Human Resource Allocation for Parallel Assembly Processes with Temporal and Spatial Constraints

1st Xiaoqing Liu
College of Information Science and Engineering
Northeastern University
Shenyang, 110819, China
liuxiaoqing@stumail.neu.edu.cn

2nd Weihua Deng
DFH Satellite Co.Ltd.
Beijing, 100094, China
dwh7875@163.com

3th Shengchao Li
College of Information Science and Engineering
Northeastern University
Shenyang, 110819, China
lishengchao@stumail.neu.edu.cn

4th Shixin Liu
College of Information Science and Engineering, Northeastern University
Shenyang, 110819, China
School of Control Engineering, Northeastern University at Qinhuangdao
Qinhuangdao, 066004, China
sxliu@mail.neu.edu.cn

2024 International Conference on Networking, Sensing and Control (ICNSC) | 979-8-3503-6522-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICNSC62968.2024.10760067

***Abstract*—Human resource allocation in parallel assembly processes (PAPs) is an important optimization problem. It can be regarded as a resource-constrained project scheduling problem with specific constraints. In this paper, a mixed-integer programming (MIP) model is established for it to minimize the maximum completion time (i.e., makespan) by comprehensively analyzing the characteristics of a PAP, including temporal constraints: worker periodic unavailability and the tail time of operations; and spatial constraints: workstation and human resource constraints. To address this problem, a heuristic rule for prioritizing idle workers that considers the precedence relationships of operations is designed as a baseline method, and an iterative greedy algorithm is further proposed to provide high-performance solutions. These methods aim to shorten the cycle and costs of a PAP. The effectiveness of the proposed algorithms is validated through extensive experiments.**

***Index Terms*—Human resource allocation, tail time, mixed integer programming, heuristic algorithm, iterative greedy algorithm.**

I. Introduction

Parallel assembly processes (PAPs) are common in manufacturing areas such as automobiles, aircraft and satellites [1]–[3]. Human resources play a crucial role in these processes [4]. Optimizing a human resource allocation problem (HRAP) in PAPs is meaningful to shorten the processing time. A PAP is usually consisted of many operations completed by limited workers in constrained workstations. These operations have a priority relationship, and some of them may have tail time. The number of workers required to deal with different operations is different, and the working hours required for different workers to deal with the same operation are also different. These operations can be interrupted and cannot be preempted, that is, a worker is allowed to rest during the current operation, but is not allowed to process other operations. HRAP determines the sequence of the operations to be processed by each worker, based on the periodic availability of workers, interruption, non-preemption assembly, tail time of operations, the workstation and human resource constraints to minimize makespan.

HRAP is an extension of a resource-constrained project scheduling problem (RCPSP), which includes additional constraints such as tail time, available time of resources (workers), and workstation constraints. RCPSP is an important class of scheduling problems whose objective is to minimize cost or makespan depending on activity precedence relations and scarce resources [5]. A project in RCPSP contains a set of activities that need to be scheduled, with an order among these activities, and each activity has a standard duration. Its resources can be of various types, such as manpower, equipment, and materials [6]. Each resource is subject to a fixed quantity limit, and its availability varies across different time periods [7]. Additionally, each activity specifies unique resource requirements. RCPSP is an NP-hard problem [8]. Its solution methods can be mainly divided into exact algorithms, heuristics and meta-heuristics. Among them, the exact algorithms mainly include branch and bound methods [9]; heuristics and meta-heuristics include priority rule-based heuristics [10], simulated annealing [11]–[13], adaptive large neighborhood search [14]–[16], and local search [17], [18].

HRAP considers an operation processing with tail time. In PAPs, some operations require a tail operation after they are completed by workers. A tail operation can be conducted during workers' rest periods. During tail operation, workers can process other operations. However, any subsequent operation that has a priority relationship with the current operation cannot start until the tail operation of the current operation is finished. That is, the tail operation affects the scheduling and arrangement of operations. Some existing work also considers tail time. A single machine scheduling problem with release and tail time whose objective is to minimize makespan is studied in [19]. This problem is strongly NP-hard, but it is polynomially solvable if all jobs have equal processing time

P. An algorithm with time complexity of $O(n^2 \log n \log P)$ is proposed, where the processing time of a job is limited to P or $2P$ and the number of jobs is n. A single machine scheduling problem with all jobs having positive tail time is studied in [20], and the objective is also to minimize makespan. The authors propose a polynomial approximation algorithm with a worst-case performance ratio of $3/2$, and they prove that the bound is tight. A dynamic programming algorithm is also proposed in [20], and it is proven that the problem has a fully polynomial-time approximation scheme.

