Journal of Combinatorial Optimization (2020) 39:547–580
https://doi.org/10.1007/s10878-019-00497-9

# A patient flow scheduling problem in ophthalmology clinic solved by the hybrid EDA–VNS algorithm

**Wenjuan Fan[1,2] · Yi Wang[1,2] · Tongzhu Liu[3] · Guixian Tong[3]**

Published online: 7 December 2019


## Abstract

This paper studies the patient flow scheduling problem in a multi-phase-multi-server system setting for a typical ophthalmology clinic, considering different patient flow processes and specific appointment time. In this problem, patients may go through the following processes, i.e., consultation, examination, re-consultation, and treatment, which form four patient flow paths according to different situations. The objective of this paper is to minimize the completion time of all the patients in the ophthalmology clinic. For solving this problem, we develop a hybrid meta-heuristic algorithm EDA–VNS combining estimation of distribution algorithm (EDA) and variable neighborhood search (VNS). We test the suitability of the approach for the ophthalmology clinic's problem. Computational results demonstrate that the proposed algorithm is capable of providing high-quality solutions within a reasonable computational time. In addition, the proposed algorithm is also compared with several high-performing algorithms to validate its efficiency. The results indicate the advantages of the proposed EDA–VNS algorithm.

**Keywords** Patient flow scheduling · Ophthalmology clinic · Appointment system · EDA–VNS · Patient-sequence rules

✉ Wenjuan Fan
fanwenjuan@hfut.edu.cn

Yi Wang
wy071008@gmail.com

Tongzhu Liu
aqwsltz@163.com

Guixian Tong
381378037@qq.com

1 School of Management, Hefei University of Technology, Hefei 23009, Anhui, China

2 Key Laboratory of Process Optimization and Intelligent Decision-Making (Ministry of Education), Hefei University of Technology, Hefei 23009, Anhui, China

3 The First Affiliated Hospital of University of Science and Technology of China, Hefei 23009, China

## 1 Introduction

Recently, numerous studies have been carried out to improve the efficiency of patient flow scheduling in clinics to reduce expenses and enhance patient satisfaction. Particularly, the number of ambulatory surgeries in ophthalmology clinics is increasing, which results in the need of solving patient flow scheduling problem.

Motivated by the above development of ophthalmology clinic, we study the patient flow scheduling problem in typical ophthalmology clinic. Scheduling patients in ophthalmic clinic can not only make better use of ophthalmic medical resources but also have a good guiding role for other similar special clinics. In this class of problem, there are mainly four processes for the patients, i.e., consultation, examination, re-consultation, and treatment, while different patients may go through different combination of process, and each possible combination of services is regarded as a path. Usually in these clinics, the patient's consultation time is shorter than the patient's examination time and treatment time, and if there is an examination phase, the patient needs to return to the original consulting doctor for a second consultation, which we call re-consultation.

The patients in an ophthalmology clinic system should have an appointment and thus the route of each patient can be determined in advance. The appointment scheduling (AS) is usually used in the medical system, based on which the patient scheduling on the operational level in the ophthalmology clinic can be improved. The operational level means arranging each patient or service to optimize appointment scheduling, including the acceptance or rejection of patients, the determination of appointment date or time, the allocation and sorting of patients (Ahmadi-Javid et al. 2017). Chakraborty et al. (2013) proposed an appointment scheduling algorithm to select the appointment time of patients by considering minimizing the total expected cost, and determined the only criterion to stop accepting the patient's request for an appointment. Salemi Parizi and Ghate (2016) investigated the possibility that dynamic and random arrival reservation appointment requests may be rejected or accepted. Lu et al. (2018) considered assigning appointment date or time for elective inpatients that wait for admission to hospital to improve patient waiting. Yan et al. (2015) considered the optimal number of patients and the appointment time of patients to maximize the profit of the outpatient clinic. Li et al. (2018) linked patient satisfaction with patients' preferences of physicians and time blocks in allocation. Besides, patient satisfaction and future request are considered to optimize the appointment system. Lin (2015) considered the sequence of patients and compared the proposed method with other common scheduling rules. This paper considers the allocation and sequence of patients. The appointment time of patients are randomly generated. The traditional appointment scheduling problem focuses on a single server, and only a few consider multi-server or multi-resource joint scheduling. By reviewing the recent papers related to patient flow, we can classify them into two types: (1) to refine the patient flow itself to improve the whole system; (2) to schedule the patients and allocate the resource properly to minimize the completion time of the total system, the waiting time of patients and so on. Different from the traditional appointment scheduling problem, this problem concentrates on whole patient flow rather than a single server.

