Applied Intelligence (2021) 51:3936–3951
https://doi.org/10.1007/s10489-020-01975-y

An imperialist competition algorithm using a global search strategy for physical examination scheduling

Hui Yu[1] · **Jun-qing Li**[1,2] · **Lijing Zhang**[3] · **Peng Duan**[2]

Accepted: 24 September 2020 / Published online: 23 November 2020

Abstract

The outbreak of the novel coronavirus clearly highlights the importance of the need of effective physical examination scheduling. As treatment times for patients are uncertain, this remains a strongly NP-hard problem. Therefore, we introduce a complex flexible job shop scheduling model. In the process of physical examination for suspected patients, the physical examiner is considered a job, and the physical examination item and equipment correspond to an operation and a machine, respectively. We incorporate the processing time of the patient during the physical examination, the transportation time between equipment, and the setup time of the patient. A unique scheduling algorithm, called imperialist competition algorithm with global search strategy (ICA_GS) is developed for solving the physical examination scheduling problem. A local search strategy is embedded into ICA_GS for enhancing the searching behaviors, and a global search strategy is investigated to prevent falling into local optimality. Finally, the proposed algorithm is tested by simulating the execution of the physical examination scheduling processes, which verify that the proposed algorithm can better solve the physical examination scheduling problem.

Keywords Flexible job shop scheduling · Physical examination scheduling · Transportation time · Setup time · Imperialist competition algorithm · Local search · Global search

1 Introduction

The problem studied here is a result of the novel coronavirus outbreak that began in 2019, which first attacked Wuhan, China, and quickly spread to most regions of the country. Three months later, the virus had swept the world, and as of April 28, 2020, the novel coronavirus epidemic infected 3034801 people and killed 210511 (according to data released by Johns Hopkins University in the United States https://gisanddata.maps.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6). Facing so many patients within a short period became a critical challenge for hospitals with limited resources and equipment [1, 2]. Using available resources with the best efficiency, improving patient flow, and optimizing treatment management are crucial for hospitals [3]. With the growth of the disease, many hospitals expanded capacity because the number of patients, even in Wuhan, far exceeded the standard carrying capacity. However, due to many obstacles, these expansion activities encountered many restrictions, which further reduced the available resources for meeting established needs efficiently.

During the process of physical examinations of suspected patients with the novel coronavirus, the traditional extensive medical approach includes most physical examinations choosing the items with less waiting time. As the examinations increased in the queue at the physical examination center, more time was wasted with longer waits.

To reduce this bottleneck, we introduce a scheduling system that determines the start time of each patient in different testing groups as well as the order in the scheduling cycle. Additional medical challenges can be solved by enhanced scheduling. Hsieh [4] proposed a viable and systematic approach to develop a scalable and sustain-able scheduling system based on multi-agent system (MAS) to shorten patient stay in a hospital. Quintana et al. [5] studied the

✉ Jun-qing Li
lijunqing@lcu-cs.com

[1] School of Information Science and Engineering, Shandong Normal University, Jinan, China

[2] School of Computer, Liaocheng University, Liaocheng, China

[3] Liaocheng Jingxin Seamless Pipe Manufacturing Co., Ltd, Liaocheng, China

Home Health Care Scheduling Problem which involves allocating professional caregivers to patients' places of residence to meet service demands. Azaiez and Al-Sharif [6] developed a computerized nurse scheduling model approximated by 0-1 goal programming for improving manual scheduling. Pham and Klinkert [7] researched a new surgical case scheduling method to ensure the quality of patient care and effectively use hospital resources. Zhu et al. [8] proposed an efficient outpatient scheduling method and created a qualification matrix to apply the group role assignment algorithm, making an automatic outpatient scheduling feasible. Cappanera et al. [9] studied the performance of three scheduling strategies. The goal of this plan is to maximize the number of scheduled surgeries and balance the workload across beds and operating rooms. Gartner and Kolisch [10] studied hospital patient flow planning to maximize the contribution margin by developing two mixed-integer programming (MIP) processes to be embedded into static and rolling horizontal programming methods under the condition of scarcity of medical resources. This approach considered the procedures of the clinical pathway (such as the types of diagnostic activities and surgery) as well as the sequence in which they must be applied to the patient. Ahmadi-Javid et al. [11] developed a method based on decentralized agents, in which patients and hospital resources are represented as agents with individual goals to solve challenges of patient scheduling and hospital resources. Erhard et al. [12] adopted a quantitative method of doctor scheduling in hospitals by describing the related characteristics of various doctor scheduling problems investigated. Jiang et al. [13] proposed the two simple scheduling strategies of weight accumulation and priority enhancement for improving waiting time management. These experimental results suggested that an effective scheduling strategy can significantly reduce patient waiting times without expensive capacity expansions.

Scheduling is becoming more and more popular in recent years [14, 15]. The flexible job shop scheduling problem (FJSP) is an extension of job shop scheduling [16]. FJSP primarily considers two problems, the first of which sorts all operations of the job into a reasonable order, and the second assigns each sorted operation an appropriate and available machine. The makespan represents the maximum completion time of a job.

The FJSP model plays an important role in the medical field and services with its application, enabling hospital resources to be effectively used in limited time. The FJSP provides a pre-analysis of the processing time and notifies employees in advance of upcoming activities so that patients can be prepared quickly. Yin et al. [17] represented surgery scheduling as an extended multi-resource, constrained FJSP that was solved by an improved ant colony algorithm. Su et al. [18] regarded the problem of determining the optimal operating room schedule as a FJSP and proposed a SOMO-based approach for solving the operating room scheduling problem. Lee et al. [19] studied the problem of completing multiple processes within one day as a flexible job shop model with fuzzy sets and proposed a scheduling strategy to determine the start time of multiple operating rooms. Behmanesh et al. [20] researched the surgical case scheduling problem in multioperating theater environment with uncertain service times in order to minimize makespan and structured the no-wait multi-resources fuzzy fexible job shop scheduling in operating theater. Also, Behmanesh and Zandieh [21] studied the Surgical case scheduling problem with fuzzy surgery time and formulated the problem as a novel bi-objective no-wait multi-resource FJSP. The surgical cases are optimally allocated to the existing resources and sequenced in the surgery list of these resources so as to minimize both objectives within their time window. Luscombe et al. [22] proposed a dynamic scheduling framework to provide real-time support for the management of scarce resources in emergency departments. Recently, many meta-heuristic algorithms have been applied to FJSP, such as the simulated annealing algorithm [23] based on local search heuristics, variable neighborhood search (VNS) [24], the Tabu search algorithm [25], and the iterative greedy (IG) algorithm [26], as well as population-based methods, including the particle swarm optimization algorithm (PSO) [27], a hybrid discrete artificial bee colony (ABC) algorithm [28], An improved Jaya (IJaya) algorithm [29], an improved artificial immune system (IAIS) algorithm [30], and discrete imperialist competitive algorithm (DICA) [31]. The traditional open shop scheduling method is not flexible enough, and many types of medical equipment cannot be fully utilized during an emergency. The main contribution of this paper is that the physical examination scheduling is regarded as a FJSP, and an improved imperialist competition algorithm is applied to the improvement of hospital physical examination scheduling. The improved local search strategy is embedded in the algorithm to enrich the search behavior and avoid premature convergence. A global search strategy is studied to prevent falling into local optimization. The optimization idea of sorting and scheduling is fully combined to make effective use of limited resources and equipment.

The remainder of this paper is organized as follows. The second section describes the scheduling environment and formally states the problem. The third section details the description of the algorithm, and the fourth section reviews the experiment to verify the algorithm. Finally, the fifth section provides a summary and suggests future research.

2 The environment of scheduling

For an example scheduling scenario, we first investigate the physical examination of the patient suspected to have the novel coronavirus. With a pre-analysis of the time required to perform the physical examination, the exam is then scheduled and completed as planned. Our goal is to identify a scheduling scheme that allows for the shortest completion times within this environment.

2.1 Flexible job shop scheduling

The FJSP is defined as follows. A set of jobs $J=\{J_1, ..., J_n\}$ must be performed on a set of machines $M=\{M_1, ..., M_n\}$ where each independent job J_i consists of a sequence of operations O_i. To apply FJSP, we consider the patient as a job, each physical examination item as an operation, and the physical equipment is considered machine. Considering the transportation time between the input warehouse and the first machine, a virtual machine with zero processing time is defined, machine 0. Within the processing time of the patient during physical examination, the transportation time between equipment and the setup time required for the patient are also incorporated. The FJSP approach is adopted because patients can be assigned to a variety of physical examination equipment. Figure 1 depicts the physical examination scheduling as a FJSP, because the transportation time and setup time are taken into account, therefore, patients will not proceed directly when it reaches equipment. In addition, some patients may have a physical examination on the same device, therefore patient 2 has a physical examination in equipment 1 twice, indicating that different physical examination items of patient 2 are carried out on the same physical examination equipment.

2.2 Assumptions

During the process of physical examination scheduling, the following constraints exist:

- All patients are ready at time zero.
- Each patient has a fixed number of physical examinations operations.
- Transportation and setup times must be considered.
- Once a patient is examined on equipment, the process is not interrupted until the entire routine is complete.
- The time of the next operation is greater than or equal to the time of the previous operation.