This paper considers the scenarios where workers are periodically unavailable and operations are interruptible but non-preemptive. Workers are periodically unavailable due to constraints on available time periods. If the operation is not completed in a working period, it will continue to be processed in the next working period. That is, the operation can be interrupted; workers are not allowed to process other operations when the current operation starts processing but is not completed. In [21], the scheduling problem of two parallel machines is discussed. In that problem, one machine is unavailable; the jobs are not preempted; and the maximum completion time is taken as the objective. A mathematical programming model is proposed, and the longest processing time first (LPT) algorithm and list scheduling (LS) algorithm are analyzed. It is also shown that an LPT algorithm outperforms an LS algorithm in 96 combinations of the two main parameters.

An iterative greedy algorithm (IGA) [22] is used in this paper. It is a simple but powerful heuristic algorithm, which is widely used to solve a variety of complex optimization problems. It solves the problem by iteratively exploring solutions in a given neighborhood and has some perturbation strategies to escape from local optima. The basic idea of IGA is to repeatedly perform the two steps of “destruction" and “construction" to generate a series of solutions, so as to gradually approach the global optimal solution. This process can effectively avoid falling into local optimum and improve the quality of solutions. In recent years, IGA has been widely used to solve scheduling problems [23], [24].

This work tackles an HRAP arising from a PAP with temporal and spatial characteristics, which has the following contributions: 1) An HRAP from a PAP is proposed for the first time where periodic unavailability of workers, interruption, non-preemption assembly, tail time of operations, workstation and human resource constraints are considered; 2) A mixed-integer programming (MIP) model is established for the problem to minimize makespan, which is then linearized; 3) A heuristic rule is developed to assign idle workers first considering the priority relationship of operations. IGA with this heuristic as the initial solution is then proposed that can well solve the problem.

This paper is structured as follows. Section II describes the problem and establishes a mathematical model. Section III designs heuristic rules and IGA. Section IV introduces numerical experiments, and Section V concludes the paper.

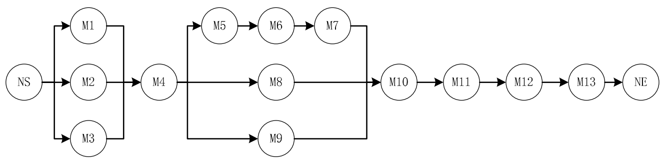

Fig. 1: Precedence Relationship of Operations

II. Problem Description and Mathematical Model

Given a set of jobs J and a set of workers I, each job j $(j \in J)$ contains the same set of operations M. Precedence relationships of operations are shown in Figure 1. NS and NE represent the virtual start and end nodes, respectively. M1-M13 indicate thirteen operations. The tuple set of directly precedence operations is denoted by A. Workers can handle any operation, but the time required for each operation varies due to varying proficiency levels in different operations. Let p_{mi} denote the time required for worker i $(i \in I)$ to process operation m $(m \in M)$. It takes n_m workers to process operation m simultaneously, and it takes $1/\sum_{i\in I'} \frac{1}{p_{mi}}$ for the set of workers I' to process this operation. After operation m is completed by workers, a tail operation requiring g_m time is needed. During a tail operation, workers can handle other operations; however, any subsequent operation with a precedence relationship to m can only start after the tail operation of m is completed. This problem considers working periods, meaning that workers must process operations during working hours, while the tail operations of operations can be carried out during non-working hours. Additionally, the number of jobs being processed simultaneously cannot exceed $|S|$. That is, processing $|S| + 1$ or more jobs concurrently is not allowed. The objective of HRAP is to minimize the maximum completion time of all operations, i.e., makespan. The parameters mentioned are further explained in Table I.

TABLE I: Parameter Description

Para.	Description
J	Set of jobs, $J = \{1, 2, ..., \lvert J\rvert\}$
M	Set of operations, $M = \{1, 2, ..., \lvert M\rvert\}$
I	Set of workers, $I = \{1, 2, ..., \lvert I\rvert\}$
S	Set of workstations, $S = \{1, 2, ..., \lvert S\rvert\}$
n_m	The number of workers required to process operation m
g_m	The tail time required for operation m
p_{mi}	The time required for worker i to process operation m
A	The set of tuples directly leading, $(m, m') \in A$

Example 1: Consider an instance of HRAP. Given $J = \{1\}$, $I = \{1, 2\}$, $M = \{1, 2, 3, 4\}$, and $S = \{1, 2\}$. The processing times required for each operation by workers P1 and P2 are respectively: $16, 13, 17, 11$; $11, 10, 11, 20$. The corresponding tail time of each operation is $44, 44, 40, 0$.