The first type of research concentrates on the irrationality of patient flow. By improving the irrationality of patient flow, medical expenses and patient waiting time can be reduced. The study by Sayah et al. (2016) compared the impacts of improving the patient flow and expanding the emergency department. Chen et al. (2016) found that changing the order of procedures of patient flow and the idle consultants for new patients also serve the follow-up patients can reduce the waiting time of patients. Armony et al. (2015) applied an exploratory data analysis method to study the performance of patient flow and find the significant features of patient flow. However, rather than improving patient flow itself, we intend to discuss the allocation and sequencing of patients in different paths. Therefore, while ensuring that patients with different paths can be represented, we only remain key procedures to simplify the problem and obtain an abstract patient flow model which is adapted to most clinics under different constraints.

There are more papers about the second type. Simulation is widely used in the complex environment to schedule the patients under numerous constraints. Saremi et al. (2013) investigated a general patient flow in a clinic. They used analytical approaches and simulation to search the near optimal solution. Some researches focus on special clinics rather than the general patient flow. Liang et al. (2015) studied the patient flow of chemotherapy patients in an outpatient clinic. They established a mathematical programming model to generate a balanced schedule and proposed a discrete event simulation model. There are also some other approaches to solve the patient flow problem. For example, queuing theory, integer programming and intelligent algorithm. Huang et al. (2015) employed the queuing theory and used the feedback method to solve the queuing system problems with various types of patients. Leeftink et al. (2017) proposed a sample average approximation to solve a stochastic integer program which considering the uncertainty of patient path. Some heuristic algorithms are also developed to deal with this problem. Chern et al. (2008) took the characteristics of health examination process into account, and proposed health examination scheduling algorithm (HESA), a heuristic algorithm to solve the health examination scheduling problem pertinently. Lin (2015) designed an adaptive scheduling heuristic method to make full use of the waiting time information. Then the initial scheduling iteration can be improved by identifying the steps with larger average waiting time and reallocating the related patient categories to less congested time blocks with a certain probability.

In this paper, because of the possible re-consultation process, we also consider an important characteristic in this problem, i.e., the re-entrant patient flow scheduling. The re-consultation stage is like the re-entrant stage in flow shop and job shop. The difference between re-entrant job shop scheduling problem and re-entrant flow shop scheduling problem is whether the jobs' procedures follow the same path (Danping and Lee 2011). Choi and Kim (2008) considered the re-entrant flow shop scheduling problem which the jobs enter the production line several times. Kim and Lee (2009) investigated the re-entrant flow shop scheduling problem with unrelated parallel machines and developed two heuristic algorithms. Shen et al. (2016) proposed a novel decoding method to handle with the re-entrant flow shop scheduling problem. The jobs are processed on machines repeatedly and the re-entrant times of each job are known in advance. There are also some re-entrant scheduling problems in which the jobs aren't processed on each stage. Zhang and Chen (2018) addressed a