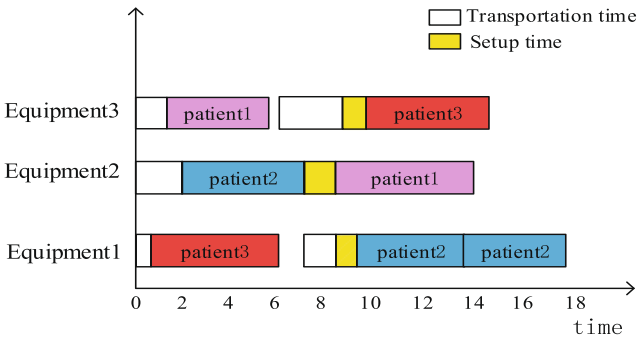


Fig. 1 The physical examination scheduling process

2.3 Problem model

The parameters and indexes are listed in Table 1. For a mathematical model of FJSP, a sequence-based mixed-integer linear programming model is established to minimize the physical examination times of patients. The objective is to to minimize the makespan of a patient's physical examination. Therefore, the objective function can be expressed as constraint (1), where $C_{\max}$ represents the maximum completion time.

$$MinC_{\max} \tag{1}$$

Constraint (2) and (3) guarantees that every physical examination item should be allocated to only one equipment. If the $Y_{j,1,i,0}$ value of is 1 that represents the first operation of patient j is processed on machine i.

$$\sum_{i=1}^{m} Y_{j,q,i,0} = 1 \quad \forall j \tag{2}$$

Table 1 The variable definitions

Variables	Comment
j, h	The indices of the patients.
i, f, k	The indices of the physical examination equipment.
q, e	The index of the physical examination items.
n	The number of patients.
m	The number of physical examination equipment.
M	A large positive number.
E_i	The available physical examination equipment.
n_j	The number of physical examination items of patient j.
$O_{j,q}$	q^{th} operation of job j.
$E_{j,q,i}$	A binary variableoftaking value 1 if $O_{j,q}$ is processed on machine i, and 0 otherwise.
$P_{j,q,i}$	The processing time of $O_{j,q}$ on machine i.
$T_{j,k,i}$	The transportation time of the job j from the machine k to machine i.
$S_{i,h,j}$	The setup time between job j and job h on machine i.
$C_{j,q}$	A continuous variable for the completion time of $O_{j,q}$.
$X_{j,q,h,e}$	A binary variable taking value 1 if $O_{j,q}$ is processed after $O_{h,e}$, and 0 otherwise.
$Y_{j,q,i,k}$	A binary variable taking value 1 if $O_{j,q}$ is processed on machine i and $O_{j,q-1}$ on machine k, and 0 otherwise.

$$\sum_{i=1}^{m}\sum_{k=1}^{m} Y_{j,q,i,k} = 1 \ \ \forall j, q > 1 \tag{3}$$

During the physical examination that each physical examination item for each patient is allocated to a single piece of equipment, and that these must be selected from a series of available equipment, which can be expressed by constraints (4) and (5).

$$\sum_{k=1}^{m} Y_{j,1,i,0} \le E_{j,1,i} \ \ \forall j, i \tag{4}$$

$$\sum_{k=1}^{m} Y_{j,q,i,k} \le E_{j,q,i} \ \ \forall j, q > 1, i \tag{5}$$

According to the definition of variable $Y_{j,q,i,k}$, $O_{j,q-1}$ should be processed on equipment k, if is processed on $O_{j,q}$ equipment i. Therefore, constraints (6) and (7) ensure that if $O_{j,q}$ is operated on machine i, $O_{j,q-1}$ is processed on machine k.

$$Y_{j,2,i,k} \le Y_{j,1,k,0} \ \ \forall j, i, k \tag{6}$$

$$Y_{j,q,i,k} \le \sum_{f=1}^{m} Y_{j,q-1,k,f} \ \ \forall j, q > 2, i, k \tag{7}$$

The constraints (8)–(17) are sequencing constraints. Constraints (8) and (9) enforce that starts $O_{j,q}$ just after the completion of patient $O_{j,q-1}$ and the transportation and setup time of patient j to equipment i. Constraints (10) to (17) ensure that equipment can only be applied to one patient at a time.

$$C_{j,1} \ge \sum_{k=1}^{m} Y_{j,1,i,0}(P_{j,1,i} + T_{j,0,i} + \sum_{h=1}^{n} X_{j,1,h,1} \cdot S_{i,h,j}) \ \ \forall j \tag{8}$$

$$\begin{aligned} C_{j,q} \ge\ & C_{j,q-1} + \sum_{i=1}^{m}\sum_{k=1}^{m} Y_{j,q,i,k}(P_{j,q,i} + T_{j,k,i} \\ & + \sum_{h=1}^{n} X_{j,q,h,e} \cdot S_{i,h,j}) \ \forall j, q > 1, h, e \end{aligned} \tag{9}$$

$$\begin{aligned} C_{j,q} \ge\ & C_{h,z} + P_{j,q,i} + S_{i,h,j} - M(1 - X_{j,q,h,z}) \\ & M(2 - \sum_{k=1}^{m} Y_{j,q,i,k} - \sum_{k=1}^{m} Y_{h,z,i,k}) \\ & \forall j, q > 1, h \neq, e > 1 \end{aligned} \tag{10}$$

$$\begin{aligned} C_{j,1} \ge\ & C_{h,z} + P_{j,1,i} + S_{i,h,j} - M(1 - X_{j,1,h,z}) \\ & -M(2 - Y_{j,1,i,0} - \sum_{k=1}^{m} Y_{h,z,i,k}) \\ & \forall j, q > 1, h \neq j, e > 1 \end{aligned} \tag{11}$$

$$\begin{aligned} C_{j,q} \ge\ & C_{h,1} + P_{j,q,i} + S_{i,h,j} - M(1 - X_{j,q,h,1}) \\ & -M(2 - \sum_{k=1}^{m} Y_{j,q,i,k} - Y_{h,1,i,0}) \\ & \forall j, q > 1, h \neq j, e > 1 \end{aligned} \tag{12}$$

$$\begin{aligned} C_{j,1} \ge\ & C_{h,1} + P_{j,1,i} + S_{i,h,j} - M(1 - X_{j,1,h,1}) \\ & -M(2 - Y_{j,1,i,0} - Y_{h,1,i,0}) \\ & \forall j, q, h \neq j, e \end{aligned} \tag{13}$$

$$\begin{aligned} C_{h,z} \ge\ & C_{j,q} + P_{h,z,i} + S_{i,h,j} - M(X_{j,q,h,z}) \\ & -M(2 - \sum_{k=1}^{m} Y_{j,q,i,k} - \sum_{k=1}^{m} Y_{h,z,i,k}) \\ & \forall j, q, h \neq j, e > 1 \end{aligned} \tag{14}$$

$$\begin{aligned} C_{h,z} \ge\ & C_{j,1} + P_{h,z,i} + S_{i,h,j} - M(X_{j,1,h,z}) \\ & -M(2 - Y_{j,1,i,0} - \sum_{k=1}^{m} Y_{h,z,i,k}) \\ & \forall j, q > 1, h \neq j, e \end{aligned} \tag{15}$$

$$\begin{aligned} C_{h,1} \ge\ & C_{j,q} + P_{h,1,i} + S_{i,h,j} - M(X_{j,q,h,1}) \\ & -M(2 - \sum_{k=1}^{m} Y_{j,q,i,k} - Y_{h,1,i,0}) \\ & \forall j, q, h \neq j, e \end{aligned} \tag{16}$$

$$\begin{aligned} C_{h,1} \ge\ & C_{j,1} + P_{h,1,i} + S_{i,h,j} - M(X_{j,1,h,1}) \\ & -M(2 - Y_{j,1,i,0} - Y_{h,1,i,0}) \\ & \forall j, q, h \neq j, e \end{aligned} \tag{17}$$

The completion time of the whole physical examination scheduling is greater than or equal to the completion time of each physical examination scheduling, and each scheduling time is more than 0, which can be expressed by constraint (18) and (19).

$$C_{\max} \ge C_{j,n_j} \ \ \forall j \tag{18}$$

$$C_{j,q} \ge 0 \ \ \forall j, q \tag{19}$$

Constraint (20) shows that those related variables are binary.

$$X_{j,q,h,z}, Y_{j,q,i,k} \in \{0, 1\} \tag{20}$$

3 Algorithm descriptions

To solve the physical examination scheduling problems, we apply a new intelligent optimization algorithm, called the imperialist competition algorithm that is inspired by the competitive behavior of the imperialist. In the ICA_GS, individuals are represented as countries. First, the better countries with smaller makespan values are

called imperialists, and others become colonies of these imperialists. Then, the imperialists who occupy the colonies attempt to assimilate them and apply a mutation before the assimilation. After the internal strengths of the imperialists change, the strategy of the imperial competition and updates is implemented. During this process, the most powerful colony replaces the weakest imperialist. When an imperialist no longer has colonies, it undergoes a demise strategy to re-assign it to the imperialist that is the most power over their colonies.

The adoption of a strategy for an imperial development plan satisfies the fact that imperialists feature certain development strengths and ruling strategies in the real world. For the process of continuous development, there may appear similar countries, and similarity significantly reduces the performance of the algorithm. Therefore, we adopt a similarity replacement strategy. In addition, a local search strategy is embedded in the ICA_GS for enhancing search behaviors and prevented the solution from falling into a local optimization that was introduced by the global search strategy. The framework for the ICA_GS is described in Algorithm 1.