Figure 2 is the Gantt chart from the perspective of workers of Example 1. The horizontal axis is in hours, starting at time 0, and the vertical axis represents workers. Among them, the mint green part represents the daily working hours of workers, with each worker working 8 hours per day. The

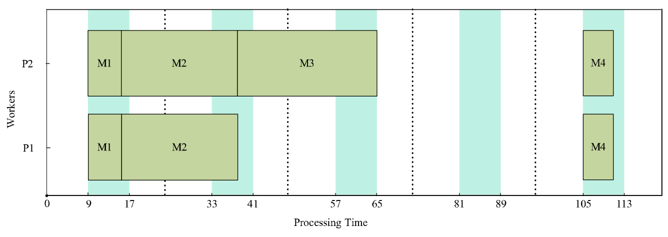

Fig. 2: Gantt Chart Example from Workers' Perspective

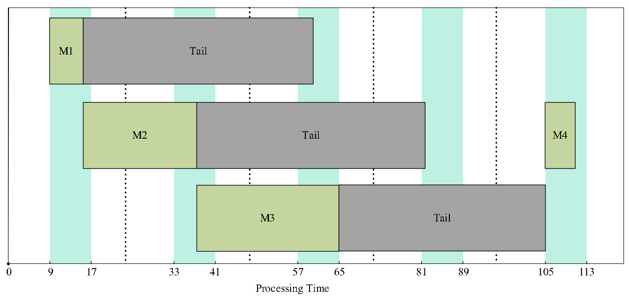

Fig. 3: Gantt Chart Example from Operations' Perspective

olive green part indicates the processing status of operations. M1-M4 refer to the four operations of the current job. The blocks in the rows of workers P1 and P2 indicate that these operations are assigned to the respective worker for processing. The processing time of each operation is represented by the width of its block. In Figure 2, M1 is assigned to workers P1 and P2 at the same time, which consumes about 6.52 hours. M4 can only start after the first three operations are completed (including the tail operation). Figure 3 is the Gantt chart from the operations' perspective of Example 1. The vertical axis has no practical meaning. The olive green part represents the processing status of operations. Immediately following is the gray part indicating the tail for the current operation. If there is no gray section, it means the current operation does not require tail operation.

In order to describe this problem better, we additionally consider the following parameters and establish a mathematical model.

- A set of location k ($k \in K$). $K := \{1, 2, ..., |J| \times |M|\}$.
- A set of time periods q ($q \in Q$). $Q = \{1, 2, ..., |Q|\}$.
- The q-th working period. It is denoted by $[\underline{t}_q, \overline{t}_q]$, with the endpoints of an interval representing the start and end times of a workday, respectively.
- Upper bound on the maximum allowable rest time for workers during which the operation m of job j is completed. Denoted by $\overline{R}_{jm}$, it is computed as follows:

$$\overline{R}_{jm} := (\lceil \frac{\max_{i \in I}\{p_{mi}\}}{\min_{q \in Q}\{\overline{t}_q - \underline{t}_q\}} \rceil + 1)(24 - \min_{q \in Q}\{\overline{t}_q - \underline{t}_q\}). \tag{1}$$

- Upper bound of processing time on operation m for job j. Denoted by $\overline{P}_{jm}$, it is computed as follows:

$$\overline{P}_{jm} := \max_{i \in I}\{p_{mi}\}. \tag{2}$$

To model this problem, this paper considers the parameters listed in Table I, the additional parameters introduced in this section, and the decision variables shown in Table II. An MIP model is established and the nonlinear components within it are linearized.

TABLE II: Decision Variables

Variable	Range	Description
x_{jmik}	$\{0,1\}$	Whether the operation m of job j is assigned to the location k of worker i
w_{js}	$\{0,1\}$	Whether the job j is assigned to the workstation s
$\underline{z}_{jmq}$	$\{0,1\}$	Whether the operation m of job j starts in the q-th working period
$\overline{z}_{jmq}$	$\{0,1\}$	Whether the operation m of job j finishes in the q-th working period (excluding tail time)
ST_{jm}	$\mathbb{R}_+$	The start time of operation m of job j
CT_{jm}	$\mathbb{R}_+$	The completion time of operation m of job j
P_{jm}	$\mathbb{R}_+$	The processing time of operation m of job j
ST^I_{ik}	$\mathbb{R}_+$	The start time of the location k of worker i
CT^I_{ik}	$\mathbb{R}_+$	The end time of the position k of worker i
R_{jm}	$\mathbb{R}_+$	The rest time for workers during the processing of operation m of job j
C_{max}	$\mathbb{R}_+$	Makespan