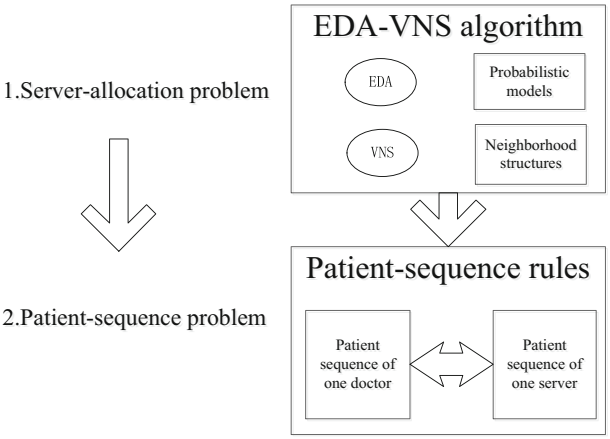



**Fig. 1** Sub-problems and methods

rotor workshop re-entrant scheduling problem which only the first stage is repeated. Zhou et al. (2018) focused on a re-entrant hybrid flow shop scheduling problem in which the inspection station and repair station are re-entrant stages. The re-entrant stage of this problem is only the consultation stage. In this paper, we study the key procedure re-consultation. In practice, patients always jump a queue when they have re-consultation with doctors on the same day, and thus the new patients have to wait for a longer time. We try to figure out how to arrange the sequence between re-consultation patients and consultation patients so as to minimize the total completion time. The main contributions of this study are summarized as follows.

1. The patient flow scheduling problem in ophthalmology is studied in this paper. To the best of our knowledge, this type of problem is rarely discussed.
2. We consider re-consultation of patients in our studied problem, while a few papers have studied the law of re-consultation arrangement.
3. A hybrid meta-heuristic algorithm EDA–VNS is presented to solve the problem and the effectiveness of the proposed algorithm is proved by experiments.

The problem can be divided into two sub-problems, server-allocation problem and patient-sequence problem. The EDA–VNS algorithm is applied to solve the server-allocation problem and generate the solution. Then we prove some structural features of the patient sequence problem, based on which some rules are proposed to determine the patient sequence on the servers. Figure 1 shows the main process for solving the problem.

The remainder sections of this paper are as follows: In Sect. 2 we provide a detail description of our studied problem and give the model. The method is developed in Sect. 3. The patient sequence rules are presented in Sect. 4. The experimental results and the comparison with other completive algorithms are presented in Sect. 5. In Sect. 6 we raise the conclusion and future research.

## 2 Problem description and modeling

In this section, the patient flow scheduling problem description in typical ophthalmology clinics is addressed, and then the problem modeling is presented. The problem modeling includes notations and mathematical model which are used to present the objective of minimizes the sum completion time of all patients.

### 2.1 Problem description

This work investigates the patient flow scheduling problem in typical ophthalmology clinics, which provide the following medical services: consultation, examination, re-consultation and treatment. In each service, the resource is considered as one server and can only serve one patient at the same time. For example, in the consultation service, each doctor is a server, and a doctor can only accept one patient at the same time, as well as a surgical team or an operating, etc.

The patients are divided into new patients and revisit patients in terms of the number of times to visit the clinic. The new patients need to go through examination service first, after which they should take the results back to the doctor for further consultation. The revisit patients already have been examined some day before the consultation day, thus they don't have to do examination for the second time. Different service combinations naturally form different patient paths. Figure 2 shows the four paths of the patients. The first path only has the consultation service in the path, the second path has consultation and treatment services for revisit patients who have appointment for treatment, the third path adds examination and re-consultation after consultation compared with the first path, and the fourth path has four successive services: consultation, examination, re-consultation and treatment services. We have the following assumptions for this problem. (1) The patients should make appointment with doctors for a specific consultation time and all patients need to be consulted first. (2) Patients who need examination must have re-consultation, because the examination results need to be given to doctor for further treatment decision. (3) If the patient should have re-consultation, he/she must go back to the same appointed doctor of consultation. Although consultation and re-consultation are two different procedures, they are essentially the same. In this regard, the patients who re-consult the previous doctor can be regarded as new arrival patients. Because we assume that patients who have examination must have re-consultation, the process of patients in path 3 or 4 can be divided into two stage, i.e., from consultation to examination, and from re-consultation to release. Since consultation and re-consultation can be regarded as the same processing, then the four patient paths can be further classified into three types with the combination of consultation and re-consultation, which can be shown in Fig. 3. All patients will go through one or two stages, including consultation (extracted from path 1 and 3), or consultation and examination (extracted from path 3 and 4), or consultation and treatment (extracted from path 2 and 4). The one-stage patients who only go through consultation can be also seen as two-stage patients with a virtual second stage.