Algorithm 1 ICA_GS.

Input: All populations
Output: The best solution
1: Initialize the populations (by LS).
2: Initialize the empires.
3: **for** each imperialist i **do**
4: Mutation operation.
5: Assimilation operation.
6: **end for**
7: Update the imperialists.
8: Do the competition mechanism.
9: **for** each imperialist i **do**
10: Conduct the imperialists' development plans.
11: Similarity substitution.
12: **end for**
13: Local Search strategy.
14: Global Search strategy.

3.1 Representation

Each solution uses a multi-layer coding rule that contains information about the equipment selection (EQS) and examination sequence (EXS) (Fig. 2). The equipment selection randomly selects available physical examination equipment according to the physical examination items of the patient and stores the available physical examination equipment numbers into the EQS array. For the examination sequence, the patient number is stored in the EXS array according to a random order of the patient underwent a physical examination. The initial solution requires the corresponding processing of the EQS part, for which we adopt the LS strategy proposed by Zhang et al. [32], and the EXS part, for which we use a random selection strategy.

3.2 Initialize the empires

In the ICA_GS, all populations (Pop) include several imperialists (Nim) and their countries (Ncl). The imperialists have the smaller makespan, and the remainder are colonies of the imperialists, such that

$$Pop = Nim + Ncl \tag{21}$$

A roulette selection process is applied to calculate the $power$ of each imperialist as

$$power(n) = \frac{1}{makespan(n)} \tag{22}$$

The calculation of the number of colonies occupied by each imperialist ($colonyNum$) is performed according to

$$colonyNum(n) = Ncl \cdot \frac{power(n)}{\sum_{n=1}^{Nim} power(n)} \tag{23}$$

3.3 Mutation operation

Mutation is a strategy adopted by the imperialist to reduce repetition (Fig. 3). If the imperialist improves after a mutation, then the new imperialist replaces the previous imperialist. Otherwise, the imperialist remains unchanged. The operation of the mutation follows the process as:

(1) For the EQS part,

Step 1: Randomly select several positions.
Step 2: Replace the elements of the selected positions with the available equipment numbers.

(2) For the EXS part,

Step 1: Randomly select two positions.
Step 2: Swap the element of the two positions.

3.4 Assimilation operation

The process by which colonies learn from empires is called assimilation. The steps for the assimilation strategies are as follows:

(1) For the EQS part, we apply the two-point crossover strategy [33] (Fig. 4).

Step 1: Choose an imperialist and one of its colonies, with EQS parts represented as Q1 and Q2, respectively.
Step 2: Randomly select two positions from Q1.

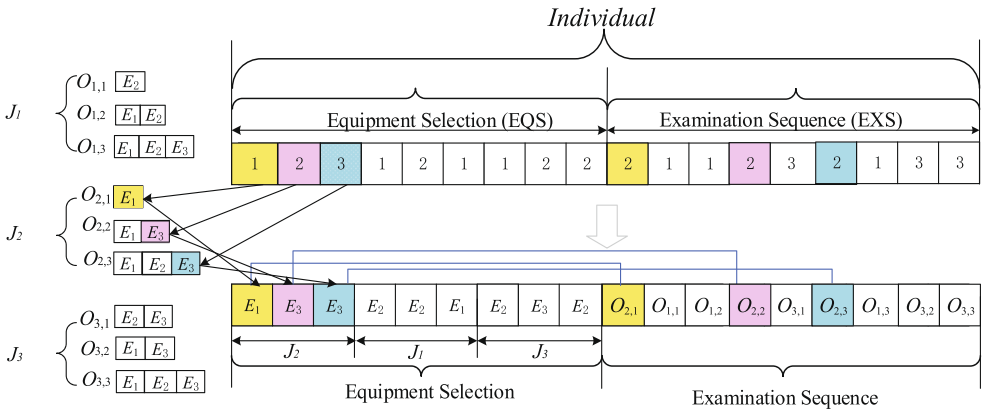


Fig. 2 An individual representation in the scheduler

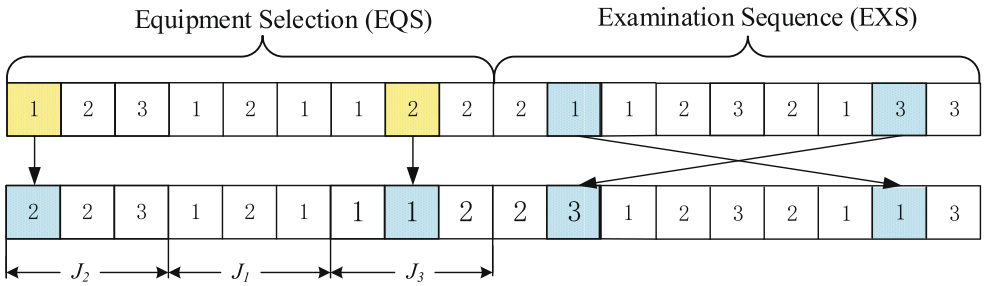


Fig. 3 The mutation process

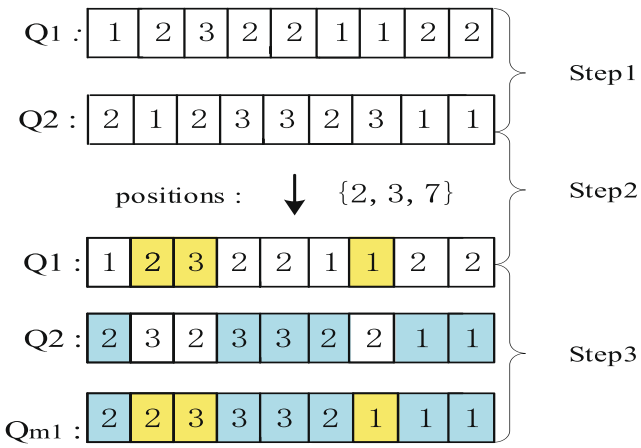


Fig. 4 The assimilation strategy of the EQS part

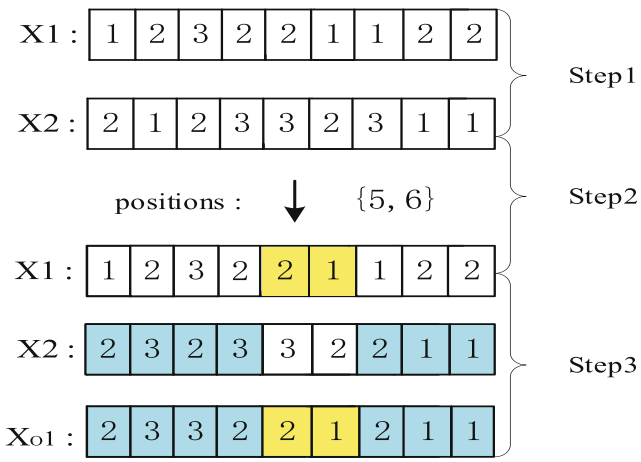


Fig. 5 The assimilation strategy of the EXS part

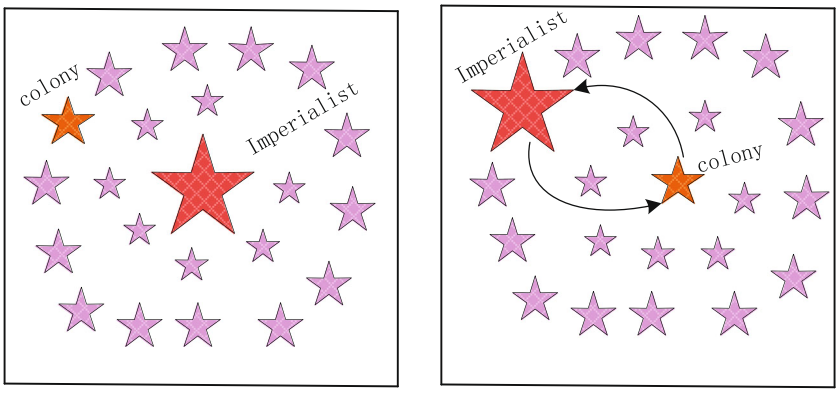


Fig. 6 The imperialist's update strategy

Step 3: Insert elements between the selected positions in Q1 into the EQS part (Qm1) of the new colony. Then, the remaining position inserts the element of the corresponding previous colony.

(2) For the EXS part, we adopt the POX strategy [34] (Fig. 5).

Step 1: Choose an imperialist and one of its colonies, with their EXS parts represented by X1 and X2, respectively.
Step 2: Randomly select several positions from X1.
Step 3: Insert the elements of the selected positions from the imperialist into the EXS part (Xo1) of the new colony, and insert the other elements of X2 into Xo1 in order.

A new colony is obtained after assimilation. If the new colony is improved over the previous version, then the previous colony is replaced. Otherwise, the previous colony remains.

3.5 The imperialists' updating

Through iteration, colonies may gain more power than the corresponding imperialist. In this scenario, the most powerful colony becomes the new imperialist. The colonies under the previous imperialist are then allocated to the new imperialist. The previous imperialist also becomes a colony of the new imperialist (Fig. 6).

3.6 The competition mechanism

During the imperialist competition, the strongest and weakest imperialists are selected. Then, the colonies of the weakest imperialist transfer to belonging to the most powerful imperialist (Fig. 7). When an imperialist does not have any colonies, then the imperialist performs the strategy of extinction.

