A Hybrid Imperialist Competitive Algorithm for the Outpatient Scheduling Problem with Switching and Preparation Times

Abstract

In recent years, outpatient scheduling problem has attracted much attention. This paper addresses the outpatient scheduling problem as an extension of the flexible job shop scheduling problem (FJSP), where the patient is considered as a job. Then, to solve the outpatient scheduling problem, a hybrid imperialist competitive algorithm (HICA) is proposed. In the proposed algorithm, the simulated annealing (SA) and estimation of distribution algorithm (EDA) are embedded to improve the quality of the solution. Furthermore, the two realistic constraints, i.e., switching time and preparation time of patients are also considered to make the problem more close to the reality. Finally, to verify the performance of the proposed HICA, 35 outpatient scheduling problem instances are randomly generated and used for simulation tests. Four efficient algorithms, including imperialist competitive algorithm (ICA), improved genetic algorithm (IGA), EDA, and modified artificial immune algorithm (MAIA), are selected for detailed comparisons. The simulation results confirm that the proposed algorithm can solve the outpatient scheduling problem with high efficiency.

1 Introduction

Hospitals are critical elements of health care systems and access to their services is very competitive around the world. As the core department of hospital, outpatient service is considered as the "door face" of the hospitals, accounting for a large proportion of hospitals' expenditures and profitability. Outpatient service, however, is very challenging due to its conflicted preferences and priorities from many stakeholders and limited resources to meet the needs (Saremi et al. 2013). In response, scheduling is introduced to optimize outpatient visits.

Scheduling problems have been investigated in recent years and have been utilized in many practical problems (Avalos-Rosales et al. 2013; Frank et al. 2016). This study focuses on the outpatient scheduling problem that consider the switching and preparation times of patients. The seemingly simple task of scheduling patients into the consultation session of a doctor is key in ensuring cost-effective health care and patient satisfaction (Deceuninck et al. 2008). Therefore, effective outpatient scheduling is important for hospitals to treat patients quickly and efficiently and to use their resources wisely. By more efficiently utilizing the existing hospital resources, it may be possible to increase the number of patients a hospital would otherwise treat, or reduce the number of resources that are required to meet given demands (Burdett et al. 2018).

The layout of the problem we study and processing route of patients lends itself well to the flexible job shop scheduling problem. To tackle this outpatient scheduling problem with switching and preparation times, a hybrid imperialist competitive algorithm (HICA) is adopted. In the algorithm, the simulated annealing (SA) and estimation of distribution algorithm (EDA) are embedded to improve the quality of the solution. Due to the intractability of the model on large instances, a position-based mathematical model is adopted and we employ an exact solver IBM ILOG CPLEX 12.7 to calculate the model.

2 Problem Description

To apply a FJSP approach it is necessary to treat patients as jobs, stages of seeking medical treatment and medical resources correspond to operations and machines. There are a set of patients $J=\{J_1, ..., J_n\}$ and a set of medical resources $M=\{M_1, ..., M_m\}$. Each independent patient J_j consists of a sequence of n_j stages of seeking medical treatment $O_{j,q}$, where q = 1, ..., n_j. Each stage can be processed by a set of medical resources in a certain sequence. Then, the two realistic constraints, i.e., the switching time and preparation time of patients are considered. For convenience of description, the following symbols are defined:

Indices:

j,u	Indices of patients
i,k,t	Indices of medical resources
f,s	Indices of positions
q,g	The indices for the stage of seeking medical treatment
n	Number of patients
m	Number of medical resources

Parameters:

n_j	The total number of stages in the process of seeking medical treatment of patient j
$O_{j,q}$	q^{th} stage of seeking medical treatment of patient j
$P_{j,q,i}$	Processing time of $O_{j,q}$ by medical resource i
$S_{j,k,i}$	Switching time of patient j from medical resource k to medical resource i
$E_{j,q,i}$	Binary variable taking value 1 if $O_{j,q}$ is eligible to be processed by medical resource i, and 0 otherwise
$Pr_{i,j}$	Preparation time of the patient j on medical resource i
R_i	Number of stages that can be processed by medical resource i
M_i	Medical resource i
$C_{j,q}$	Completion time of $O_{j,q}$
M	A large positive number

Decision variables:

$W_{j,k,i}$	Binary variable taking value 1 if patient j needs switching time from medical resource k to medical resource i, and 0 otherwise
$X_{j,q,i,f,k}$	Binary variable taking value 1 if $O_{i,q}$ is processed in position f of medical resource i after the processing of patient j by medical resource k

The position-based mathematical model which considers the position of the patient on the medical resource is provided as follows:

