47E+08	2.07E+08	4.6	44.0	2.92E+08	5.71E+08	2.37E+08	3.03E+08	23.3	88.3
*p*180*m*45*c*13	3.19E+08	5.72E+08	3.11E+08	4.43E+08	2.7	29.0	4.93E+08	6.92E+08	4.60E+08	5.72E+08	7.1	21.0
*p*200*m*50*c*15	4.66E+08	8.52E+08	4.17E+08	5.44E+08	11.7	56.7	6.84E+08	1.08E+09	6.00E+08	7.27E+08	14.1	48.9
*p*250*m*65*c*15	1.31E+09	2.00E+09	1.15E+09	1.41E+09	14.5	42.3	1.71E+09	2.51E+09	1.53E+09	1.78E+09	11.9	41.2
*p*300*m*75*c*15	2.33E+09	6.43E+09	2.04E+09	2.55E+09	14.2	152.2	2.87E+09	4.77E+09	2.61E+09	3.16E+09	10.0	50.9
*p*350*m*90*c*15	3.82E+09	5.98E+09	3.28E+09	3.89E+09	16.7	53.9	4.60E+09	5.90E+09	4.19E+09	4.87E+09	9.9	21.1
*p*400*m*100*c*15	8.43E+09	1.34E+10	7.52E+09	9.49E+09	12.0	40.6	9.98E+09	1.77E+10	9.18E+09	1.10E+10	8.7	60.9
*p*450*m*120*c*15	1.09E+10	1.88E+10	9.63E+09	1.17E+10	13.7	60.2	1.30E+10	2.32E+10	1.18E+10	1.38E+10	9.8	68.0
Average					30.3	78.7					24.1	53.1


```
Max(Div(Add(Div(Max(0.14894625
742195577,RT), ...............Sub(Mul(
Div(Max(PT,PT)),Add(0.148946257
42195577,RT)),Max(W,Div(Max(O
P,PT),Sub(AT,W)))),Add(Sub(DD,
W),Add(Add(PT,Div(Mul(Div(W,P
T),Max(W,Div(Max(PT,PT),Sub(A
T,W)))),Max(DD,PT))),Div(Max(M
ax(Max(OP,Max(Max(0.148946257
42195577,DD),Sub(AT,W))),Sub(W
,RT)),Mul(AT,W)),Add(0.14894625
742195577,RT))))))),Div(Mul(Div(
Max(PT,PT),Sub(AT,W)),Div(W,P
T)),Max(DD,PT)))))
```

```
Max(Div(Add(Div(Max(0.1489462
5742195577,RT),...............Sub(Mul
(Div(PT),Add(0.1489462574219557
7,RT)),Max(W,Div(Max(OP,PT),S
ub(AT,W)))),Add(Sub(DD,W),Add
(Add(PT,Div(Mul(Div(W,PT),Max
(W,Div(PT,Sub(AT,W)))),Max(DD
,PT))),Div(Max(Max(Max(OP,Max
(Max(0.14894625742195577,DD),
Sub(AT,W))),Sub(W,RT)),Mul(AT
,W)),Add(0.14894625742195577,R
T))))))),Div(Mul(Div(PT,Sub(AT,
W)),Div(W,PT)),Max(DD,PT)))))
```

Fig. 6. Simplification of a GP-generated rule.



$\frac{d_i}{w_i}$
WEDD
$\frac{w_i}{p_{ijm}} exp(-\frac{max(d_i - p_{ijm} - t, 0)}{K\bar{p}}$
ATC
①
②