3.7 The imperialists' development

To improve the performance of ICA_GS, we apply the development mechanism to make some changes to the imperialist. This process is also separated into two parts.

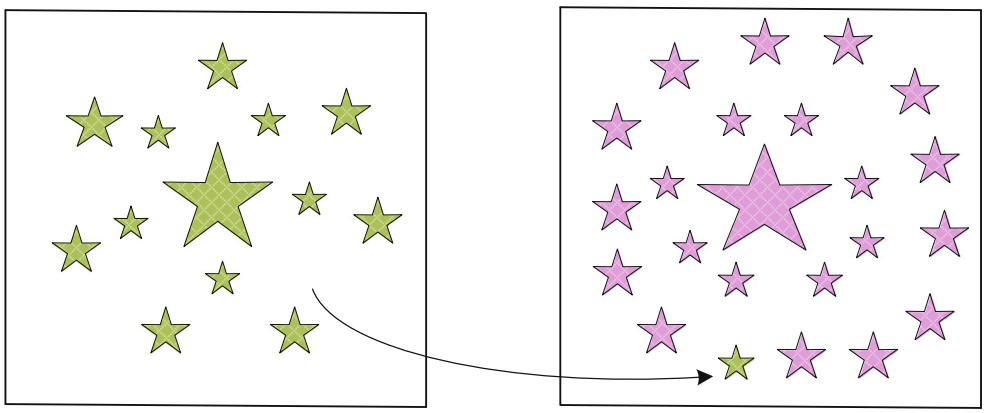


Fig. 7 The imperialist's update strategy

Table 2 Combination of the key parameter values

Parameter	Level			
	1	2	3	4
Pop	50	100	150	200
Nim	5	10	15	20
$IterOS$	1	3	6	9
$IterMS$	1	3	6	9

Step 1: For the EQS part, a position is randomly selected and replaced with an available equipment number .

Step 2: For the EXS part, two positions are randomly selected, and the elements of the first position are moved into the second position. The element between these two positions moves forward.

3.8 Similarity substitution

During the process of continuous development of the empires, highly similar countries may appear; this reduces the performance of the algorithm. For addressing this issue, we adopt the following similarity replacement strategy.

Step 1: Randomly select two colonies within the imperialist.

Step 2: Compare the similarity between the elements in the two parts of the solution of the selected colonies.

Step 3: After comparison, those pairs with the highest similarity are selected, and then initialize one of the countries to reduce similarity of the colonies.

3.9 Local search strategy

To further enhance the performance of the proposed algorithm, a local search strategy is next introduced. The number of iterations in the inner loop is expressed by $IterMS$, and the number of iterations in the outer loop is expressed by $IterOS$. The internal loop section uses the same strategy as the EQS part of the mutation operation. If a better solution can be obtained, then it replaces the previous solution. Otherwise, the previous solution is replaced with a lower probability. The outer loop uses the two-point crossover strategy and replaces the previous solution if it obtains a better one. Otherwise, the previous solution is replaced with a lower probability. The framework for this local search approach is described in Algorithm 2.

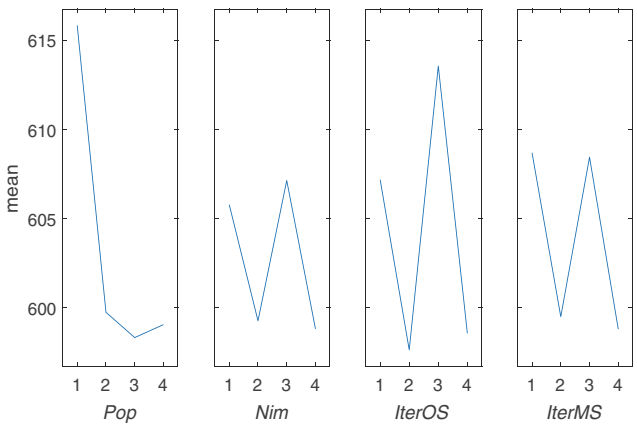


Fig. 8 The factor level trends of the four key parameters

Algorithm 2 Local search strategy.

Input: Each imperialist
Output: Local optimal solution

```
for each imperialist i do
  for j = 1 to IterOS do
    Two-point crossover strategy.
    for k = 1 to IterMS do
      Conduct the same strategy as the EQS part of the
       mutation operation described in subsection 3.3.
      if new solution <current solution then
        Accept the new solution.
      else
        Accept the new solution with a low
         probability.
      end if
    end for
    if new solution <current solution then
      Accept the new solution.
    else
      Accept the new solution with a low probability.
     end if
  end for
end for
```

3.10 Global search strategy

A global search strategy is applied to improve the unimproved solution during the last several iterations and eliminate the state that exists in a local optimum. This final optimization enables the entire population to achieve a better solution.

Step 1: Create a vector $W1$ for storing the replaced solution from each evolution when the solution cannot be improved after a given iteration.

Step 2: Create a vector $W2$ for storing the local optimal solution.

Step 3: Apply the following two methods of random selection to produce the optimal solution.

Table 3 Compare the experimental data ICA and ICA_GS

Instance	Scale	Best	ICA			ICA_GS			Dev	
			min	avg	max	min	avg	max	dev1	dev2
Inst1	3x3	87.00	87.00	87.00	87.00	87.00	87.00	87.00	**0.00**	**0.00**
Inst2	3x5	126.00	132.00	136.47	146.00	126.00	133.50	134.00	4.76	**0.00**
Inst3	3x6	142.00	158.00	175.00	187.00	142.00	158.27	169.00	11.27	**0.00**
Inst4	3x7	167.00	180.00	200.70	213.00	167.00	183.77	193.00	7.78	**0.00**
Inst5	3x8	221.00	235.00	247.50	256.00	221.00	234.30	241.00	6.33	**0.00**
Inst6	3x10	275.00	282.00	300.93	310.00	275.00	286.83	296.00	2.55	**0.00**
Inst7	5x3	137.00	137.00	145.10	150.00	138.00	139.07	143.00	**0.00**	0.73
Inst8	5x5	154.00	164.00	185.30	190.00	154.00	164.40	180.00	6.49	**0.00**
Inst9	5x6	220.00	236.00	246.10	256.00	220.00	228.47	237.00	7.27	**0.00**
Inst10	5x7	240.00	253.00	276.90	286.00	240.00	256.53	267.00	5.42	**0.00**
Inst11	5x8	273.00	282.00	302.90	315.00	273.00	282.83	290.00	3.30	**0.00**
Inst12	5x10	306.00	327.00	347.33	361.00	306.00	323.97	335.00	6.86	**0.00**
Inst13	10x3	114.00	114.00	115.27	116.00	114.00	114.03	115.00	**0.00**	**0.00**
Inst14	10x5	165.00	168.00	187.70	199.00	165.00	174.93	181.00	1.82	**0.00**
Inst15	10x6	249.00	276.00	285.60	295.00	249.00	266.17	275.00	10.84	**0.00**
Inst16	10x7	193.00	215.00	227.33	239.00	193.00	213.10	221.00	11.40	**0.00**
Inst17	10x8	328.00	361.00	378.57	401.00	328.00	348.53	361.00	10.06	**0.00**
Inst18	10x10	319.00	368.00	389.90	407.00	319.00	339.20	354.00	15.36	**0.00**
Inst19	20x3	301.00	309.00	315.77	321.00	301.00	306.30	309.00	2.66	**0.00**
Inst20	20x5	279.00	289.00	308.87	329.00	279.00	288.43	298.00	3.58	**0.00**
Inst21	20x6	390.00	417.00	444.73	459.00	390.00	416.33	428.00	6.92	**0.00**
Inst22	20x7	344.00	374.00	395.83	412.00	344.00	364.77	380.00	8.72	**0.00**
Inst23	20x8	439.00	514.00	542.33	565.00	439.00	469.07	520.00	17.08	**0.00**
Inst24	20x10	487.00	547.00	567.63	595.00	487.00	518.47	538.00	12.32	**0.00**
Inst25	30x3	424.00	438.00	449.83	458.00	424.00	429.20	443.00	3.30	**0.00**
Inst26	30x5	467.00	488.00	508.40	527.00	467.00	482.07	494.00	4.50	**0.00**
Inst27	30x6	674.00	699.00	741.33	771.00	674.00	688.27	712.00	3.71	**0.00**
Inst28	30x7	517.00	547.00	608.03	628.00	517.00	568.87	593.00	5.80	**0.00**
Inst29	30x8	654.00	710.00	734.03	771.00	654.00	687.03	711.00	8.56	**0.00**
Inst30	30x10	700.00	728.00	784.37	815.00	700.00	725.87	761.00	4.00	**0.00**
Inst31	50x3	742.00	753.00	764.23	776.00	742.00	753.20	758.00	1.48	**0.00**
Inst32	50x5	731.00	778.00	803.33	830.00	731.00	760.77	776.00	6.43	**0.00**
Inst33	50x6	924.00	1027.00	1065.33	1105.00	924.00	1001.80	1024.00	11.15	**0.00**
Inst34	50x7	827.00	881.00	920.20	961.00	827.00	868.93	892.00	6.53	**0.00**
Inst35	50x8	961.00	1027.00	1067.27	1118.00	961.00	1003.30	1039.00	6.87	**0.00**
Inst36	50x10	1073.00	1159.00	1214.27	1261.00	1073.00	1132.47	1180.00	8.01	**0.00**
Inst37	100x3	1547.00	1562.00	1573.33	1591.00	1547.00	1558.03	1566.00	0.97	**0.00**
Inst38	100x5	1513.00	1551.00	1596.17	1636.00	1513.00	1540.77	1562.00	2.51	**0.00**
Inst39	100x6	1991.00	2099.00	2140.30	2182.00	1991.00	2055.80	2094.00	5.42	**0.00**
Inst40	100x7	1578.00	1636.00	1699.50	1746.00	1578.00	1619.97	1643.00	3.68	**0.00**
Inst41	100x8	1833.00	1889.00	1971.90	2022.00	1833.00	1887.23	1929.00	3.06	**0.00**
Inst42	100x10	1858.00	1858.00	2006.43	2084.00	1859.00	1937.63	1980.00	**0.00**	0.05
mean		594.52	625.12	653.79	675.64	594.57	619.04	635.93	5.92	**0.02**