Minimize C_{max}

$$\sum_{i=1}^{m}\sum_{f=1}^{R_i}\sum_{k=1}^{m} X_{j,q,i,f,k} = 1 \quad \forall j \in \{1, 2, ..., n\},\ q \in \{2, ..., n_j\} \tag{1}$$

$$\sum_{j=1}^{n}\sum_{q=2}^{n_j}\sum_{k=1}^{m} X_{j,q,i,f,k} + \sum_{j=1}^{n} X_{j,1,i,f,0} \le 1 \quad \forall i \in \{1, 2, ..., m\},\ f \in R_i \tag{2}$$

$$\sum_{f=1}^{R_i}\sum_{k=1}^{m} X_{j,q,i,f,k} \le E_{j,q,i} \quad \forall j \in \{1, 2, ..., n\},\ q \in \{2, ..., n_j\},\ i \in \{1, 2, ..., m\} \tag{3}$$

$$\sum_{f=1}^{R_i} X_{j,q,i,f,k} \le \sum_{f=1}^{R_k}\sum_{t=1}^{m} X_{j,q-1,k,f,t} \quad \forall j \in \{1, 2, ..., n\},\ q \in \{2, ..., n_j\}, i \in \{1, 2, ..., m\},\ k \in \{1, 2, ..., m\} \tag{4}$$

$$\mathrm{C}_{j,q} \ge \mathrm{C}_{j,q-1} + \sum_{i=1}^{m}\sum_{f=1}^{R_i}\sum_{k=1}^{m} X_{j,q,i,f,k}\begin{pmatrix} P_{j,q,i} + pr_{i,j} \\ + \sum_{u=1}^{n} W_{j,k,i} \cdot S_{j,k,i} \end{pmatrix} \quad \forall j \in \{1, 2, ..., n\},\ q \in \{2, ..., n_j\} \tag{5}$$

$$\mathrm{C}_{j,q} \ge \mathrm{C}_{u,g} + P_{j,q,i} - M \cdot \begin{pmatrix} 1 - W_{j,k,i} \\ 2 - \sum_{k=1}^{m} X_{j,q,i,f,k} - \sum_{s=1}^{f-1}\sum_{k=1}^{m} X_{u,g,i,s,k} \end{pmatrix} \forall j \in \{1, 2, ..., n\}, u \in \{1, 2, ..., n\}, j \ne u, i \in \{1, 2, ..., m\},\ q \in \{2, ..., n_j\}, f \in R_i > 1 \tag{6}$$

$$\mathrm{C}_{u,g} \ge \mathrm{C}_{j,q} + P_{u,g,i} - M \cdot \begin{pmatrix} 2 - \sum_{k=1}^{m} X_{j,q,i,f,k} \\ - \sum_{s=1}^{f-1}\sum_{k=1}^{m} X_{u,g,i,s,k} - W_{j,k,i} \end{pmatrix} \forall j \in \{1, 2, ..., n\},\ q \in \{2, ..., n_j\},\ u \in \{1, 2, ..., n\}, j \ne u,\ g \in \{2, ..., n_j\},\ i \in \{1, 2, ..., m\},\ f \in R_i > 1 \tag{7}$$

$$C_{\max} \ge C_{j,n_j} \quad \forall j \in \{1, 2, ..., n\} \tag{8}$$

$$X_{j,q,i,f,k},\ W_{j,k,i} \in \{0,1\} \tag{9}$$

The objective function is to minimize the makespan. Constraint (1) ensures that each stage of seeking medical treatment uniquely occupies one position on one medical resource among all of the available medical resource. Constraint (2) guarantees that each position of each medical resource is occupied once. Constraint (3) ensures that each stage of seeking medical treatment is operated by an available medical resource. Constraint (4) ensures that $O_{j,q-1}$ is processed by medical resource k when $O_{j,q}$ is processed by medical resource i. Constraint (5) is logical precedence constraints that take into account the switching and preparation times of patients. Constraint sets (6)–(7) ensure that each medical resource processes only one stage of seeking medical treatment of patient at a time. Constraint (8) defines the makespan, and constraint (9) defines the values of the decision variables.

3 Hybrid Imperialist Competitive Algorithm

For solving the outpatient scheduling problem in hospital systems, we propose a hybrid ICA, named HICA. The detailed components of HICA are described in this section.

In contrast to other algorithms, the major advantages of the ICA method are the ease of performing neighborhood movement, the less dependency on initial solutions and its reduced calculation time (Atashpaz-Gargari and Lucas 2007; Hosseini and Al Khaled 2014). Therefore, the ICA is chosen as the backbone algorithm. SA is a local search method that finds its inspiration in the physical annealing process studied in statistical mechanics (Bouleimen and Lecocq 2003). While exploring solution space, the SA offers the possibility to accept worse neighbor solutions in a controlled manner in order to escape from local minima. The most important advantage of the SA algorithm compared to the conventional local search-based heuristic methods is that the SA is capable of escaping from being trapped into a local optimum by accepting, in small probability, worse solutions during its iterations (Ekren et al. 2010). EDA is a relatively new paradigm in the field of evolutionary computation. Due to its effectiveness and global search ability, EDA has already been applied to solve combinatorial optimization problems (Pan and Ruiz 2012). After the assimilation strategy of algorithm, SA and EDA are embedded to improve the solutions, under which the global search ability and local search ability can be balanced well. The framework of the HICA is shown in Algorithm 1.

3.1 Fusion of ICA

ICA is a population-based meta-heuristic (Lei et al. 2018). Each individual of population represents a country, and some best countries are selected as imperialists, while the remaining countries are called colonies.