①
```
Max(Max(Add(Div(Sub(AT,DD),Div(Sub(Add(PT,PT),
Max(Max(Div(W,PT),Add(0.8039775361822307,RT)),S
ub(Sub(Mul(DD,RT),Max(W,DD)),Div(DD,W)))),Max(
Max(Add(AT,W),PT),Div(W,PT)))),Max(Max(DD,OP),
Sub(PT,AT))),Mul(Div(DD,W),Max(Add(Add(Div(DD,
0.8039775361822307),Sub(OP,W)),Max(Div(W,PT),Su
b(PT,AT))),Mul(Max(Div(DD,W),Mul(RT,AT)),Mul(Su
b(AT,DD),Max(DD,OP)))))),Max(Max(Max(Mul(W,PT)
,Add(DD,Add(RPT,AT))),Sub(Sub(OP,0.803977536182
2307),Sub(0.8039775361822307,RPT))),Max(W, OP)))
```

②
```
Sub(Sub(W,RT),Max(RPT,W))),Div(Mul(Div(DD,0.148
94625742195577),Max(Sub(Sub(DD,PT),Max(RPT,W
)),0.1489462),Div(Add(PT,Div(Add(0.14894625742195
577, RT), Max(DD, PT))), ...
```

Fig. 7. Similarities between GP-generated rules and human-made rules.

As shown in Fig. 5, in both loose and tight due date settings, a big fluctuation in the solution quality is observed in GP-based HH throughout the whole evolutionary process, while a better convergence is realized in 2-stage HH. The reason lies in that, in GP-based HH, new rules are added to every generation, resulting in a fluctuating performance in the evolutionary process. In 2-stage HH, good heuristic rules are selected by GA, which ensures a better convergence.

5) Comparison With 2-Stage HH: To evaluate the effectiveness and efficiency of the HEH approach, a comparison experiment between 2-stage HH and HEH is conducted. As shown in Table XIII, the gap which shows the percentage deviation between 2-stage HH and HEH is defined in (16)

$$Gap_{2-stage} = \frac{score_{2-stage\ HH} - score_{HEH}}{score_{HEH}} \tag{16}$$

where $score_{2-stage\ HH}$ and $score_{HEH}$ represent the average objective function values obtained by 2-stage HH and that obtained by HEH, respectively.

In loose due date settings, for the best cases, the gap between 2-stage HH and HEH ranges from 5.2% to 77.1%, and the HEH approach outperforms the 2-stage HH approach in all test problems with an average gap of 15.0%. While for the worst cases, the gap between 2-stage HH and HEH ranges from 0.4% to 119.0%, and HEH outperforms 2-stage HH in all test problems with an average gap of 16.6%, as shown in Table XIII (a).

Similarly, in tight due date settings, as shown in Table XIII (b), HEH outperforms 2-stage HH in all test problems with an average gap of 7.6% for the best cases. While for the worst cases, HEH also outperforms 2-stage HH in all test problems with an average gap of 8.5%.