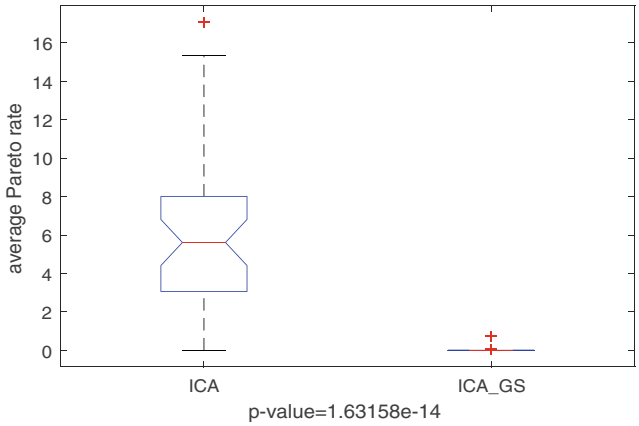


Fig. 9 ANOVA result for the ICA and ICA_GS

Step 3.1: Cross two randomly selected solutions from $W1$ to produce a new solution that replaces the original optimal value only if the new solution is better.

Step 3.2: Cross two solutions from $W2$ to produce a new solution by replacing the optimal value generated from the local search strategy only if the newly generated solution is better. Otherwise, do not perform the replacement.

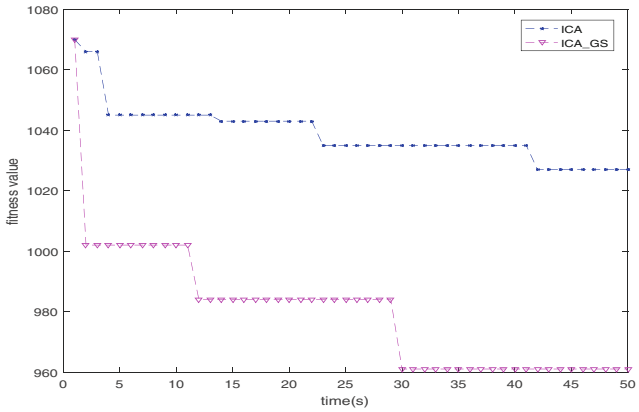
The framework for the global search approach is described in Algorithm 3.

Algorithm 3 Global search strategy.

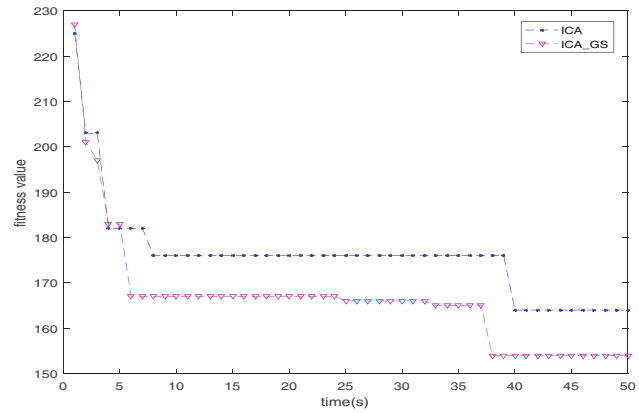
Input: $W1$, $W2$
Output: Global optimal solution
1: I = rand () % 2.
2: **switch** I
3: **case** 0:
4: **for** j= 1 to $W1$.size () **do**
5: Select two solutions from $W1$.
6: Crossover strategy.
7: **if** new solution <current solution **then**
8: Accept the new solution.
9: **end if**
10: **end for**
11: **case** 1:
12: **for** j= 1 to $W2$.size () **do**
13: Select two solutions from $W2$.
14: Crossover strategy.
15: **if** new solution <current solution **then**
16: Accept the new solution.
17: **end if**
18: **end for**
19: **end switch**

4 Experimental analysis

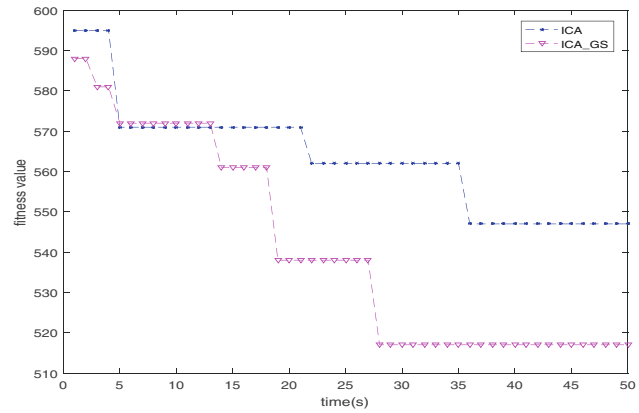
We simulate the execution of the physical examination scheduling process by studying the operation of jobs on different machines and verifying the advantages of the algorithm. All numerical experiments are performed on a Lenovo PC with a 3.3-GHz processor and 4-GB memory



(a) Convergence curve for Inst8.



(b) Convergence curve for Inst18.



(c) Convergence curve for Inst28.

Fig. 10 Convergence curves of the ICA and ICA_GS

Table 4 Global search numerical verification

Instance	Scale	Best	ICA_NGS			ICA_GS			Dev	
			min	avg	max	min	avg	max	dev1	dev2
Inst1	3x3	87.00	87.00	87.00	87.00	87.00	87.00	87.00	**0.00**	**0.00**
Inst2	3x5	126.00	126.00	136.47	146.00	126.00	133.50	134.00	**0.00**	**0.00**
Inst3	3x6	142.00	143.00	175.00	187.00	142.00	158.27	169.00	0.01	**0.00**
Inst4	3x7	167.00	167.00	200.70	213.00	167.00	183.77	193.00	**0.00**	**0.00**
Inst5	3x8	221.00	221.00	247.50	256.00	221.00	234.30	241.00	**0.00**	**0.00**
Inst6	3x10	275.00	275.00	300.93	310.00	275.00	286.83	296.00	**0.00**	**0.00**
Inst7	5x3	137.00	137.00	145.10	150.00	138.00	139.07	143.00	**0.00**	0.01
Inst8	5x5	154.00	166.00	189.30	196.00	154.00	164.40	180.00	0.08	**0.00**
Inst9	5x6	220.00	220.00	246.10	256.00	220.00	228.47	237.00	**0.00**	**0.00**
Inst10	5x7	240.00	241.00	276.90	286.00	240.00	256.53	267.00	**0.00**	**0.00**
Inst11	5x8	273.00	273.00	282.90	291.00	273.00	282.83	290.00	**0.00**	**0.00**
Inst12	5x10	306.00	306.00	347.33	361.00	306.00	323.97	335.00	**0.00**	**0.00**
Inst13	10x3	114.00	114.00	115.27	116.00	114.00	114.03	115.00	**0.00**	**0.00**
Inst14	10x5	165.00	168.00	187.70	199.00	165.00	174.93	181.00	0.02	**0.00**
Inst15	10x6	249.00	259.00	285.60	295.00	249.00	266.17	275.00	0.04	**0.00**
Inst16	10x7	193.00	193.00	227.33	239.00	193.00	213.10	221.00	**0.00**	**0.00**
Inst17	10x8	328.00	336.00	378.57	401.00	328.00	348.53	361.00	0.02	**0.00**
Inst18	10x10	319.00	323.00	369.90	407.00	319.00	339.20	354.00	0.01	**0.00**
Inst19	20x3	301.00	306.00	315.77	321.00	301.00	306.30	309.00	0.02	**0.00**
Inst20	20x5	279.00	283.00	308.87	329.00	279.00	288.43	298.00	0.01	**0.00**
Inst21	20x6	390.00	394.00	444.73	459.00	390.00	416.33	428.00	0.01	**0.00**
Inst22	20x7	344.00	359.00	395.83	412.00	344.00	364.77	380.00	0.04	**0.00**
Inst23	20x8	439.00	479.00	542.33	565.00	439.00	469.07	520.00	0.09	**0.00**
Inst24	20x10	487.00	504.00	567.63	595.00	487.00	518.47	538.00	0.03	**0.00**
Inst25	30x3	424.00	428.00	449.83	458.00	424.00	429.20	443.00	0.01	**0.00**
Inst26	30x5	467.00	469.00	508.40	527.00	467.00	482.07	494.00	**0.00**	**0.00**
Inst27	30x6	670.00	670.00	741.33	771.00	674.00	688.27	712.00	**0.00**	0.01
Inst28	30x7	517.00	553.00	608.03	628.00	517.00	568.87	593.00	0.07	**0.00**
Inst29	30x8	654.00	668.00	734.03	771.00	654.00	687.03	711.00	0.02	**0.00**
Inst30	30x10	698.00	698.00	784.37	815.00	700.00	725.87	761.00	**0.00**	**0.00**
Inst31	50x3	742.00	742.00	764.23	776.00	742.00	753.20	758.00	**0.00**	**0.00**
Inst32	50x5	731.00	736.00	803.33	830.00	731.00	760.77	776.00	0.01	**0.00**
Inst33	50x6	924.00	984.00	1065.33	1105.00	924.00	1001.80	1024.00	0.06	**0.00**
Inst34	50x7	827.00	832.00	920.20	961.00	827.00	868.93	892.00	0.01	**0.00**
Inst35	50x8	961.00	965.00	1067.27	1118.00	961.00	1003.30	1039.00	**0.00**	**0.00**
Inst36	50x10	1073.00	1085.00	1214.27	1261.00	1073.00	1132.47	1180.00	0.01	**0.00**
Inst37	100x3	1547.00	1547.00	1573.33	1591.00	1547.00	1558.03	1566.00	**0.00**	**0.00**
Inst38	100x5	1513.00	1517.00	1596.17	1636.00	1513.00	1540.77	1562.00	**0.00**	**0.00**
Inst39	100x6	1991.00	2001.00	2140.30	2182.00	1991.00	2055.80	2094.00	0.01	**0.00**
Inst40	100x7	1578.00	1585.00	1699.50	1746.00	1578.00	1619.97	1643.00	**0.00**	**0.00**
Inst41	100x8	1833.00	1837.00	1971.90	2022.00	1833.00	1887.23	1929.00	**0.00**	**0.00**
Inst42	100x10	1859.00	1980.00	2006.43	2084.00	1859.00	1937.63	1980.00	0.07	**0.00**
mean		594.4	604.21	652.93	675.21	594.57	619.04	635.93	0.02	**0.00**