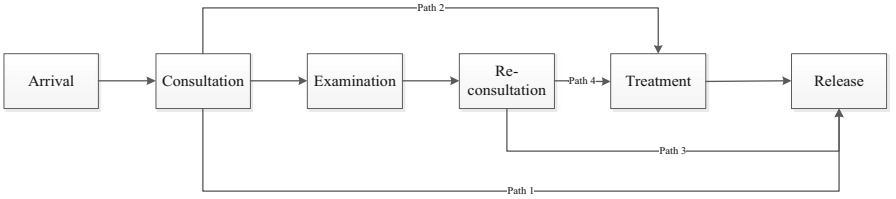



**Fig. 2** The four paths of patient flow

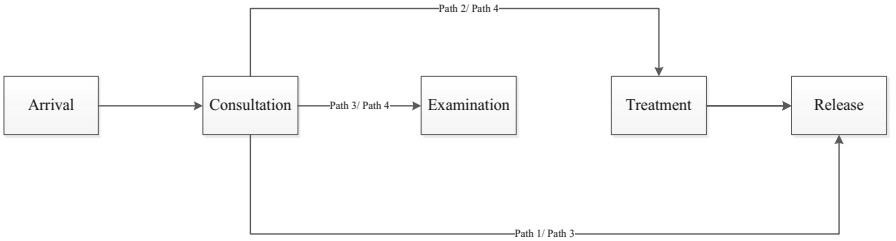



**Fig. 3** The patient flow after amalgamating consultation and re-consultation

Since we assume that patients make an appointment with doctors in advance, and thus the patients' paths and appointment time can be confirmed at the time of appointment. The first come first serve rule which is commonly adopted in clinics is not applied in this problem. We compare the patients in pairs according to their arrival time to decide the sequence of consultation. Our objective function is to minimize the completion time of all patients. The assumptions of this problem are summarized as follows:

1. Each patient makes an appointment with a doctor and the emergency patients and walk-in patients are not considered in this problem.
2. Patients must not be interrupted when they receive services at a server.
3. The duration for patients to receive each service is known in advance.
4. The patients arrive on time; no patient comes late or breaks the appointment.

## 2.2 Notations

The notation of the mathematic model is presented as follows:

Notation

| | |
|---|---|
| $N$ | Number of patients |
| $i$ | Patient index, $i = 1, \ldots, N$ |
| $M_e$ | Number of examination servers |
| $M_t$ | Number of treatment servers |
| $N_p$ | Number of patients in path $p$, $p = 1, 2, 3, 4$ |
| $P_p$ | Set of patients in path $p$, $p = 1, 2, 3, 4$ |
| $j$ | Server index, $j = 1, \ldots, M_e + M_t$ |
| $L_i$ | Number of consultations of patient $i$, $i = 1, \ldots, N$, $L_i \in \{1, 2\}$ |
| $A_{i,l}$ | The arrival time of patient $i$ for $l$ th consultation, $i = 1, \ldots, N, l = 1, \ldots, L_i$ |
| $T_{i,l}$ | The available time of patient $i$'s doctor for $l$ th consultation, $i = 1, \ldots, N, l = 1, \ldots, L_i$ |
| $X_{i,l}$ | The available time of patient $i$'s server for $l$ th consultation, $i = 1, \ldots, N, l = 1, \ldots, L_i$ |
| $d_{i,s,l}$ | The service duration of patient $i$ on stage $s$ for $l$ th consultation, $i = 1, \ldots, N, s = 1, 2$, $l = 1, \ldots, L_i$ |
| $E_{i,l}$ | The completion consultation time of patient $i$ for $l$ th consultation, $i = 1, \ldots, N$, $l = 1, \ldots, L_i$ |
| $C_{i,l}$ | The completion time of patient $i$ for $l$ th consultation, $i = 1, \ldots, N, l = 1, \ldots, L_i$ |
| $m_{ij}(g)$ | The selection probability that patient $i$ is served on server $j$ on generation $g$ |
| $\theta$ | Proportion of elite individuals, $\theta \in (0, 1)$ |
| $\varphi$ | learning rate, $\varphi \in (0, 1)$ |
| $pop$ | Scale of population |