$$\min \quad C_{max} \tag{3}$$

$$s.t. \quad C_{max} \geq CT_{jm}, \quad \forall j \in J, m \in M \tag{4}$$

$$\sum_{j \in J} \sum_{m \in M} x_{jmik} \leq 1, \quad \forall i \in I, k \in K \tag{5}$$

$$\sum_{k \in K} x_{jmik} \leq 1, \quad \forall j \in J, m \in M, i \in I \tag{6}$$

$$\sum_{i \in I} \sum_{k \in K} x_{jmik} = n_m, \quad \forall j \in J, m \in M \tag{7}$$

$$\sum_{q \in Q} \underline{z}_{jmq} = 1, \quad \forall j \in J, m \in M \tag{8}$$

$$\sum_{q \in Q} \overline{z}_{jmq} = 1, \quad \forall j \in J, m \in M \tag{9}$$

$$\sum_{s \in S} w_{js} = 1, \quad \forall j \in J \tag{10}$$

$$CT_{jm} = ST_{jm} + P_{jm} + g_m + R_{jm}, \quad \forall j \in J, m \in M \tag{11}$$

$$ST_{jm'} \geq CT_{jm}, \quad \forall j \in J, (m, m') \in A \tag{12}$$

$$P_{jm} = 1/\sum_{i \in I} \sum_{k \in K} \frac{1}{p_{mi}} x_{jmik}, \quad \forall j \in J, m \in M \tag{13}$$

$$ST_{jm} \geq ST^I_{ik} + big\text{-}M * (x_{jmik} - 1), \quad \forall j \in J, m \in M, i \in I, k \in K \tag{14}$$

$$ST_{jm} \leq ST^I_{ik} + big\text{-}M * (1 - x_{jmik}), \quad \forall j \in J, m \in M, i \in I, k \in K \tag{15}$$

$$ST_{jm} \leq \overline{t}_q + big\text{-}M * (1 - \underline{z}_{jmq}), \quad \forall j \in J, m \in M, q \in Q \tag{16}$$

$$ST_{jm} \geq \underline{t}_q + big\text{-}M * (\underline{z}_{jmq} - 1),$$
$$\forall j \in J, m \in M, q \in Q \tag{17}$$
$$CT_{jm} - g_m \leq \overline{t}_q + big\text{-}M * (1 - \overline{z}_{jmq}),$$
$$\forall j \in J, m \in M, q \in Q \tag{18}$$
$$CT_{jm} - g_m \geq \underline{t}_q + big\text{-}M * (\overline{z}_{jmq} - 1),$$
$$\forall j \in J, m \in M, q \in Q \tag{19}$$
$$CT^I_{ik} = ST^I_{ik} + \sum_{j \in J} \sum_{m \in M} P_{jm} x_{jmik}$$
$$+ \sum_{j \in J} \sum_{m \in M} R_{jm} x_{jmik}, \quad \forall i \in I, k \in K \tag{20}$$
$$ST^I_{i,k+1} \geq CT^I_{ik}, \quad \forall i \in I, k \in K \backslash |K| \tag{21}$$
$$ST_{j'm'} - CT_{jm} \geq big\text{-}M * (w_{js} + w_{j's} - 2),$$
$$\forall j, j' \in J, j < j', s \in S, m, m' \in M \tag{22}$$
$$R_{jm} \geq \sum_{q=q_1}^{q_2-1} (\underline{t}_{q+1} - \overline{t}_q) + big\text{-}M * (\underline{z}_{jmq_1} + \overline{z}_{jmq_2} - 2),$$
$$\forall j \in J, m \in M, q_1, q_2 \in Q, q_1 \leq q_2 \tag{23}$$
$$x_{jmik}, \underline{z}_{jmq}, \overline{z}_{jmq} \in \{0, 1\},$$
$$\forall j \in J, m \in M, i \in I, k \in K, q \in Q \tag{24}$$
$$R_{jm} \in [0, \overline{R}_{jm}], \quad \forall j \in J, m \in M \tag{25}$$
$$P_{jm} \in [0, \overline{P}_{jm}], \quad \forall j \in J, m \in M \tag{26}$$
$$ST_j, CT_j, ST^I_{ik}, CT^I_{ik}, C_{max} \geq 0$$
$$\forall j \in J, m \in M, i \in I, k \in K \tag{27}$$