running Windows 7. The FJSP method was written in C++ to improve speed and robustness.

4.1 Experimental parameters

We test the influence of the four parameters of Pop, Nim, $IterOS$, and $IterMS$ on the performance of the algorithm. First, the levels of each parameter are listed in Table 2. With these values, the Taguchi method (Montgomery [35] 2005) was introduced with which an orthogonal array L_{16} is constructed. For each parameter combination, the proposed algorithm independently ran 30 times, and the average fitness value of the algorithm was collected as the response variable. Finally, the factor level trend chart for the four parameters was drawn based on the obtained data. As seen in Fig. 8, when Pop is at level 3, Nim at level 4, $IterOS$ at level 2, and $IterMS$ at level 4, the proposed ICA_GS algorithm achieves the best performance.

4.2 Comparison of ICA_GS and ICA

For evaluating the performance of the proposed algorithm, the ICA [36] was selected for comparison because it adopts a new local search strategy that makes the algorithm converge well. Our algorithm is also formed by improving the local search strategy as well as adding a new strategy. The performance measure considered is the percentage deviation (dev) of the best value, calculated as

$$dev = \frac{f_c - f_b}{f_b} * 100\% \tag{24}$$

where f_b represents the best solution of all comparison algorithms and f_c is the best solution to the tested algorithm. Each algorithm runs 30 times independently, each time for 30 s, the best solution, the worst solution, and the average solution from the algorithm are presented in Table 3.

The first column in Table 3 represents the instance name, The second columns provides scale size of the algorithm, in which two numbers represent the number of patients and the number of physical examination equipment, respectively. The best fitness values for each instance are included in the third column. The subsequent six columns describe the minimum value, average value, maximum value collected by the two algorithms, respectively, while the dev values obtained from the two compared algorithms are provided in the last two columns. From this information in Table 3, ICA_GS obtains a better value in the same running environment.

To determine if the resulting comparisons are significantly different, we performed a multifactor analysis of variance (ANOVA) with results shown in Fig. 9 for the ICA and ICA_GS. This comparison suggests that the improved

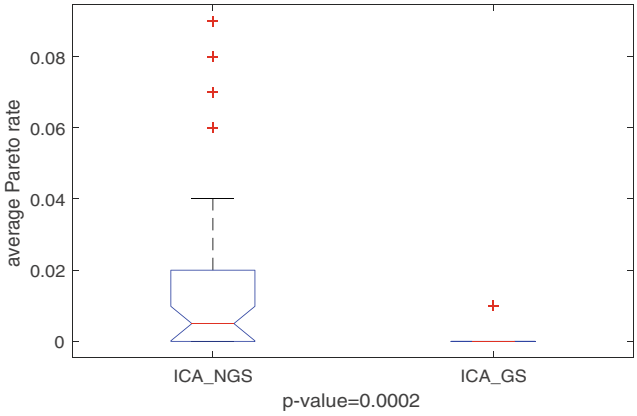


Fig. 11 ANOVA result for strategies with and without global search

algorithm obtains better values. Also, based on the generated convergence curve in Fig. 10, ICA_GS demonstrates a strong convergence ability.

4.3 Effectiveness of the Global Search Strategy

To validate the value of the global search strategy, we conducted detailed comparisons between the algorithm

Table 5 Comparison between CPLEX and ICA_GS

Scale	Best	Algorithm		Dev	
		CPLEX	ICA_GS	CPLEX	ICA_GS
3x2	116.00	119.00	116.00	0.03	**0.00**
3x3	97.00	99.00	97.00	0.02	**0.00**
3x4	121.00	124.00	121.00	0.02	**0.00**
4x2	171.00	175.00	171.00	0.02	**0.00**
4x3	122.00	125.00	122.00	0.02	**0.00**
4x4	124.00	125.00	124.00	0.01	**0.00**
5x2	173.00	181.00	173.00	0.05	**0.00**
5x3	116.00	123.00	116.00	0.06	**0.00**
5x4	135.00	136.00	135.00	0.01	**0.00**
6x2	109.00	114.00	109.00	0.05	**0.00**
6x3	75.00	77.00	75.00	0.03	**0.00**
6x4	123.00	123.00	124.00	**0.00**	0.01
7x2	88.00	88.00	88.00	**0.00**	**0.00**
7x3	108.00	112.00	108.00	0.04	**0.00**
7x4	112.00	112.00	112.00	**0.00**	**0.00**
8x2	152.00	158.00	152.00	0.04	**0.00**
8x3	78.00	80.00	78.00	0.03	**0.00**
8x4	173.00	173.00	173.00	**0.00**	**0.00**
9x2	149.00	154.00	149.00	0.03	**0.00**
9x3	109.00	112.00	109.00	0.03	**0.00**
9x4	133.00	133.00	133.00	**0.00**	0.02
mean	123.05	125.86	123.10	0.02	**0.00**

Table 6 Multi-algorithm comparison of experimental data

Instance	Best	Fitness				Deviation			
		EGA	AIA	MIG	ICA_GS	dev1	dev2	dev3	dev4
Inst1	87.00	93.00	87.00	87.00	87.00	6.90	**0.00**	**0.00**	**0.00**
Inst2	126.00	135.00	126.00	134.00	126.00	7.14	**0.00**	6.35	**0.00**
Inst3	142.00	176.00	154.00	162.00	142.00	23.94	8.45	14.08	**0.00**
Inst4	167.00	197.00	176.00	182.00	167.00	17.96	5.39	8.98	**0.00**
Inst5	221.00	246.00	224.00	236.00	221.00	11.31	1.36	6.79	**0.00**
Inst6	267.00	288.00	267.00	284.00	275.00	7.87	**0.00**	6.37	3.00
Inst7	137.00	137.00	137.00	142.00	138.00	**0.00**	**0.00**	3.65	0.73
Inst8	154.00	176.00	157.00	171.00	154.00	14.29	1.95	11.04	**0.00**
Inst9	213.00	231.00	213.00	229.00	220.00	8.45	**0.00**	7.51	3.29
Inst10	240.00	275.00	253.00	259.00	240.00	14.58	5.42	7.92	**0.00**
Inst11	273.00	294.00	283.00	277.00	273.00	7.69	3.66	1.47	**0.00**
Inst12	306.00	351.00	325.00	313.00	306.00	14.71	6.21	2.29	**0.00**
Inst13	114.00	114.00	115.00	114.00	114.00	**0.00**	0.88	**0.00**	**0.00**
Inst14	165.00	179.00	174.00	170.00	165.00	8.48	5.45	3.03	**0.00**
Inst15	249.00	289.00	267.00	257.00	249.00	16.06	7.23	3.21	**0.00**
Inst16	193.00	227.00	234.00	204.00	193.00	17.62	21.24	5.70	**0.00**
Inst17	328.00	376.00	360.00	338.00	328.00	14.63	9.76	3.05	**0.00**
Inst18	319.00	391.00	352.00	354.00	319.00	22.57	10.34	10.97	**0.00**
Inst19	301.00	308.00	310.00	307.00	301.00	2.33	2.99	1.99	**0.00**
Inst20	279.00	306.00	285.00	284.00	279.00	9.68	2.15	1.79	**0.00**
Inst21	390.00	452.00	439.00	419.00	390.00	15.90	12.56	7.44	**0.00**
Inst22	344.00	385.00	377.00	363.00	344.00	11.92	9.59	5.52	**0.00**
Inst23	439.00	538.00	512.00	496.00	439.00	22.55	16.63	12.98	**0.00**
Inst24	487.00	559.00	544.00	520.00	487.00	14.78	11.70	6.78	**0.00**
Inst25	424.00	443.00	446.00	439.00	424.00	4.48	5.19	3.54	**0.00**
Inst26	467.00	509.00	507.00	484.00	467.00	8.99	8.57	3.64	**0.00**
Inst27	674.00	783.00	714.00	710.00	674.00	16.17	5.93	5.34	**0.00**
Inst28	517.00	611.00	590.00	567.00	517.00	18.18	14.12	9.67	**0.00**
Inst29	654.00	743.00	696.00	673.00	654.00	13.61	6.42	2.91	**0.00**
Inst30	700.00	761.00	741.00	735.00	700.00	8.71	5.86	5.00	**0.00**
Inst31	742.00	770.00	762.00	764.00	742.00	3.77	2.70	2.96	**0.00**
Inst32	731.00	800.00	745.00	755.00	731.00	9.44	1.92	3.28	**0.00**
Inst33	924.00	1012.00	1021.00	1023.00	924.00	9.52	10.50	10.71	**0.00**
Inst34	827.00	899.00	1028.00	865.00	827.00	8.71	24.30	4.59	**0.00**
Inst35	961.00	1062.00	974.00	986.00	961.00	10.51	1.35	2.60	**0.00**
Inst36	1073.00	1193.00	1073.00	1172.00	1073.00	11.18	**0.00**	9.23	**0.00**
Inst37	1547.00	1562.00	1560.00	1555.00	1547.00	0.97	0.84	0.52	**0.00**
Inst38	1513.00	1574.00	1531.00	1565.00	1513.00	4.03	1.19	3.44	**0.00**
Inst39	1991.00	2045.00	2033.00	2085.00	1991.00	2.71	2.11	4.72	**0.00**
Inst40	1578.00	1643.00	1605.00	1649.00	1578.00	4.12	1.71	4.50	**0.00**
Inst41	1833.00	1908.00	1864.00	1926.00	1833.00	4.09	1.69	5.07	**0.00**
Inst42	1847.00	1974.00	1847.00	1974.00	1859.00	6.88	**0.00**	6.88	0.65
mean	593.90	643.21	621.62	624.50	594.57	8.30	5.65	5.42	**0.18**