Encoding Scheme Encoding explains how to represent a real solution. In this paper, we use the encoding scheme proposed by Zhang et al. (2011). The solution consists of two parts: medical resource selection (MRS) and the treatment stage sequence (TSS). The MRS part stores the available medical resource number, while the TSS part contains the patient number. Moreover, in the MRS part, the local selection was adopted. In the TSS part, the sequence of all the stages is randomly determined (Figure 1).

Algorithm 1. HICA

```
1  Generate initial populations.
2  Construct the initial empires.
3  While the stopping criterion is not met do
4      for $x$ = 1 to $Nim$ do
5          Mutation operation.
6          If $x$ =1 do
7              Assimilation strategy.
8          else
9              Assimilation strategy change.
10         end
11         Execute the empires invasion strategy.
12         Update the empires.
13         Implement the strategy of the demise of the
           empires.
14     end
15 end
```

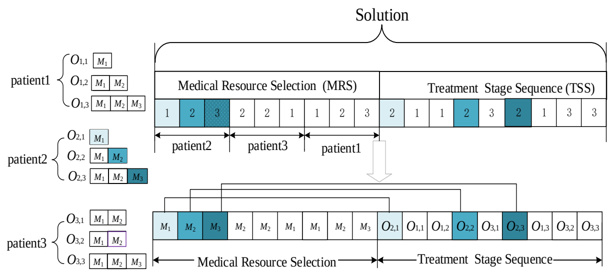

Solution
patient1
$O_{1,1}$ M_1
$O_{1,2}$ M_1 M_2
$O_{1,3}$ M_1 M_2 M_3
patient2
$O_{2,1}$ M_1
$O_{2,2}$ M_1 M_2
$O_{2,3}$ M_1 M_2 M_3
patient3
$O_{3,1}$ M_1 M_2
$O_{3,2}$ M_1 M_2
$O_{3,3}$ M_1 M_2 M_3
Medical Resource Selection (MRS)
Treatment Stage Sequence (TSS)
1 2 3 2 2 1 1 2 3 2 1 1 2 3 2 1 3 3
patient2
patient3
patient1
M_1 M_2 M_3 M_2 M_2 M_1 M_1 M_2 M_3 $O_{2,1}$ $O_{1,1}$ $O_{1,2}$ $O_{2,2}$ $O_{3,1}$ $O_{2,3}$ $O_{1,3}$ $O_{3,2}$ $O_{3,3}$
Medical Resource Selection
Treatment Stage Sequence

Figure 1: Representation of a solution.

Initialization of Empires In the HICA, countries with better objective functions are imperialists and others are the colonies of these imperialists. The population size (*Pop*) includes several imperialists (*Nim*) and their colonies (*Ncl*). The number of colonies in each empire is expressed as NC_x, where x =1, 2, ..., *Nim*.

$$pop = Nim + Ncl \tag{10}$$

The roulette wheel selection procedure is used to calculate the power of each imperialist as

$$power(x) = \frac{1}{makespan(x)} \tag{11}$$

The calculation of the power of the imperialists (*colnum*) can be normalized as follows.

$$colnum(x) = Ncl \cdot \frac{power(x)}{\sum_{n=1}^{Nim} power(x)} \tag{12}$$

Mutation Operation Mutation is a strategy adopted by imperialists to reduce duplication and increase diversity of solution. Through many experiments, it could be found that mutation for solutions with higher fitness often obtain better solutions. Therefore, the probability formula of mutation is described as follows:

$$\eta = \begin{cases} 1 & F \le F_{avg} \\ 1 - \dfrac{F - F_{avg}}{2(F_{avg} - F_{\min})} & F > F_{avg} \end{cases} \tag{13}$$

where η is the individual's mutation probability, F is the individual's fitness, F_{avg} is the average individual's fitness, and F_{min} is the worst individual's fitness. Better individual fitness increases the mutation probability of the individual, enlarges the search space of the individual, and avoids falling into a locally optimal solution. Worse individual fitness reduces the mutation probability of the individual and retains the better individual formed by the mutation.

Diversified Assimilation Strategy Assimilation strategy is the main strategy to produce new solution. Generally, the strategy is implemented in the same way. In this study, the diversified methods are adopted. For empire 1, there are two strategies: (i) all the colonies in the empire adopt the assimilation strategy, (ii) randomly select colonies to perform the assimilation strategy. For the remaining empire x, the assimilation strategy is conducted by learning from any other imperialists.

In the process of assimilation, for the MRS part, the two-point crossover (Watanabe et al. 2005) is adopted. First, two points of the imperialist are randomly selected. Then, the elements between the two points are inserted into the corresponding positions of the new colony, and the remaining positions insert the elements of the old colony. For the TSS part adopts the POX crossover proposed by Lee et al. (1998). First, several numbers are selected at random. Then, the numbers selected from the imperialist are inserted into the new colony, and the remaining numbers are inserted into the new colony according to the position of the old colony.