Decision variable

| | |
|---|---|
| $I_{ij}$ | $= 1$, if server $j$ is selected to serve patient $i$, $i = 1, \ldots, N$, $j = 1, \ldots, M_e + M_t$ |

### 2.3 Mathematic model

This problem is formulated as follows.

$$Minimize \sum_{i=1}^{N} C_{i,L_i}, \quad i = 1, \ldots, N \tag{1}$$

subject to:

$$\sum_{p=1}^{4} N_p = N \tag{2}$$

$$P_1 = \{1, \ldots, N_1\} \tag{3}$$

$$P_p = \left\{ \sum_{1}^{p-1} N_{p-1} + 1, \ldots, \sum_{1}^{p} N_p \right\}, \quad p = 2, 3, 4 \tag{4}$$

$$E_{i,l} = max\left(A_{i,l}, T_{i,l}\right) + d_{i,1,l}, \quad i = 1, \ldots, N, \quad l = 1, \ldots, L_i \tag{5}$$

$$C_{i,L_i} = max\left(E_{i,L_i}, X_{i,L_i}\right) + d_{i,2,L_i}, \quad i = 1, \ldots, N \tag{6}$$

Objective function (1) minimizes the sum completion time of all patients. Constraint (2) is the quantitative constraint of patients which guarantees each patient belongs to a path. Constraints (3, 4) ensure the index of each patient is only one. Constraint (5) defines the end consultation time of each patient is the sum of start consultation time and the length of consultation. The start consultation time is the larger value of patient arrival time and doctor available time. Constraint (6) demonstrates the completion time of each patient is the completion time of stage 2 for the last consultation. After last consultation, patients go to stage 2 to have service then leave the clinic. The start time of stage 2 is the larger value of the end time of last consultation and the server available time.

## 3 EDA–VNS algorithm for sever-allocation

The hybrid EDA–VNS algorithm is presented to solve the server-allocation problem. Server-allocation problem is to determine which server the patient is served on. We propose the framework of EDA–VNS algorithm, the probabilistic models of EDA, the encoding and decoding schemes and three neighborhood structures of VNS algorithm to generate the server allocation solutions. Especially, we give a modified neighborhood structure of VNS algorithm which can modify the solution according to the probabilistic models of EDA.

### 3.1 Framework of EDA–VNS algorithm

The patients' examination and treatment servers are not dedicated servers. Changing the patient's server assignment may affect the value of the objective function. We propose a hybrid EDA–VNS algorithm to solve the server-allocation problem. Estimation of distribution algorithm (EDA) is one of the evolutionary algorithms and based on statistics theory (Larranaga and Lozano 2001). We choose EDA because uses statistical learning to improve the search space and sample the search space to predict and generate new individuals, of which the evolution is on the macro-level and furthermore, it has stronger global search ability and faster convergence speed. The EDA algorithm is extensively used in flow shop and job shop scheduling problem. Wang et al. (2013) used the EDA algorithm to deal with the complex job shop problem with fuzzy process time and proved the effectiveness of EDA. Zhou et al. (2018) combined EDA algorithm and differential evolution algorithm (DE) to solve the flow shop scheduling problem with a partly re-entrant system. However, it also may easily fall into local optimum. In order to avoid it, EDA is combined with variable neighborhood search (VNS) algorithm (Hansen and Mladenović 2001), which is an improved local search algorithm. The VNS algorithm tends to perform well when combined with other algorithms. There are a number of effective hybrid algorithms which combines

**Procedure:** EDA-VNS algorithm

**Input:** patient's path, patient arrival time and other initial settings

**Output:** the best allocation solution

**Start**

Establish initial probabilistic models;

Generate initial population;

Evaluate solutions by decoding routine;

**While** $t \leq maxIter$ **do**

Select elite solutions;

Further search for solutions by VNS;

Renew the probabilistic models;

Generate new population;

$t = t + 1$

**Output** best allocation solution

**End**

**Fig. 4** The Pseudocode of EDA–VNS algorithm

the VNS algorithm for solving the scheduling problem (Liu et al. 2018; Pei et al. 2018, 2019). Figure 4 represents the pseudocode of EDA–VNS algorithm and Fig. 5 shows the flow chart.