featuring the global search strategy, ICA_GS, and the one without, ICA_NGS. We recognize that ICA_GS and ICA_NGS have the same content, except for the global search strategy that is performed using a random method. After each algorithm runs independently for 30 s and 30 runs, the comparison results of the algorithm are shown in Table 4. The following observations are seen from Table 4. (1) For solving the given 42 instances, the proposed ICA_GS algorithm obtained 40 optimal values, whereas the ICA_NGS obtained only 21 optimal values. (2) The last row in the table shows that, with a dev value of 0.00, the proposed ICA_GS algorithm performs better than ICA_NGS. A comparison of the two algorithms with ANOVA is presented in Fig. 11, which suggests that the proposed global search strategy enhances the searching capabilities of the proposed algorithm.

4.4 CPLEX verification

For evaluating the performance of ICA_GS, we use the exact solver IBM ILOG CPLEX 12.7.1 to calculate the MIP model. The settings for the precision solver configured as the following. The maximum number of threads is three, and the time limit is set to 3600 seconds. For the ICA_GS, due to its ability to obtain a satisfactory solution within an acceptable time, the maximum CPU time of 30 s is applied as a stop criterion. Twenty-one small-scale examples are tested, with the number of jobs $n \in \{3, 4, 5, 6, 7, 8, 9\}$ and the number of machines $m \in \{2, 3, 4\}$. The experimental results are reported in Table 5, showing that ICA_GS obtained 16 optimal values out of 21 instances representing that ICA_GS can obtain the optimal Pareto solution better than CPLEX.

4.5 Multi-algorithm comparison

In order to verify the advantage of the proposed algorithm in solving the physical examination scheduling problem,

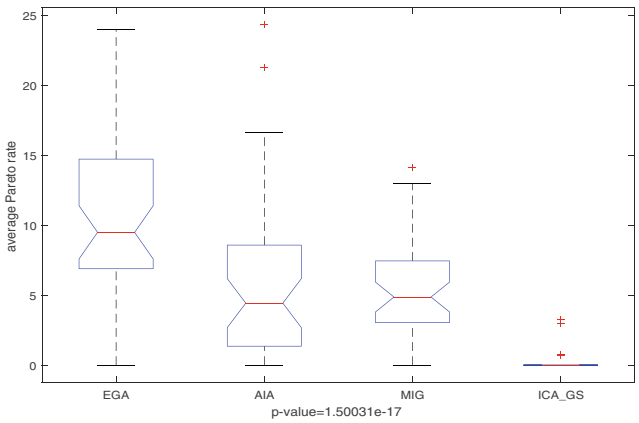
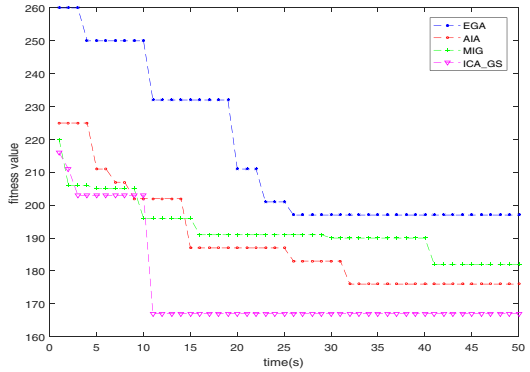
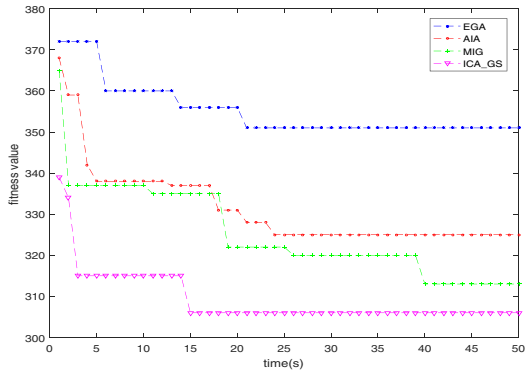


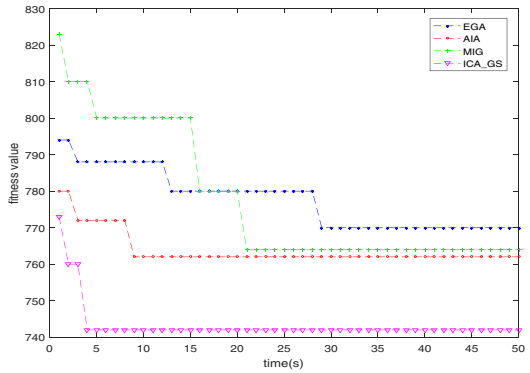
Fig. 12 ANOVA results for comparing multiple algorithms



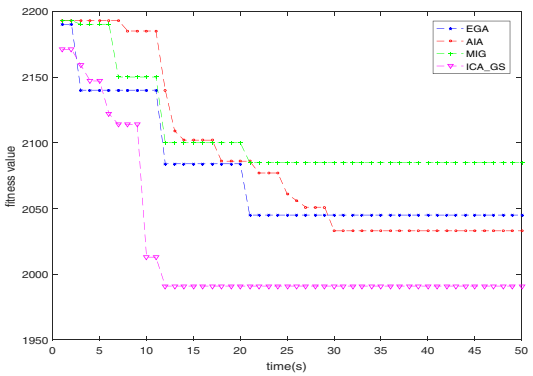
(a) Convergence curve for Inst4.



(b) Convergence curve for Inst12.



(c) Convergence curve for Inst31.



(d) Convergence curve for Inst39.

Fig. 13 Convergence comparisons

we compared the ICA_GS with other current popular algorithms, including enhanced GA (EGA) [37], AIA [38], and modified IG (MIG) [39]. We coded these four algorithms and ran each in the same environment. The instances ran 30 iterations to thousands of iterations, and all comparison algorithms used the same stop criteria. The optimal values obtained from comparing the four algorithms are shown in Table 6, with the instance name in the first column, followed by the best fitness values for each instance in the second column. The best solution of all the comparing algorithms are included in the third column, which is obtained by comparing the optimal values of all algorithms. The subsequent four columns describe the fitness values collected from the four algorithms, respectively. The dev values obtained by the four compared algorithms are listed in the last four columns, respectively.

From Table 6, we make the following observations. (1) For the 42 instances with different problem scales, we use bold to specify the optimal solution obtained, through which we can see that ICA_GS obtained 38 better results, which is better than all others. (2) The last line in the table suggests that, on average, the proposed algorithm obtained a value of 0.18, which is better than the other algorithms. (3) Overall, ICA_GS shows significantly better performance compared with the other four algorithms.

Figure 12 illustrates the ANOVA results for the four compared algorithms suggesting that ICA_GS can obtain the most optimal value. Figure 13 compares the convergence curves of four examples with different problem sizes, and the simulation results demonstrate that the algorithm features good convergence for the problems considered.

5 Conclusion and future research

The outbreak of the novel coronavirus, with its serious infectivity, caused a large number of patients to become sick within a short period. With the condition of limited public resources in hospitals, more patients must receive more efficient treatments within a limited time. Therefore, we introduced a complex flexible job shop scheduling model based on the ICA_GS to solve the physical examination scheduling problem. Considering the processing time of the patient during the physical examination, the transportation time between equipment, and the setup time of the patient. In addition, the local search strategy was embedded into ICA_GS to enrich the searching behaviors along with a global search procedure to enhance the exploration ability. The effectiveness of the algorithm for physical examination scheduling was verified through multiple simulation experiments.