After assimilation, the SA and EDA are embedded into the algorithm to enhance the performance of the solution, that is, $\lfloor \alpha \times NC_x \rfloor$ good colonies improve through the SA strategy, and $\lfloor \beta \times NC_x \rfloor$ poor colonies improve through the global search of the EDA, where α and β represent the ratio of colonies improved through SA and EDA, respectively. The main steps are displayed in Algorithm 2.

Empires Invasion To gain more powerful power, some empires may conduct empire invasion strategy, that is, the empire randomly occupies the colonies of other empire to change the power.

Updating and Destruction of Empires Through the iterations, colonies may obtain greater power (better objective functions) than their imperialist. In this case, the country with the greatest power becomes the new imperialist, colonies under the old imperialist are assigned to the new imperialist, and the old imperialist becomes a colony of the new imperialist. Then, the worst empire is updated by EDA.

In an empire, colonies under the rule of the imperialist no longer exist. Then, the strategy of empire destruction is implemented, and the imperialist is allocated to other empires as colony.

Algorithm 2. diversified assimilation strategy

Input: all populations
Output: the new solution

```
1   for x = 1 to Nim do
2       if x = 1 do
3           r = rand () % 2.
4           if r = 0 do
5               All colonies in the Empire 1 are chosen for
                assimilation.
6           else
7               Randomly select colonies for assimilation.
8           end
9       end
10      else if x <= Nim do
11          for x = 2 to Nim do
12              Each colony in Empire x can learn from
                other randomly selected imperialist and
                performs assimilation strategy with the
                chosen imperialist.
13          end
14      end
15      $\lfloor \alpha \times NC_x \rfloor$ good colonies improve by SA.
16      $\lfloor \beta \times NC_x \rfloor$ poor colonies improve by EDA.
17  end
```

3.2 Fusion of SA

SA was first introduced by Kirkpatrick et al. (1983) [47] as an intriguing technique for optimizing functions of various variables. It is a heuristic strategy that provides a means for optimization of NP-complete problems for which an exponentially number of steps is required to generate an exact answer.

Application of SA The basic algorithm of SA may be described as follows: Beginning with a high temperature (*T*), a metal is slowly cooled, so that the system is in thermal equilibrium at every stage. At high temperatures, the metal is in liquid phase, and the atoms of the system are disordered and randomly arranged. By gradually cooling the metal, the system becomes more ordered, until it finally reaches a "frozen" ground state (ΔT), where the energy of the system has acquired the globally minimal value.

To overcome the problem of stagnating in local optima, SA utilizes a certain probability to accept a worse solution. The algorithm starts with a randomly generated solution (initial solution); in each iteration, a neighbor solution to the best one so far is generated according two neighborhood structures are randomly selected from the following part and evaluated using a fitness function.

Annealing rate is another important module of the SA. Annealing rate has a significant relation with the acceptance probability of inferior solutions, which can directly affect the speed and accuracy of the SA. The SA algorithm terminates when the temperature recedes to the specified temperature. The annealing rate function is as follows:

$$T(\lambda) = \delta \times T(\lambda\text{-}1) \tag{14}$$

where $\delta \in (0,\ 1)$ is an annealing rate coefficient; $T(\lambda)$ and $T(\lambda-1)$ are current annealing temperature and previous annealing temperature, respectively; λ is the iteration number. The main steps of SA are shown in Algorithm 3.

```
Algorithm 3. SA
Input: good colonies
Output: the optimized solution
1   for $y$ = 1 to the number of good colonies do
2       if $T > \Delta T$ do
3           Execute the neighborhood structure to create a new solution, as discussed in this paper.
4           if the fitness of the new solution < the current best solution do
5               Accept this new solution.
6           else
7               Calculate the probability of accepting the new solution ($rr$).
8               if $rr > 0$ do
9                   Accept the worst solution.
10              end
11          end
12          Update the current temperature by (14).
13      end
14  end
```

Neighborhood Structures The neighborhood structures N_1, N_2, N_3, and N_4 are described below. Neighborhood structure N_1 is performed by replacing a randomly selected position of the MRS part. Neighborhood structure N_2 acts on the TSS part, selecting two elements in a random manner and swapping them. Neighborhood structure N_3 is performed by selecting two positions r_1 and r_2 at random, where $r_1 < r_2$, the element at position r_2 is inserted before r_1 in the MRS part. Neighborhood structure N_4 generates a new solution by performing one swap operation two times in the TSS part (Figure 2).

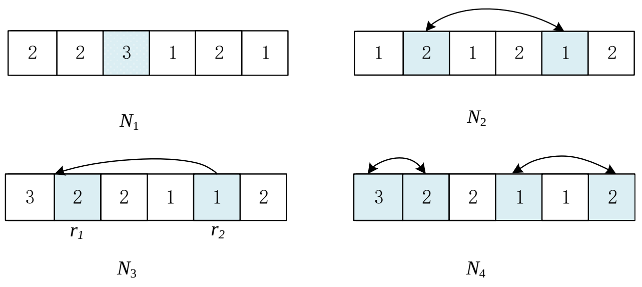


Figure 2: Examples of neighborhood structures.