TABLE XIII
COMPARISON BETWEEN 2-STAGE HH AND HEH

Problem scale	(a) loose due date						(b) tight due date					
	2-stage HH		HEH		*Gap*$_{2\text{-}stage}$ (%)		2-stage HH		HEH		*Gap*$_{2\text{-}stage}$ (%)	
	min	max	min	max	min	max	min	max	min	max	min	max
*p*5*m*6*c*3	0.00E+00	2.20E+02	0.00E+00	0.00E+00	-	-	1.16E+03	2.10E+03	1.16E+03	1.71E+03	0.0	22.7
*p*15*m*8*c*3	0.00E+00	4.59E+03	0.00E+00	2.10E+03	-	119.0	1.66E+04	4.60E+04	1.58E+04	4.21E+04	5.1	9.3
*p*20*m*11*c*3	0.00E+00	8.81E+03	0.00E+00	7.07E+03	-	24.7	1.73E+04	5.24E+04	1.65E+04	4.90E+04	4.7	6.9
*p*40*m*13*c*5	0.00E+00	3.59E+05	0.00E+00	2.30E+05	-	56.0	4.54E+05	1.07E+06	4.26E+05	9.69E+05	6.5	10.0
*p*50*m*15*c*5	5.52E+04	1.28E+06	4.28E+04	1.28E+06	28.9	0.4	1.11E+06	3.20E+06	1.05E+06	3.05E+06	5.1	5.0
*p*60*m*16*c*5	9.29E+04	1.47E+06	5.25E+04	1.39E+06	77.1	5.8	2.10E+06	4.63E+06	1.98E+06	4.10E+06	6.1	13.0
*p*70*m*20*c*7	1.91E+05	3.15E+06	1.58E+05	2.62E+06	20.3	20.3	4.56E+06	9.04E+06	4.30E+06	8.28E+06	6.1	9.1
*p*80*m*21*c*7	7.88E+06	1.53E+07	6.70E+06	1.44E+07	17.6	5.8	1.79E+07	2.65E+07	1.70E+07	2.55E+07	5.3	3.8
*p*90m21*c*7	9.88E+06	2.00E+07	8.22E+06	1.79E+07	20.2	11.8	2.40E+07	3.56E+07	2.15E+07	3.35E+07	11.6	6.0
*p*100*m*25*c*9	1.80E+07	3.05E+07	1.59E+07	2.84E+07	13.1	7.1	3.56E+07	4.94E+07	3.23E+07	4.60E+07	10.4	7.4
*p*120*m*30*c*9	6.10E+07	9.47E+07	5.66E+07	8.58E+07	7.8	10.4	9.54E+07	1.27E+08	9.01E+07	1.22E+08	5.9	4.1
*p*140*m*35*c*11	8.32E+07	1.25E+08	7.84E+07	1.17E+08	6.1	6.4	1.46E+08	1.90E+08	1.36E+08	1.80E+08	7.4	5.8
*p*160*m*40*c*11	1.47E+08	2.07E+08	1.37E+08	1.90E+08	7.5	8.8	2.37E+08	3.03E+08	2.17E+08	2.89E+08	8.8	5.0
*p*180*m*45*c*13	3.11E+08	4.43E+08	2.94E+08	4.24E+08	5.5	4.4	4.60E+08	5.72E+08	4.17E+08	5.41E+08	10.4	5.7
*p*200*m*50*c*15	4.17E+08	5.44E+08	3.93E+08	5.15E+08	6.2	5.7	6.00E+08	7.27E+08	5.32E+08	6.93E+08	12.7	4.9
*p*250*m*65*c*15	1.15E+09	1.41E+09	1.08E+09	1.34E+09	5.9	5.3	1.53E+09	1.78E+09	1.42E+09	1.72E+09	7.5	3.7
*p*300*m*75*c*15	2.04E+09	2.55E+09	1.92E+09	2.39E+09	6.3	6.4	2.61E+09	3.16E+09	2.37E+09	2.88E+09	10.1	9.8
*p*350*m*90*c*15	3.28E+09	3.89E+09	3.06E+09	3.73E+09	7.0	4.3	4.19E+09	4.87E+09	3.81E+09	4.57E+09	9.8	6.8
*p*400*m*100*c*15	7.52E+09	9.49E+09	7.15E+09	8.93E+09	5.2	6.3	9.18E+09	1.10E+10	8.46E+09	9.25E+09	8.5	18.8
*p*450*m*120*c*15	9.63E+09	1.17E+10	9.07E+09	1.11E+10	6.1	5.9	1.18E+10	1.38E+10	1.07E+10	1.22E+10	10.1	12.7
Average					15.0	16.6					7.6	8.5

Moreover, the evolutionary processes of HEH and 2-stage HH are also given in Fig. 5.

It is observed in Fig. 5 that in both loose and tight due date settings, 2-stage HH falls into local optima in the early stages of the evolutionary processes despite its ability in finding good solutions in the early generations. In contrast, HEH converges to better solutions in much later stages during the evolutionary processes, indicating HEH performs better in exploring promising regions in the search space. This is because of the introduction of the catastrophe operator and the local search algorithm in the heuristic search methodology. By the adoption of the catastrophe operator, the mutation probability is adaptively updated to avoid being trapped into the local optima.

Moreover, the hill climbing algorithm is used to perform local searches, which enables HEH to exploit more deeply for better solutions. Therefore, a better balance between exploration and exploitation is achieved by HEH.