Objective function (3) denotes minimizing makespan. Constraint (4) states that makespan is not less than the completion time of any operation. Constraint (5) states that one position of a worker can be assigned to at most one operation of one job. Constraint (6) states that any operation can be assigned to a worker at most once. Constraint (7) represents the limit of workers required for one operation of one job. Constraint (8) represents the assigned limit on the start processing time of one operation for one job. Constraint (9) represents the assigned limit on the completion time of one operation for one job. Constraint (10) represents the limit of job assignment to a workstation. Constraint (11) represents the relationship between the completion time of an operation and the start time, processing time, tail time, and worker rest time. Constraint (12) represents the relationship between the start and end times of two operations in any immediate leading operation tuple. Constraint (13) represents the relationship between the processing time of an operation and the processing time of workers assigned to the operation. Constraints (14) and (15) denote the relationship between the start time of an operation and the start time of workers' position assigned to the current operation. Constraints (16)-(19) represent the relationship among the start time, end time, tail time of the operation and the available period of workers. Constraint (20) represents the relationship between the operation completion time (excluding tail time) at a location of a worker and its start time, processing time, and rest time. Constraint (21) represents the relationship between the operation start time at one location of a worker and the completion time at its previous location. Constraint (22) represents that jobs assigned to a workstation must be processed sequentially. That is, the next job cannot be processed until all operations of the previous job have been completed. Constraint (23) represents the relationship between the rest time of the worker who completes the operation and the time period assigned by the operation. Constraints (24)-(27) define the range of the decision variables.

III. Heuristic Algorithms

In this paper, a heuristic of priority assignment rules for idle workers considering the priority relationship between operations is designed. Subsequently, the heuristic solution is utilized as the initial solution for IGA.

A. Initial Solution

The initial solution is obtained through the heuristic that considers prioritizing idle workers based on the precedence relationships of operations. By maintaining a list of operations where the number of jobs does not exceed the number of workstations, one operation is assigned at one time. When the list is empty, new pending job operations are added. The algorithm is shown in Algorithm 1.

The input of the algorithm is a set of scheduled jobs J, a set of workers I, and daily working hours $[\underline{t}, \overline{t}]$. The output is the set of assignments $\{x_{jmi}\}$, start time $\{ST_{jm}\}$, and end time $\{CT_{jm}\}$ for all operations of all jobs. Before the algorithm starts, the following parameters are defined. Daily working time of workers $\Delta t := \overline{t} - \underline{t}$; processing time of operation m for job j, $P_{jm} := 1/\sum_{i \in I'} \frac{1}{p_{mi}}$; the intermediate variable $t' := \Delta t - (ST_{jm}\%24 - \underline{t})$, which represents the worker-hours consumed for job j's operation m on the day it starts processing.

The first line of Algorithm 1 initializes all assignments and empties the list L of pending operations. Next, iterate continuously when either the list of pending operations or the list of pending jobs is not empty, retrieving assignments, start times, and end times for job j's operation m. The purpose of lines 3 to 5 of the algorithm is to sequentially select elements from the list of pending jobs into the list of pending operations and to sort the elements in the list of pending operations. The purpose of lines 6 to 10 of the algorithm is to maintain the list of pending operations and to obtain the processing status of jobs and operations. Line 11 assigns the set of workers I' to the operation m of job j. The purpose of lines 12 to 20 of the algorithm is to obtain the start time of job j's operation m; and line 21 is to retrieve the completion time of job j's operation m. The final line returns the algorithm's result.

B. Iterative Greedy Algorithm

The input d represents the number of elements to be removed in IGA, and other inputs of IGA are consistent with those of Algorithm 1. The output is historical best solution. The pending operation list is maintained, as detailed in Algorithm 2. The initial solution is then obtained by