3.3 Fusion of EDA

EDA is a relatively new paradigm in the field of evolutionary computation, which employs explicit probability distributions in optimization (Larrañaga and Lozano 2001). Instead of using the conventional crossover and mutation operations of regular genetic algorithms, EDA adopts a probabilistic model learned from a population of selected individuals to produce new solutions at each generation. Meanwhile, the probability model is updated in each generation with the potential individuals of the new population. In such an iterative way, the population evolves, and finally satisfactory solutions can be obtained (Wang et al. 2012).

In this paper, two probability matrices P_1 and P_2 are designed for the MRS and TSS parts. The element $M_{j,q,i}(L)$ of the MRS probability matrix P_1 represents the probability of processing stage $O_{j,q}$ is processed by medical resource i in generation L. $m_{j,q}$ represents the array of available medical resources for the q^{th} stage of seeking medical treatment of patient j. The probability matrix P_1 is initialized as follows:

$$M_{j,q,i}(0) = \begin{cases} \dfrac{1}{m_{j,q}} & \text{if } O_{j,q} \text{ can be processed by medical resource } i \\ 0 & \text{else} \end{cases} \tag{15}$$

Element $H_{j,f}(L)$ of TSS probability matrix P_2 represents the probability that patient j will appear at position f in generation L, while Q represents the total number of stages in the TSS part. The probability matrix P_2 is initialized as $H_{j,f}$ (0)=1/Q, which ensures that the entire solution space can be sampled uniformly. And then, the probability matrices P_1 and P_2 are updated according to the following equations:

$$M_{j,q,i}(L+1) = (1-\theta) M_{j,q,i}(L) \tag{16}$$

$$H_{j.f}(L+1) = (1-\sigma) H_{j.f}(L) \tag{17}$$

where $\theta, \sigma \in (0,\ 1)$ are the learning rates of P_1 and P_2 respectively.

Sampling is performed through probabilistic models P_1 and P_2 to generate new individuals. The process of generating new solutions through the EDA is demonstrated in Algorithm 4.

Algorithm 4. EDA

Input: poor colonies
Output: the optimized solution

```
1 While stopping criteria are not satisfied do
2     for $y$ = 1 to the number of poor colonies do
3         A new solution is generated from random sampling of the probabilistic models $P_1$ and $P_2$.
4         Compare the newly generated solution to the previous one.
5         If the new solution is better do
6             Accept this new solution.
7         end
8     end
9 end
```

4. Simulation Analysis

This section evaluates the effectiveness of the proposed HICA on outpatient scheduling problem. All numerical experiments have been conducted on a Lenovo PC with a 3.3-GHz processor and 4-GB memory running Windows 7. The FJSP approach has been coded in C++ to add speed and robustness.

4.1 Simulation Analysis

The algorithm has four important parameters: the country size *pop*, the ratio of the imperialist to countries *Nim*/*pop*, the ratio of good colonies improved through SA α, and the ratio of poor colonies improved through the EDA β. The levels of each parameter are listed in Table 1. To evaluate the influence of these parameters on the performance of the HICA, the DOE (Design of Experiment) Taguchi method (Montgomery 2005) was used, in which an orthogonal array L_{16} is constructed. Then, 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, a factor level trend chart for the four parameters was created based on the obtained data. As seen in Figure 3, when *pop* is at level 2, *Nim/Pop* is at level 1, α is at level 2, and β is at level 1, the proposed algorithm shows the best performance.

Table 1. Key parameter levels.

Parameter	Level			
	1	2	3	4
Pop	50	100	150	200
Nim/ Pop	5%	10%	15%	20%
α	0.2	0.3	0.4	0.5
β	0.1	0.2	0.3	0.4

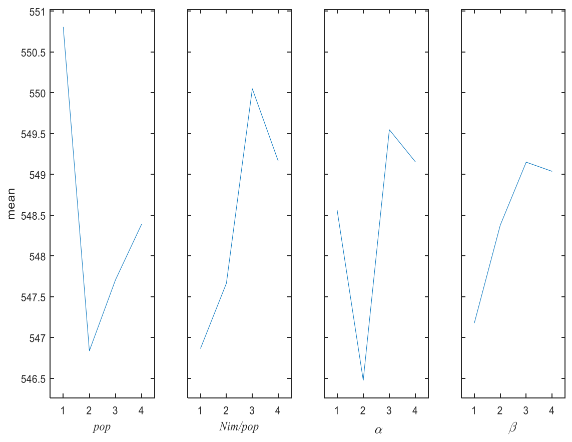


Figure 3: Factor level trends for the four key parameters.

4.2 Performance Comparison with Exact Solver CPLEX

To verify the quality of the algorithm for solving outpatient scheduling problem, this section compares HICA and IBM ILOG CPLEX 12.7 algorithm. The settings for the precision solver were configured as follows. The maximum number of threads was 3, and the time limit was set to 3 h. For the HICA, due to its ability to obtain a satisfactory solution within an acceptable time, a maximum CPU time of 30 s was applied as a stop criterion. Then, 18 small-scale instances of outpatient scheduling problem were randomly generated: the number of patients $n \in \{2, 3, 4, 5\}$, the number of medical resources $m \in \{2, 3\}$, and the number of the stages of seeking medical treatment $q \in \{2, 3, 4\}$.