In summary, better performance in both loose and tight due date settings indicates the effectiveness and stability of HEH. Moreover, it is observed from the above comparison that the advantage of HEH is more obvious in loose due date settings. The reason is the same as that illustrated in Section IV-C2. Now that the due dates are tight, it is always hard to complete the parts no matter what approaches are adopted.

Considering all of the four hyper-heuristic approaches in Fig. 5 in an integrated way, 2-stage HH outperforms GA-based HH and GP-based HH owing to the advantage of combining heuristic generation with heuristic selection, and HEH outperforms 2-stage HH by adopting the catastrophe operator to enhance the exploration ability and the local search procedure to enhance the exploitation ability. Therefore, the HEH approach is both effective and reliable in finding good solutions as compared with other hyper-heuristics.

6) Comparison With TS: To further evaluate the effectiveness and efficiency of HEH, a comparison experiment between HEH and TS is conducted.

Due to the dramatic difference between these two approaches, the experiment in this sub-section is designed according to the principle described in [29] that all approaches are allowed to run within the same time limit, with distinctions between the quality of solutions obtained. For the sake of equity, the running time for TS to find the optimal solutions are taken as the running time limit of HEH, and the quality of the solutions obtained within the same time limit is presented in Table XIV.

The gap which shows the percentage deviation between TS and HEH is defined in (17)

$$Gap_{\text{TS}} = \frac{score_{\text{TS}} - score_{HEH}}{score_{HEH}} \tag{17}$$

where $score_{\text{TS}}$ and $score_{HEH}$ represent the average objective function values obtained by TS and HEH, respectively.

As shown in Table XIV (a), in loose due date settings, the gap between HEH and TS ranges from 2.4% to 613.5%, and HEH outperforms TS with an average gap of 82.1%.

Similarly, in tight due date settings, as shown in Table XIV (b), HEH outperforms TS with an average gap of 14.7%.

TABLE XIV
COMPARISON BETWEEN HEH AND TS

Problem scale	(a) loose due date				(b) tight due date			
	HEH	TS	Time(s)	Gap_{TS}(%)	HEH	TS	Time(s)	Gap_{TS}(%)
*p*5*m*6*c*3	0.00E+00	0.00E+00	2.61E+03	-	1.16E+03	1.51E+03	2.61E+03	29.8
*p*15*m*8*c*3	0.00E+00	0.00E+00	1.07E+04	-	1.70E+04	2.50E+04	1.07E+04	47.0
*p*20*m*11*c*3	0.00E+00	0.00E+00	1.94E+04	-	1.77E+04	2.62E+04	1.94E+04	47.7
*p*40*m*13*c*5	0.00E+00	3.16E+04	8.03E+04	-	4.48E+05	5.40E+05	8.03E+04	20.6
*p*50*m*15*c*5	4.74E+04	2.49E+05	1.42E+05	426.0	1.12E+06	1.38E+06	1.42E+05	23.1
*p*60*m*16*c*5	6.65E+04	4.75E+05	1.97E+05	613.5	2.20E+06	2.58E+06	1.97E+05	17.0
*p*70*m*20*c*7	4.52E+05	1.12E+06	3.93E+05	146.6	5.07E+06	6.16E+06	3.93E+05	21.5
*p*80*m*21*c*7	7.79E+06	9.42E+06	5.47E+05	20.9	1.78E+07	2.01E+07	5.47E+05	12.9
*p*90m21*c*7	9.32E+06	1.26E+07	7.56E+05	34.7	2.41E+07	2.56E+07	7.56E+05	6.2
*p*100*m*25*c*9	1.87E+07	2.11E+07	8.81E+05	13.2	3.74E+07	3.98E+07	8.81E+05	6.6
*p*120*m*30*c*9	6.18E+07	6.63E+07	1.11E+06	7.3	9.41E+07	1.01E+08	1.11E+06	7.7
*p*140*m*35*c*11	8.43E+07	9.11E+07	1.42E+06	8.1	1.44E+08	1.52E+08	1.42E+06	5.5
*p*160*m*40*c*11	1.42E+08	1.56E+08	2.19E+06	9.5	2.35E+08	2.48E+08	2.19E+06	5.5
*p*180*m*45*c*13	3.03E+08	3.31E+08	2.41E+06	9.5	4.30E+08	4.66E+08	2.41E+06	8.4
*p*200*m*50*c*15	4.21E+08	4.50E+08	3.02E+06	7.0	5.83E+08	6.19E+08	3.02E+06	6.2
*p*250*m*65*c*15	1.15E+09	1.19E+09	4.91E+06	3.7	1.52E+09	1.57E+09	4.91E+06	3.3
*p*300*m*75*c*15	2.04E+09	2.08E+09	7.34E+06	2.4	2.47E+09	2.72E+09	7.34E+06	10.2
*p*350*m*90*c*15	3.28E+09	3.36E+09	9.93E+06	2.6	4.20E+09	4.32E+09	9.93E+06	3.0
*p*400*m*100*c*15	7.45E+09	7.86E+09	1.43E+07	5.5	8.85E+09	9.37E+09	1.43E+07	5.8
*p*450*m*120*c*15	9.57E+09	9.86E+09	2.31E+07	3.0	1.12E+10	1.19E+10	2.31E+07	5.4
Average				82.1				14.7