Algorithm 1: Idle Worker Assignment Rule

```
   Input: $J$, $I$, $[\underline{t}, \overline{t}]$
   Output: $\{x_{jmi}\}$, $\{ST_{jm}\}$ and $\{CT_{jm}\}$
1  Initialization. $x_{jmi} \leftarrow 0, \forall j \in J, m \in M, i \in I$. A list of
     pending operations $L \leftarrow \emptyset$.
2  repeat
3      if $L = \emptyset$ then
4          $L \leftarrow$The operations of the first $\min\{|J|, n\}$ jobs;
             Delete the first $\min\{|J|, n\}$ operations from $J$;
5          Reorder $L$ according to ascending order of operation
             index first, followed by ascending order of job
             index;
6      Select the first operation $m$ from $L$ and remove it from
         $L$, noting that $m$ is an operation of $j$;
7      Obtain the set $T_1$ of completion times (including tail
         time) for the immediate predecessor operations of job
         $j$'s operation $m$;
8      Obtain the set $I'$ of the earliest available $n_m$ workers;
         Obtain the set $T_2$ of the earliest available times for the
         workers in $I'$;
9      Let $n_1$ denote the number of completed jobs, and $n_2$ the
         number of jobs that have started processing but are not
         yet completed;
10     Obtain the set $T_3$ of completion times (including tail
         time) for jobs where all operations have been
         completed;
11     $x_{jmi} \leftarrow 1, \forall i \in I'$;
12     if $n - n_2 > n_1$ then  $ST_{jm} \leftarrow \max_{t \in T_1 \cup T_2} t$ ;
13     else if $n - n_2 = 0$ then
14         if  $n_1 = 0$ then
15             $ST_{jm} \leftarrow \max_{t \in T_1 \cup T_2} t$;
16         else
17             $ST_{jm} \leftarrow \max_{t \in T_1 \cup T_2 \cup T_3} t$;
18     else
19         Retrieve the $(n - n_2)$-th largest value from the set
             $T_3$;
20         $ST_{jm} \leftarrow \max\{t', \max_{t \in T_1 \cup T_2} t\}$;
21     $CT_{jm} \leftarrow ST_{jm} + P_{jm} + \lceil (P_{jm} - t')/\Delta t \rceil \times (24 - \Delta t)$;
22 until $L = \emptyset$ and $J = \emptyset$;
23 return $\{x_{jmi}\}$, $\{ST_{jm}\}$, $\{CT_{jm}\}$.
```

invoking Algorithm 1 and is inserted into L in a specific order. Subsequently, the iterative process of destruction and construction begins. After reaching the maximum number of iterations, the algorithm terminates and returns the incumbent solution.

Algorithm 2: Iterated Greedy Algorithm

```
   Input: $J$, $I$, $[\underline{t}, \overline{t}]$, $d$
   Output: historical best solution $x^{best}$
1  Initialization. An ordered list of pending operations $L \leftarrow \emptyset$.
2  Call Algorithm 1 to obtain initial solution $x^{ini}$;
3  Update $x^{best}$;
4  Sort the operations in $x^{ini}$ in ascending order of their start
     times, and place the results into $L$;
5  repeat
6      Randomly select $d$ operations from $L$ and remove them
         from $L$;
7      Randomly select $d$ positions from $L$ to insert $d$
         operations, ensuring that these positions satisfy the
         precedence relationships of operations and workstation
         limit;
8      Assign operations from $L$ sequentially according to lines
         6 to 21 of Algorithm 1, and calculate makespan;
9      If the current solution outperforms $x^{best}$, then update
         $x^{best}$;
10 until reach the maximum number of iterations;
11 return $x^{best}$.
```

IV. Experiments

A large number of cases are generated to validate the effectiveness of the algorithms proposed in this paper. Consider $|J| \in \{5, 10, 15\}$, $|M| = 13$, $|I| \in \{2, 4, 6\}$, $|S| \in \{2, 3\}$. The processing time required by workers for operations follows a uniform distribution of integers: $p_{mi} \in U[10, 20]$; the number of workers n_m required to process operation m is set to 2; tail times of some operations are $g_1 = 44, g_2 = 44, g_3 = 44, g_{10} = 24$, $g_{11} = 24$, and the remaining tail time of operations is set to zero. A total of 18 parameter combinations are selected to ensure the reliability of the experiment. We randomly generate 10 times for the same $|J|$, $|I|$ and $|S|$ cases, and record the average results for ten groups of similar cases. A total of 180 cases are considered. The maximum iteration number of IGA is set to 1000. In this paper, the heuristic algorithm of priority assignment rules for idle workers considering the priority relationship of operations and IGA are implemented by C++ language. The simulation environment is AMD Ryzen 7 8845H processor, CPU frequency 3.8Hz, 32GB memory, 8 cores and 16 logical processors.

The solution results are shown in Table III. The columns of Heuristic and IGA represent their objective function values, respectively. Ratio is the optimization ratio which calculated by (Heuristic − IGA) / Heuristic. IGA(3) and Ratio(3) represent the number of elements removed in IGA's destruction phrase is 3. IGA(6) and Ratio(6) mean 6 elements are removed in IGA's destruction phase. It can be seen that when solving HRAP, IGA with the heuristic of priority assignment rules of idle workers considering the priority relationship of operations as the initial solution can achieve significant improvements. The average improvement rate of 3 removed elements is 4.74% and the 6 removed elements is 6.02%. IGA has more obvious improvement than the initial solution provided by the heuristic, and removing 6 elements in IGA's destruction phase outperforms removing 3 elements.