In future work, we will consider the following aspects. (1) In the real world, patients may have special circumstances in the process of physical examination, such as patients whose symptoms are close to the novel coronavirus. In order to minimize the spread of the virus, we need to examine such patients in a timely manner. At this time, it may be necessary to disturb the original scheduling order, which may lead to the reallocation of scheduling space. (2) Consider the obstruction limit and no waiting condition, time and occupation limit, robust buffer, fixed activity and sequence, release time and strict cut-off time during the actual physical examination. (3) Expand the research objectives [40], what we are studying at present is to minimize the completion time of the whole physical examination process, but in the physical examination scheduling in the real world, just minimizing the completion time of the physical examination process can not meet the actual requirements at all. Patients expect to have a physical examination as soon as possible and minimize the completion time of the whole physical examination process. Hospitals expect to minimize the use of medical resources and minimize medical costs, which will be our future research objectives. (4) In the future research, our method will be compared with the current practice to see the applicability of our method and constantly improve our method, such as integrate the algorithm with other algorithms to improve the ability of exploration.

Acknowledgments This research is partially supported by National Science Foundation of China under Grant 61773192.

References

1. Burdett RL, Kozan E (2018) An integrated approach for scheduling health care activities in a hospital. Eur J Oper Res 264(2):756–773
2. Gupta D, Denton B (2008) Appointment scheduling in health care: challenges and opportunities. IIE Trans 40(9):800–819
3. Xiang W, Yin J, Lim G (2015) An ant colony optimization approach for solving an operating room surgery scheduling problem. Comput Ind Eng 85:335–345
4. Hsieh FS (2017) A hybrid and scalable multi-agent approach for patient scheduling based on Petri net models. Appl Intell 47(4):1068–1086
5. Quintana D, Cervantes A, Saez Y, Isasi P (2017) Clustering technique for large-scale home care crew scheduling problems. Appl Intell 47(2):443–455
6. Azaiez MN, Al-Sharif SS (2005) A 0-1 goal programming model for nurse scheduling. Comput Operat Res 32(3):491–507
7. Pham DN, Klinkert A (2008) Surgical case scheduling as a generalized job shop scheduling problem. Eur J Oper Res 185(3):1011–1025
8. Zhu H, Hou M, Wang C, Zhou M (2012) An efficient outpatient scheduling approach. IEEE Trans Automat Sci Eng 9(4):701–709
9. Cappanera P, Visintin F, Banditori C (2014) Comparing resource balancing criteria in master surgical scheduling: a combined optimisation-simulation approach. Int J Prod Econ 158:179–196

10. Gartner D, Kolisch R (2014) Scheduling the hospital-wide flow of elective patients. Eur J Oper Res 233(3):689–699
11. Ahmadi-Javid A, Jalali Z, Klassen KJ (2017) Outpatient appointment systems in healthcare: a review of optimization studies. Eur J Oper Res 258(1):3–34
12. Erhard M, Schoenfelder J, Fügener A, Brunner JO (2018) State of the art in physician scheduling. Eur J Oper Res 265(1):1–18
13. Jiang Y, Abouee-Mehrizi H, Diao Y (2020) Data-driven analytics to support scheduling of multi-priority multi-class patients with wait time targets. Eur J Oper Res 281(3):597–611
14. Li JQ, Tao XR, Jia BX, Han YY, Liu C, Duan P, Sang HY (2020) Efficient multi-objective algorithm for the lot-streaming hybrid flowshop with variable sub-lots. Swarm Evol Comput 52:100600. https://doi.org/10.1016/j.swevo.2019.100600
15. Li JQ, Han YQ, Duan PY, Han YY, Niu B, Li CD, Zheng ZX, Liu YP (2020) Meta-heuristic algorithm for solving vehicle routing problems with time windows and synchronized visit constraints in prefabricated systems. J Clean Prod 2020. https://doi.org/10.1016/j.jclepro.2019.119464
16. Bagheri A, Zandieh M, Mahdavi I, Yazdani M (2010) An artificial immune algorithm for the flexible job-shop scheduling problem. Futur Gener Comput Syst 26(4):533–541
17. Yin J, Xiang W (2012) Ant colony algorithm for surgery scheduling problem. In: International conference in swarm intelligence. Springer, pp 198–205
18. Su MC, Lai SC, Wang PC, Hsieh YZ, Lin SC (2011) A SOMO-based approach to the operating room scheduling problem. Expert Syst Appl 38(12):15447–15454
19. Lee S, Yih Y (2014) Reducing patient-flow delays in surgical suites through determining start-times of surgical cases. Eur J Oper Res 238(2):620–629
20. Behmanesh R, Zandieh M, Molana SH (2019) The surgical case scheduling problem with fuzzy duration time: an ant system algorithm. Scientia Iranica. Transaction E Ind Eng 26(3):1824–1841
21. Behmanesh R, Zandieh M (2019) Surgical case scheduling problem with fuzzy surgery time: an advanced bi-objective ant system approach. Knowl-Based Syst 186:104913
22. Luscombe R, Kozan E (2016) Dynamic resource allocation to improve emergency department efficiency in real time. Eur J Oper Res 255(2):593–603
23. Mirsanei HS, Zandieh M, Moayed MJ, Khabbazi MR (2011) A simulated annealing algorithm approach to hybrid flow shop scheduling with sequence-dependent setup times. J Intell Manuf 22(6):965–978
24. Roshanaei V, Naderi B, Jolai F, Khalili M (2009) A variable neighborhood search for job shop scheduling with set-up times to minimize makespan. Futur Gener Comput Syst 25(6):654–661
25. Saidi-Mehrabad M, Fattahi P (2007) Flexible job shop scheduling with tabu search algorithms. Int J Adv Manuf Tech 32(5-6):563–570
26. Ruiz R, Stützle T (2007) A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European J Oper Res 177(3):2033–2049
27. Pandey S, Wu L, Guru SM, Buyya R (2010) A particle swarm optimization-based heuristic for scheduling workflowapplications in cloud computing environments. In: 2010 24th IEEE international conference on advanced information networking and applications. IEEE, pp 400–407
28. Li JQ, Han YQ (2019) A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system. Clust Comput, pp 1–17. https://doi.org/10.1007/s10586-019-03022-z
29. Li JQ, Deng JW, Li CY, Han YY, Tian J, Zhang B, Wang CG (2020) An improved Jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times. Knowledge-Based Syst. https://doi.org/10.1016/j.knosys.2020.106032
30. Li JQ, Liu ZM, Li C, Zheng Z (2020) Improved artificial immune system algorithm for Type-2 fuzzy flexible job shop scheduling problem. IEEE Trans Fuzzy Syst 2020. https://doi.org/10.1109/TFUZZ.2020.3016225
31. Tao XR, Li JQ, Huang TH, Duan P (2020) Discrete imperialist competitive algorithm for the resource-constrained hybrid flow-shop problem with energy consumption. Complex Intel Syst 2020. https://doi.org/10.1007/s40747-020-00193-w
32. Zhang G, Gao L, Shi Y (2011) An effective genetic algorithm for the flexible job-shop scheduling problem. Expert Syst Appl 38(4):3563–3573
33. Watanabe M, Ida K, Gen M (2005) A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem. Comput Indust Eng 48(4):743–752
34. Lee KM, Yamakawa T, Lee KM (1998) A genetic algorithm for general machine scheduling problems. In: 1998 second international conference. Knowledge-based intelligent electronic systems. Proceedings KES'98 (Cat. No. 98EX111), vol 2, pp 60–66
35. Montgomery DC (2005) Design and analysis of experiments. Wiley, New York
36. Karimi S, Ardalan Z, Naderi B, Mohammadi M (2017) Scheduling flexible job-shops with transportation times: Mathematical models and a hybrid imperialist competitive algorithm. Appl Math Model 41:667–682
37. Dai M, Tang D, Giret A, Salido MA (2019) Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. Robot Comput Integr Manuf 59:143–157
38. Lin SW, Ying KC (2013) Minimizing makespan in a blocking flowshop using a revised artificial immune system algorithm. Omega 41(2):383–389
39. Al Aqel G, Li X, Gao L (2019) A modified iterated greedy algorithm for flexible job shop scheduling problem. Chin J Mech Eng 32(1):21
40. Li J, Du Y, Gao K, Duan P, Gong D, Pan Q, Suganthan P (2020) A hybrid iterated greedy algorithm for a crane transportation flexible job shop problem. IEEE Trans Automat Sci Eng, pp 1–18. In press

Hui Yu received the the B.S. degree from the Institute of engineering, University of Jinan Quancheng College, Penglai, China. She is currently pursuing the M.S. degree in Shandong Normal University. Her major research interest is intelligent optimization and control.

Jun-qing Li received the Master degree of computer science and technology in 2004 from Shandong Economic University, Shandong, China, and Ph.D. degree in 2016 from Northeastern University, Shenyang, China. Since 2004,he was with School of Computer, Liaocheng University. Since 2017, he has been with School of Information Science and Engineering, Shandong Normal University, where he became a Professor in 2017. His current research interests include intelligent optimization and scheduling. He has authored more than 30 refereed papers.