In summary, as compared with TS, HEH provides better scheduling solutions within the same running time, indicating the effectiveness and efficiency of scale problems in practice.

V. CONCLUSION

To minimize TWT, the problem of intercell scheduling with the constraint of transportation capacity is addressed in this paper, in which the coordination of part scheduling for machines and batch formation for vehicles is required. Due to the high complexity and large problem size, the HEH approach is proposed, in which GP is first used to generate heuristic rules, and then GA is applied to the extended rule set for the selection of suitable combinations of heuristic rules. Finally, the scheduling solutions are obtained through the selected rules. The comparison experiments are conducted by comparing HEH with the combinatorial rules, with other hyper-heuristics, and with TS meta-heuristic. Additionally, the structure of GP-generated rules is analyzed by the comparison with the most competing human-made rules for the weighted tardiness criterion. The computational results have shown a superiority of HEH in both computational efficiency and solution quality, and therefore, it is especially suitable for large scale problems.

The future work will focus on three aspects which are listed as follows.

Owing to the large computational time cost by GP in its evolution of rules, it is beneficial to simplify the rules in its evolutionary process. On one hand, appropriate terminals for GP are worth selecting. Therefore, a comprehensive study of potential attributes and their effects on the manufacturing environments is needed, which is done by counting the frequency of a certain attribute found in the best GP-generated rules. On the other hand, as the rules generated in the evolution are complex-looking and lengthy, it might be beneficial to simplify rules in each iteration.

Besides, due to the high correlation among the different sub-problems, it is not in accord with actual production if rules for different sub-problems evolve separately. Therefore, cooperative co-evolution [30] can be adopted where rules for different sub-problems evolve in different sub-populations concurrently. Each sub-population represents one rule in the combinatorial rules, e.g., a sequencing rule for a machine or a batch formation rule for a vehicle. For an individual in the sub-population, its performance is evaluated by combining the individual with a representative individual, i.e., the one with the best performance, from other sub-populations to form complete combinatorial rules.

Finally, to obtain a generated rule with better reliability, the effectiveness of rules should be evaluated by other objective criteria, such as the maximum completion time and the utilization rate of vehicles, etc.

REFERENCES

[1] Z. Liu, D. Li, L. Wang, and Y. Tian, "An inter-cell scheduling approach considering transportation capacity constraints," *Acta Autom. Sin.*, vol. 41, no. 5, pp. 885–898, May 2015.

[2] D. J. Johnson and U. Wemmerlöv, "Why does cell implementation stop? Factors influencing cell penetration in manufacturing plants," *Prod. Oper. Manag.*, vol. 13, no. 3, pp. 272–289, Aug. 2004.