With an increase in the number of patients and stages, there is an exponential increase in time to solve the problem to optimality on CPLEX because of the NP-hard property (Zhang et al. 2019). Thus, for the other instances, the optimal solutions cannot be obtained on CPLEX within 3h. Table 2 presents the comparison results between the proposed algorithm and the CPLEX solver. The first column lists the instance name, the second column lists the scale size of the algorithm (where *X*-*Y*-*Z* represents *X* patients, *Y* the stages of seeking medical treatment, and *Z* medical resources), and the best fitness values for each instance are listed in the third column. The fitness values for each instance are provided in the fourth and fifth columns, while the sixth and seventh columns provide the deviation of the objective value of each algorithm compared with the best value.

Table 2. Comparison results for the CPLEX and HICA.

Instance	Scale	Best	Fitness		Deviation	
			CPLEX	HICA	CPLEX	HICA
Inst1	2-2-2	78.00	79.00	78.00	0.01	**0.00**
Inst2	2-3-2	105.00	105.00	105.00	**0.00**	**0.00**
Inst3	2-4-2	140.00	141.00	141.00	**0.00**	**0.00**
Inst4	3-2-2	119.00	121.00	119.00	0.02	**0.00**
Inst5	3-3-2	167.00	167.00	167.00	**0.00**	**0.00**
Inst6	3-4-2	182.00	182.00	182.00	**0.00**	**0.00**
Inst7	3-2-3	69.00	69.00	69.00	**0.00**	**0.00**
Inst8	3-3-3	87.00	87.00	87.00	**0.00**	**0.00**
Inst9	3-4-3	129.00	-	129.00	-	**0.00**
Inst10	4-2-2	101.00	101.00	101.00	**0.00**	**0.00**
Inst11	4-3-2	162.00	162.00	162.00	**0.00**	**0.00**
Inst12	4-4-2	231.00	233.00	231.00	0.01	**0.00**
Inst13	4-2-3	63.00	63.00	63.00	**0.00**	**0.00**
Inst14	4-3-3	121.00	-	121.00	-	**0.00**
Inst15	5-2-2	147.00	147.00	147.00	**0.00**	**0.00**
Inst16	5-3-2	238.00	238.00	238.00	**0.00**	**0.00**
Inst17	5-4-2	309.00	-	309.00	-	**0.00**
Inst18	5-2-3	86.00	-	86.00	-	**0.00**

The following observations can be made from Table 1. (i) For solving the given 18 instances, HICA obtained higher quality solutions than the CPLEX solver. (ii) CPLEX did not solve large-scale examples better than HICA.

4.3 Evaluation of Strategy Effectiveness

Three variants of HICA were constructed. HICA1 did not include the SA strategy, but the remainder of the content was consistent with the HICA. The difference between HICA2 and HICA was that the EDA was eliminated from HICA. HICA3 was similar to HICA except for the empires invasion strategy. The parameters were set, and the running environment was the same as for HICA. Each calculation instance is executed 30 times independently, each for 30 s.

To determine whether the resulting comparisons were significantly different, we performed a multifactor analysis of variance (ANOVA) (Li et al. 2020). The confidence level was set to 95%. When the p-value was less than 0.05, the difference between the algorithms was significant, and Figure 4 indicates that our strategy is effective.

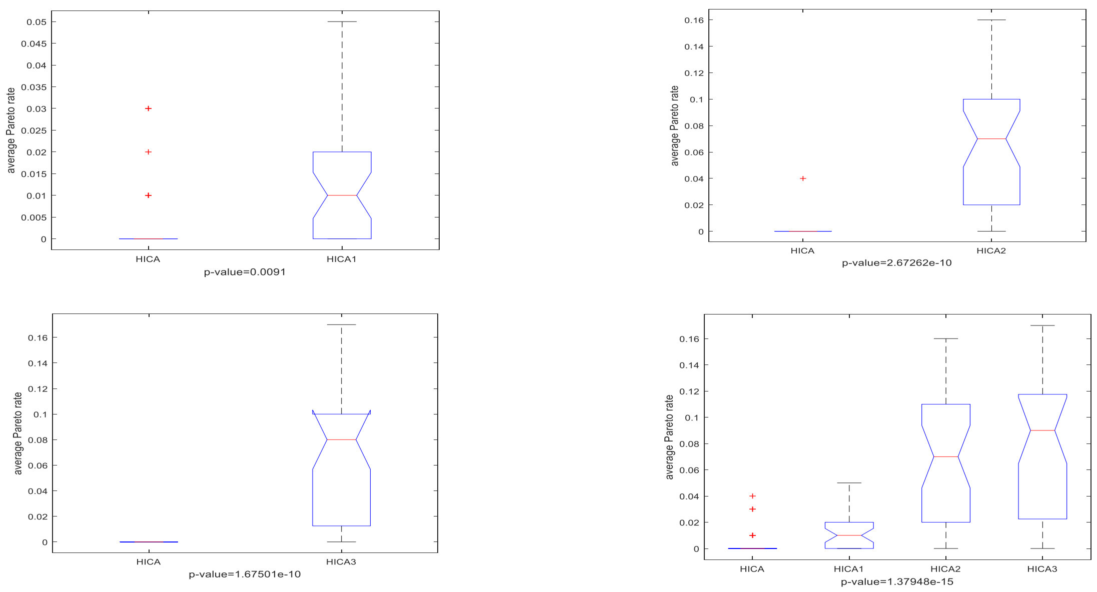


Figure 4: ANOVA results of HICA and its variants.