V. Conclusion

In this paper, an MIP model is proposed for HRAP in PAP, which is characterized by temporal and spatial constraints including the periodic unavailability of workers, interruption, non-preemption assembly, tail time of operations, workstation and human resource constraints. Furthermore, a heuristic for the priority assignment rules of idle workers considering the priority relationship of operations and IGA are designed, which realize the cycle optimization of PAP. A large number of simulation data are generated to verify the effectiveness of algorithms showing that the heuristic algorithm and IGA

TABLE III: The Solution Results of Cases

$\|S\|$	$\|J\|$	$\|I\|$	Heuristic	IGA(3)	Ratio(3)	IGA(6)	Ratio(6)
2	5	2	1738.7	1738.7	0.0%	**1716.0**	1.3%
2	5	4	1453.5	1426.5	1.9%	**1406.5**	3.2%
2	5	6	1259.1	**1167.5**	7.3%	**1167.5**	7.3%
2	10	2	3421.5	3346.0	2.2%	**3322.0**	2.9%
2	10	4	2676.1	2676.1	0.0%	**2578.0**	3.7%
2	10	6	2222.3	2222.3	0.0%	**2197.0**	1.1%
2	15	2	5650.8	5198.5	8.0%	**5079.5**	10.1%
2	15	4	3922.8	**3616.0**	7.8%	3640.0	7.2%
2	15	6	**2844.1**	**2844.1**	0.0%	**2844.1**	0.0%
3	5	2	1549.4	1476.5	4.7%	**1455.0**	6.1%
3	5	4	924.6	805.0	12.9%	**804.5**	13.0%
3	5	6	1114.9	1021.0	8.4%	**998.5**	10.4%
3	10	2	3225.7	**3014.0**	6.6%	**3014.0**	6.6%
3	10	4	2079.8	2050.5	1.4%	**2002.5**	3.7%
3	10	6	2097.4	**2002.0**	4.5%	**2002.0**	4.5%
3	15	2	4857.7	4431.0	8.8%	**4406.5**	9.3%
3	15	4	2920.8	2751.0	5.8%	**2632.0**	9.9%
3	15	6	2389.6	2268.5	5.1%	**2196.5**	8.1%

can effectively allocate human resources within the concerned PAP. IGA can optimize resource utilization and significantly enhance the effectiveness of the heuristic algorithm.

The concerned human resource allocation problems can be further extended by combining it with some robot-assisted systems and other smart manufacturing systems [25]–[27]. Designing high-performance solution methods to deal with the extension problems would be meaningful future work.

Acknowledgment

This work was supported in part by National Key R&D Program of China (2021YFB3301200), National Natural Science Foundation of China (62073069 and 62203093), Guangdong Basic and Applied Basic Research Foundation (2021A1515110827), Liaoning Provincial Science and Technology Program Joint Program (Fund) Project (2023-MSBA-074), and Fundamental Research Funds for the Central Universities (N2404021).

References

[1] B. Wu, P. Lu, J. Lu, J. Xu, and X. Liu, "A hierarchical parallel multi-station assembly sequence planning method based on ga-dfla," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 236, no. 4, pp. 2029–2045, 2022.

[2] M. Ebrahimi, M. Mahmoodjanloo, B. Einabadi, A. Baboli, and E. Rother, "A mixed-model assembly line sequencing problem with parallel stations and walking workers: a case study in the automotive industry," *International Journal of Production Research*, vol. 61, no. 3, pp. 993–1012, 2023.

[3] C. Quan, Y. Lei, G. Jianming, L. Xingchen, Z. Yong, and C. Xiaoqian, "LEO mega-constellation network: networking technologies and state of the art," *Journal on Communications*, vol. 43, no. 5, pp. 177–189, 2022.

[4] Z. Liu, J. Liu, C. Zhuang, and F. Wan, "Multi-objective complex product assembly scheduling problem considering parallel team and worker skills," *Journal of Manufacturing Systems*, vol. 63, pp. 454–470, 2022.

[5] H. Ding, C. Zhuang, and J. Liu, "Extensions of the resource-constrained project scheduling problem," *Automation in Construction*, vol. 153, p. 104958, 2023.

[6] Z. Zhang, Z. Zhao, S. Qin, S. Liu, and M. Zhou, "Multi-product multi-stage multi-period resource allocation for minimizing batch-processing steel production cost," *IEEE Trans. on Automation Science and Engineering*, 2024, doi: 10.1109/TASE.2024.3418370.