[3] O. Garza and T. L. Smunt, "Countering the negative impact of intercell flow in cellular manufacturing," *J. Oper. Manag.*, vol. 10, no. 1, pp. 92–118, Feb. 1991.

[4] M. Solimanpur, P. Vrat, and R. Shankar, "A heuristic to minimize makespan of cell scheduling problem," *Int. J. Prod. Econ.*, vol. 88, no. 3, pp. 231–241, Apr. 2004.
[5] R. Logendran, L. Mai, and D. Talkington, "Combined heuristics for bi-level group scheduling problems," *Int. J. Prod. Econ.*, vol. 38, no. 1, pp. 133–145, Mar. 1995.
[6] R. Logendran and N. Nudtasomboon, "Minimizing the makespan of a group scheduling problem: A new heuristic," *Int. J. Prod. Econ.*, vol. 22, no. 3, pp. 217–230, Dec. 1991.
[7] V. A. Petrov, *Flow Line Group Production Planning*. London, U.K.: Business Publications, 1968.
[8] R. Tavakkoli-Moghaddam, N. Javadian, A. Khorrami, and Y. Gholipour-Kanani, "Design of a scatter search method for a novel multi-criteria group scheduling problem in a cellular manufacturing system," *Expert Syst. Appl.*, vol. 37, no. 3, pp. 2661–2669, Mar. 2010.
[9] A. Golmohammadi and R. Ghodsi, "Applying an integer electromagnetism-like algorithm to solve the cellular manufacturing scheduling problem with an integrated approach," in *Proc. Int. Conf. Comput. & Ind. Eng.*, Troyes, France, Jul. 2009, pp. 34–39.
[10] J. Tang, X. Wang, I. Kaku, and K. Yung, "Optimization of parts scheduling in multiple cells considering intercell move using scatter search approach," *J. Intell. Manuf.*, vol. 21, no. 4, pp. 525–537, Aug. 2010.
[11] A. Elmi, M. Solimanpur, S. Topaloglu, and A. Elmi, "A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts," *Comput. Ind. Eng.*, vol. 61, no. 1, pp. 171–178, Aug. 2011.
[12] D. Li, X. Meng, M. Li, and Y. Tian, "An ACO-based intercell scheduling approach for job shop cells with multiple single processing machines and one batch processing machine," *J. Intell. Manuf.*, Jan. 2014, DOI: 10.1007/s10845-013-0859-2.
[13] H. Sarper and M. C. Henry, "Combinatorial evaluation of six dispatching rules in a dynamic two-machine flow shop," *Omega*, vol. 24, no. 1, pp. 73–81, Feb. 1996.
[14] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. O. Zcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *J. Oper. Res. Soc.*, vol. 64, no. 12, pp. 1695–1724, Dec. 2013.
[15] R. H. Storer, S. D. Wu, and R. Vaccari, "New search spaces for sequencing problems with application to job shop scheduling," *Manage. Sci.*, vol. 38, no. 10, pp. 1495–1509, Oct. 1992.
[16] H. Aytug, S. Bhattacharyya, G. J. Koehler, and J. L. Snowdon, "A review of machine learning in scheduling," *IEEE Trans. Eng. Manag.*, vol. 41, no. 2, pp. 165–171, May 1994.
[17] J. W. Herrmann, C. Y. Lee, and J. Hinchman, "Global job shop scheduling with a genetic algorithm," *Prod. Oper. Manag.*, vol. 4, no. 1, pp. 30–45, Mar. 1995.
[18] J. A. V. Rodríguez, S. Petrovic, and A. Salhi, "An investigation of hyper-heuristic search spaces," in *Proc. IEEE Congr. Evolutionary Computation*, Singapore, Sep. 2007, pp. 3776–3783.
[19] C. D. Geiger, R. Uzsoy, and H. Aytug, "Rapid modeling and discovery of priority dispatching rules: an automated learning approach," *J. Scheduling*, vol. 9, no. 1, pp. 7–39, Feb. 2006.
[20] J. A. V. Rodríguez and S. Petrovic, "A new dispatching rule based genetic algorithm for the multi-objective job shop problem," *J. Heuristics*, vol. 16, no. 6, pp. 771–793, Dec. 2010.
[21] G. Ochoa, R. Qu, and E. K. Burke, "Analyzing the landscape of a graph based hyper-heuristic for timetabling problems," in *Proc. 11th Annu. Conf. Genetic and Evol. Comput. Conf.*, Montreal, Canada, Jul. 2009, pp. 341–348.
[22] C. D. Geiger and R. Uzsoy, "Learning effective dispatching rules for batch processor scheduling," *Int. J. Prod. Res.*, vol. 46, no. 6, pp. 1431–1454, Mar. 2008.
[23] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 621–639, Oct. 2013.
[24] C. W. Pickardt, T. Hildebrandt, J. U. R. Branke, J. Heger, and B. Scholz-Reiter, "Evolutionary generation of dispatching rule sets for complex dynamic scheduling problems," *Int. J. Prod. Econ.*, vol. 145, no. 1, pp. 67–77, Sep. 2013.
[25] S. Luke, 2013, *Essentials of Metaheuristics*, 2nd ed. [Online]. Available: http://cs.gmu.edu/_sean/book/metaheuristics/
[26] A. P. J. Vepsalainen and T. E. Morton, "Priority rules for job shops with weighted tardiness costs," *Manage. Sci.*, vol. 33, no. 8, pp. 1035–1047, Aug. 1987.
[27] E. Nowicki and C. Smutnicki, "A fast taboo search algorithm for the job shop problem," *Manage. Sci.*, vol. 42, no. 6, pp. 797–813, June 1996.
[28] J.-P. Watson, A. E. Howe, and L. D. Whitley, "Deconstructing Nowicki and Smutnicki's i-TSAB tabu search algorithm for the job shop scheduling problem," *Comput. Oper. Res.*, vol. 33, no. 9, pp. 2623–2644, Sep. 2006.
[29] R. L. Rardin and R. Uzsoy, "Experimental evaluation of heuristic optimization algorithms: a tutorial," *J. Heuristics*, vol. 7, no. 3, pp. 261–304, May 2001.
[30] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 193–208, Apr. 2014.