4.4 Comparison with other Algorithms

To further evaluate the performance of the proposed HICA, we selected the following algorithms for comparison: ICA (Karimi et al. 2017), improved GA (IGA) (Zhang et al. 2020), EDA (Zhao et al. 2016), and modified AIA (MAIA) (Zeng and Wang 2018). For each compared algorithm, the results obtained after 30 independently run are used to make a detailed comparison.

Figure 5 presents the ANOVA results of the average solutions of the five comparison algorithms, from which it can be conclude that the HICA has better performance in solving the problems studied. Furthermore, to demonstrate the convergence ability of the proposed algorithm, we conducted detailed comparisons between the HICA and other four algorithms, and the convergence curves for the four instances with different features are depicted in Figure 6 (a)-(d).

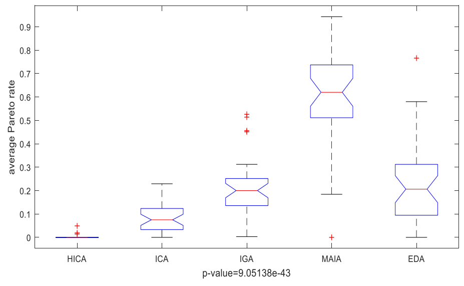


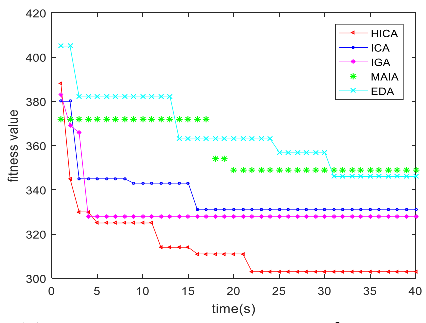
Figure 5: Comparison of the average solutions of multiple algorithms

5 Conclusion and Future Research

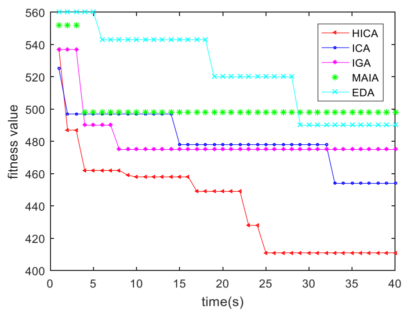
This study addresses the outpatient scheduling problem with the switching time and preparation time. The problem is the extension of the FJSP, where patients are treated as jobs, the stages of seeking medical treatment and the medical resources correspond to the operations and machines. To solve the problem, the HICA was proposed, in which SA and EDA are embedded to improve the solution quality. In proposed algorithm, first, the empires invasion strategy is proposed to alter the power of the empires, and therefore the population diversity capabilities can be enhanced. Then, to increase the searching abilities of the proposed algorithm, four neighborhood structures are embedded in the SA component to generate new promising solutions. Finally, two different probabilistic models are embedded in the EDA component to enhance the global search abilities.

We simulate the outpatient scheduling process and verify the effectiveness of the algorithm through experiment. In the experiment, first, the Taguchi method was used to evaluate the parameters of the algorithm. Then, to verify the quality of the algorithm for solving outpatient scheduling problem, the comparison of HICA and IBM ILOG CPLEX 12.7 algorithm was executed. Furthermore, the effectiveness of the algorithm strategy was verified. Finally, to verify the performance of our algorithm, four efficient algorithms, the ICA, IGA, EDA, and AIA are selected to conduct detailed simulation tests.

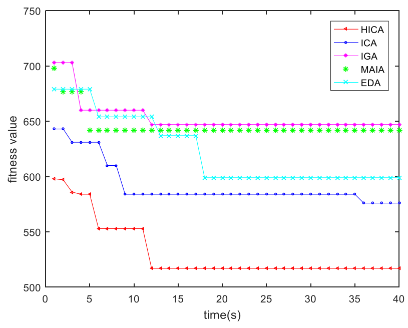
In future work, we will focus on the following issues. (i) Add other practical constraints or consider optimizing multiple objectives in the outpatient scheduling problem. (ii) Consider the insertion and handling of acute patients. (ii) Improve the balance between the algorithm's development and exploration abilities.