[7] N. Christofides, R. Alvarez-Valdés, and J. M. Tamarit, "Project scheduling with resource constraints: A branch and bound approach," *European journal of operational research*, vol. 29, no. 3, pp. 262–273, 1987.

[8] S. Hartmann and D. Briskorn, "An updated survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of operational research*, vol. 297, no. 1, pp. 1–14, 2022.

[9] G. Zhu, J. F. Bard, and G. Yu, "A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem," *INFORMS Journal on Computing*, vol. 18, no. 3, pp. 377–390, 2006.

[10] R. Klein, "Bidirectional planning: improving priority rule-based heuristics for scheduling resource-constrained projects," *European Journal of Operational Research*, vol. 127, no. 3, pp. 619–638, 2000.

[11] K. Bouleimen and H. Lecocq, "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version," *European journal of operational research*, vol. 149, no. 2, pp. 268–281, 2003.

[12] Z. Zhao, Z. Bian, J. Liang, S. Liu, and M. Zhou, "Scheduling and logistics optimization for batch manufacturing processes with temperature constraints and alternative thermal devices," *IEEE Transactions on Industrial Informatics*, 2024.

[13] Z. Zhao, Q. Jiang, S. Liu, M. Zhou, X. Yang, and X. Guo, "Energy, cost and job-tardiness-minimized scheduling of energy-intensive and high-cost industrial production systems," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108477, 2024.

[14] D. Pisinger and S. Ropke, "Large neighborhood search," *Handbook of metaheuristics*, pp. 99–127, 2019.

[15] R. K. Chakrabortty, A. Abbasi, and M. J. Ryan, "Multi-mode resource-constrained project scheduling using modified variable neighborhood search heuristic," *International Transactions in Operational Research*, vol. 27, no. 1, pp. 138–167, 2020.

[16] Z. Zhao, J. Cheng, J. Liang, S. Liu, M. Zhou, and Y. Al-Turki, "Order picking optimization in smart warehouses with human-robot collaboration," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16 314–16 324, 2024.

[17] R. Kolisch and A. Drexl, "Local search for nonpreemptive multi-mode resource-constrained project scheduling," *IIE Transactions*, vol. 29, no. 11, pp. 987–999, 1997.

[18] L.-Y. Tseng and S.-C. Chen, "Two-phase genetic local search algorithm for the multimode resource-constrained project scheduling problem," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 848–857, 2009.

[19] N. Vakhania, "Single-machine scheduling with release times and tails," *Annals of Operations Research*, vol. 129, no. 1, pp. 253–271, 2004.

[20] I. Kacem, "Approximation algorithms for the makespan minimization with positive tails on a single machine with a fixed non-availability interval," *Journal of Combinatorial Optimization*, vol. 17, pp. 117–133, 2009.

[21] D. Xu and D.-L. Yang, "Makespan minimization for two parallel machines scheduling with a periodic availability constraint: mathematical programming model, average-case analysis, and anomalies," *Applied Mathematical Modelling*, vol. 37, no. 14-15, pp. 7561–7567, 2013.

[22] Z. Zhao, M. Zhou, and S. Liu, "Iterated greedy algorithms for flow-shop scheduling problems: A tutorial," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1941–1959, 2021.

[23] Z. Zhao, S. Liu, M. Zhou, D. You, and X. Guo, "Heuristic scheduling of batch production processes based on Petri nets and iterated greedy algorithms," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 251–261, 2020.

[24] Z. Zhao, S. Liu, M. Zhou, and A. Abusorrah, "Dual-objective mixed integer linear program and memetic algorithm for an industrial group scheduling problem," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 6, pp. 1199–1209, 2020.

[25] C. Zhang, Y. Wang, Z. Zhao, X. Chen, H. Ye, S. Liu, Y. Yang, and K. Peng, "Performance-driven closed-loop optimization and control for smart manufacturing processes in the cloud-edge-device collaborative architecture: A review and new perspectives," *Computers in Industry*, vol. 162, p. 104131, 2024.

[26] Z. Zhao, S. Li, S. Liu, M. Zhou, X. Li, and X. Yang, "Lexicographic dual-objective path finding in multi-agent systems," *IEEE Transactions on Automation Science and Engineering*, 2024.

[27] Z. Zhao, X. Li, S. Liu, M. Zhou, and X. Yang, "Multi-mobile-robot transport and production integrated system optimization," *IEEE Transactions on Automation Science and Engineering*, 2024.