### 3.2 The probabilistic models of EDA

In this section, we introduce the probabilistic models of EDA in our problem. The probabilistic models are proposed to generate the server allocation solutions, including examination server allocation and treatment server allocation respectively. The probabilistic model for examination server allocation is a $(N_3 + N_4) \times M_e$ matrix, where $(N_3 + N_4)$ represents the total number of path 3 and path 4 patients, and $M_e$ represents the number of examination servers. The probabilistic model for treatment server allocation is a $(N_2 + N_4) \times M_t$ matrix, where $(N_2 + N_4)$ represents the total number of path 2 and path 4 patients, and $M_t$ represents the number of treatment servers. The element $m_{ij}(g)$ of server allocation probability matrix $M_{(e+t)}$ represents the probability that patient $i$ is allocated to server $j$ in the $g$ th generation.

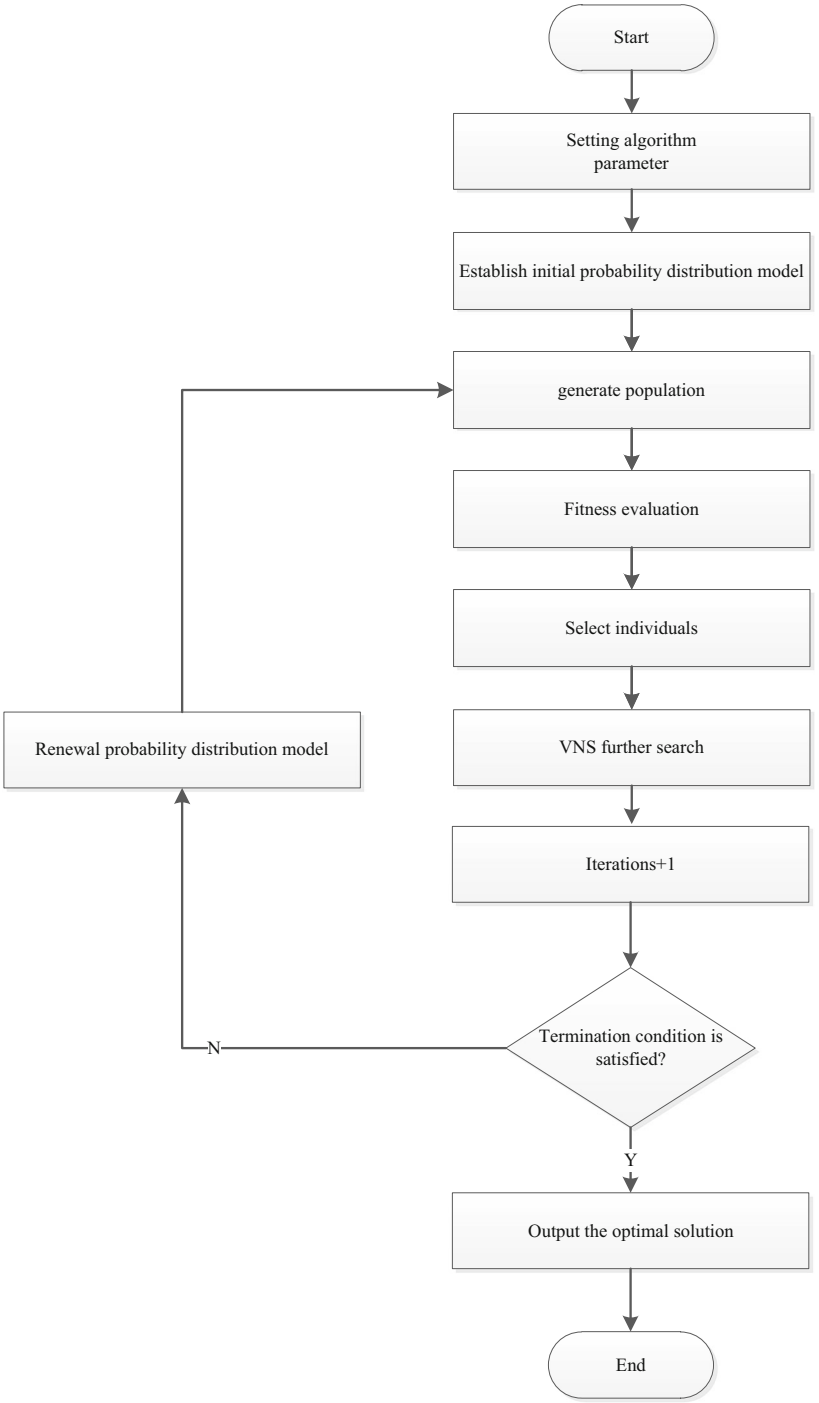



**Fig. 5** The flow chart of the EDA–VNS algorithm

The probabilistic matrix of examination server allocation is initialized as follows:

$$m_{ij}(0) = {}^{1}\!/_{m_e}, \quad j = 1, \ldots, m_e, \quad \forall i \in P_3 \cup P_4 \tag{7}$$

The probabilistic matrix of treatment server allocation is initialized as follows:

$$m_{ij}(0) = {}^{1}\!/_{m_t}, \quad j = 1, \ldots, m_t, \quad \forall i \in P_2 \cup P_4 \tag{8}$$

The probability matrix is updated by the following equations:

$$m_{ij}(g+1) = (1-\varphi) \times m_{ij}(g) + \frac{\varphi}{\theta \times pop} \sum_{1}^{\theta \times pop} I_{ij}, \quad \forall i, \theta \in (0,1), \varphi \in (0,1), j \neq 0 \tag{9}$$