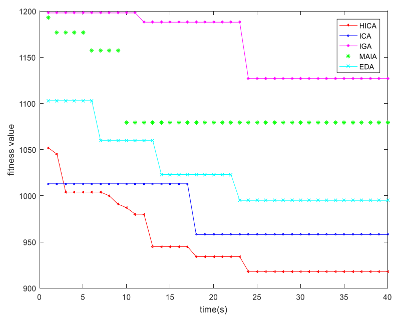
(a) Convergence comparison for Inst10



(b) Convergence comparison for Inst19



(c) Convergence comparison for Inst23



(d) Convergence curve for Inst 29

Figure 6: Convergence comparison between multiple algorithms.

References

Saremi, A.; Jula, P.; ElMekkawy, T.; and Wang, G. G. 2013. Appointment scheduling of outpatient surgical services in a multistage operating room department. *International Journal of Production Economics* 141(2): 646-658.

Avalos-Rosales, O.; Alvarez, A.; and Angel-Bello, F. 2013. A reformulation for the problem of scheduling unrelated parallel machines with sequence and machine dependent setup times. In *Proc. of the Twenty-Third International Conference on International Conference on Automated Planning and Scheduling* (*ICAPS*), 278-282

Frank, J.; Do, M.; and Tran, T. T. 2016. Scheduling ocean color observations for a GEO-stationary satellite. In *Proc. of the Twenty-Sixth International Conference on International Conference on Automated Planning and Scheduling (ICAPS)*, 376-384.

Deceuninck, M.; Fiems, D.; and De Vuyst, S. 2018. Outpatient scheduling with unpunctual patients and no-shows. *European Journal of Operational Research* 265(1): 195-207.

Burdett, R. L.; and Kozan, E. 2018. An integrated approach for scheduling health care activities in a hospital. *European Journal of Operational Research* 264(2): 756-773.

Atashpaz-Gargari, E.; and Lucas, C. 2007. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In *IEEE congress on evolutionary computation*.

Hosseini, S.; and Al Khaled, A. 2014. A survey on the imperialist competitive algorithm metaheuristic: implementation in engineering domain and directions for future research. *Applied Soft Computing* 24:1078-1094.

Bouleimen, K. L. E. I. N.; and Lecocq, H. O. U. S. N. I. 2003. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. *European journal of operational research* 149(2):268-281.

Ekren, O.; and Ekren, B. Y. 2010. Size optimization of a PV/wind hybrid energy conversion system with battery storage using simulated annealing. *Applied energy* 87(2): 592-598.

Pan, Q. K.; and Ruiz, R. 2012. An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. *Omega* 40(2):166-180.

Lei, D.; Li, M.; and Wang, L. 2018. A two-phase meta-heuristic for multiobjective flexible job shop scheduling problem with total energy consumption threshold. *IEEE Transactions on Cybernetics* 49(3):1097-1109.

Zhang, G.; Gao, L.; and Shi, Y. 2011. An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications* 38(4):3563-3573.

Watanabe, M.; Ida, K.; and Gen, M. 2005. A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem. *Computers & Industrial Engineering* 48(4):743-752.

Lee, K. M.; Yamakawa, T., and Lee, K. M. 1998. A genetic algorithm for general machine scheduling problems. *Knowledge-Based Intelligent Electronic Systems* 2:60-66. IEEE.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. 1983. Optimization by simulated annealing. *science* 220(4598): 671-680.

Larrañaga, P.; and Lozano, J. A. (Eds.). 2001. Estimation of distribution algorithms: A new tool for evolutionary computation (Vol. 2). *Springer Science & Business Media*.

Wang, L.; Wang, S.; Xu, Y.; Zhou, G.; and Liu, M. 2012. A bi-population based estimation of distribution algorithm for the flexible job-shop scheduling problem. *Computers & Industrial Engineering* 62(4):917-926.

Montgomery, D.C. 2005. Design and analysis of experiments, *John Wiley & Sons*, Arizona.

Zhang, B.; Pan, Q. K.; Gao, L.; Meng, L. L.; Li, X. Y.; & Peng, K. K. 2019. A three-stage multiobjective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Li, J. Q.; Han, Y. Q.; Duan, P. Y.; Han, Y. Y.; Niu, B.; Li, C. D.; and Liu, Y. P. (2020). Meta-heuristic algorithm for solving vehicle routing problems with time windows and synchronized visit constraints in prefabricated systems. *Journal of Cleaner Production* 250:119464.

Karimi, S.; Ardalan, Z.; Naderi, B.; and Mohammadi, M. 2017. Scheduling flexible job-shops with transportation times: Mathematical models and a hybrid imperialist competitive algorithm. *Applied mathematical modelling* 41: 667-682.

Zhang, G.; Hu, Y.; Sun, J.; and Zhang, W. 2020. An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints. *Swarm and Evolutionary Computation* 54:100664.

Zhao, F.; Shao, Z.; Wang, J.; and Zhang, C. 2016. A hybrid differential evolution and estimation of distribution algorithm based on neighbourhood search for job shop scheduling problems. *International Journal of Production Research* 54(4):1039-1060.

Zeng, R.; and Wang, Y. 2018. A chaotic simulated annealing and particle swarm improved artificial immune algorithm for flexible job shop scheduling problem. *EURASIP Journal on Wireless Communications and Networing* 2018(1):101.