Dongni Li received the B.S. and Ph.D. degrees in computer science from Northeastern University, Shenyang, China, in 2000 and 2005, respectively.

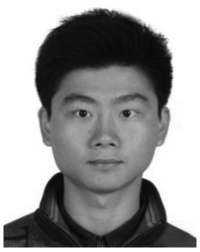
Since 2008, she has been an Associate Professor with the Beijing Key Lab of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, China, where she was a lecturer from 2005 to 2007. From 2010 to 2011, she was a postdoctoral researcher with the School of Operations Research and Information Engineering, Cornell University, Ithaca, NY, USA. Her research interests include intelligent decision-making approaches and their applications to the manufacturing industry.

Rongxin Zhan received the B.S. degree in computer science from Zhengzhou University, Henan, China, in 2014. He is currently pursuing the M.E. degree with a major in biology at the Beijing Institute of Technology, Beijing, China.

His research interests include swarm intelligence and production scheduling.

Dan Zheng received the B.S. degree in computer science from Beijing Foreign Studies University, Beijing, China, in 2013. She is currently pursuing the M.S. degree at the School of Computer Science, Beijing Institute of Technology, Beijing, China.

Her research interests include optimization approaches and production scheduling.



Miao Li received the B.S. degree in computer science from Beijing Institute of Technology, Beijing, China, in 2013, where he is currently pursuing the M.S. degree in the School of Computer Science.

His research interests include heuristic-based optimization approaches.

Ikou Kaku is a Professor with the Department of Environmental and Information Studies, Tokyo City University, Tokyo, Japan. His research interests include industrial engineering and management.