where $g$ is the generation index, $i$ is the patient index, $pop$ is the population size, $\theta$ is the proportion of elite individuals, and $\varphi$ is the learning rate. $I_{ij} = 1$ if $j$ is selected to server patient $i$, otherwise $I_{ij} = 0$.

### 3.3 Encoding and decoding schemes

Sampling from the probabilistic models, we can obtain the server allocation solution. We propose the encoding and decoding procedures. Table 1 represents a simple server assignment example.

The Fig. 6 shows the encoding strategy to represent a solution. A patient index array is generated to record the index of patients (indicated by the number in each slot) who need examination service and treatment service. The server allocation solution includes two parts, i.e., the examination server allocation solution and treatment server allocation solution. The number in each slot of server allocation represents the server index. The sequence of the server allocation solution is according to the sequence of patient index. The number in each slot of server allocation array represents the server assigned to the patient in the corresponding slot of the patent index. The examination done array is proposed to record the patients who have already been examined.

The decoding processes: (1) search the target patient index in the patient index array. (2) Find the value of the corresponding position in the server allocation array. (3) If

**Table 1** A simple example

| Patient index | Path | Server |
|---|---|---|
| 1 | 4 | 1,3 |
| 2 | 3 | 2 |
| 3 | 3 | 2 |
| 4 | 4 | 1,4 |
| 5 | 2 | 3 |
| 6 | 2 | 4 |

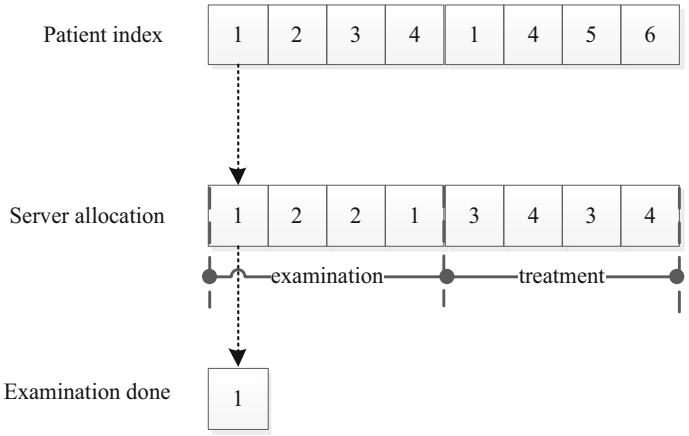



**Fig. 6** A simple example of encoding strategy

there are two servers for the patient, search the patient index in the examination done array, otherwise output the index of server. (4) If the patient index is in the examination done array, output the second server's index, otherwise output the first server's value. For example, patient 1 needs examination and treatment services. The first time we search the server allocation of patient 1, the number 1 in the patient index array can be found in the slot 1 and slot 5, the values of the corresponding positions in the server allocation array are 1 and 3. Since patient 1 is not in the examination done array, then output the server index 1, which means patient 1 is served by the examination server 1. When s/he has been examined, the next step after consultation is treatment. Patient 1 will be recorded in the examination done list after examination. The end examination time of patient 1 is updated as a new arrival time in the patient's arrival list. Then search for the next server of patient 1, because the number 1 is in the examination done array, output the second server's index 3.

### 3.4 Neighborhood structure of VNS

Since EDA algorithm may fall into local optimum, a basic variable neighborhood search (BVNS) algorithm is applied to improve the solutions by changing the neighborhood as it is an improved local search algorithm. It uses the neighborhood structure composed of different shakes to search alternately. Neighborhood structures are utilized to create new individuals by searching for neighborhood solutions. There are many papers about the modified VNS. Adibi and Shahrabi (2014) combined the VNS algorithm with the k-means algorithm as a modified VNS algorithm. Some papers modify the neighborhood structures to improve the efficiency of the VNS. Kong et al. (2019) proposed a reduced VNS algorithm with multiple random mutations neighborhood structures to solve the composition problem for Distributed Virtual Manufacturing Network. Puerto et al. (2014) used new neighborhood structures to make a more efficient encoding. We use three neighborhood structures to solve the problem of server allocation, including server-change neighborhood structure, server-swap neigh-

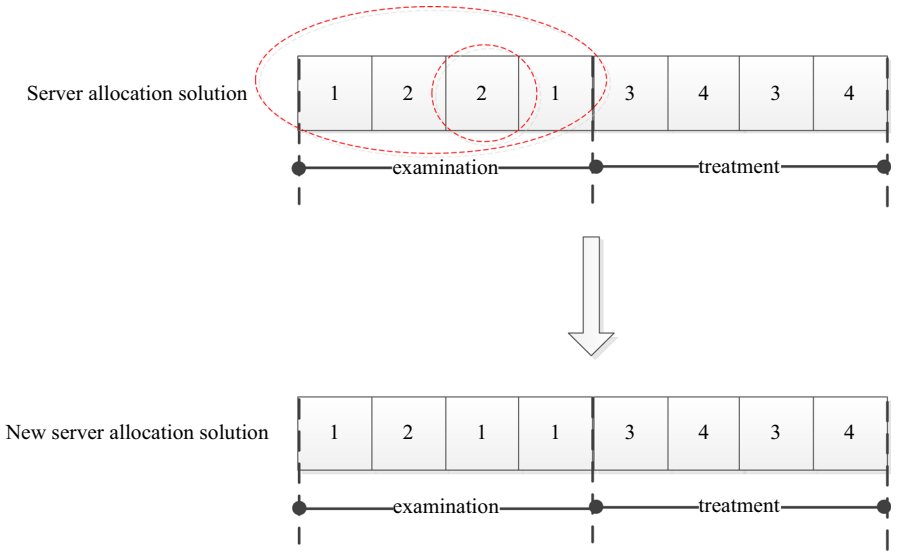



**Fig. 7** A simple example for server-change neighborhood structure

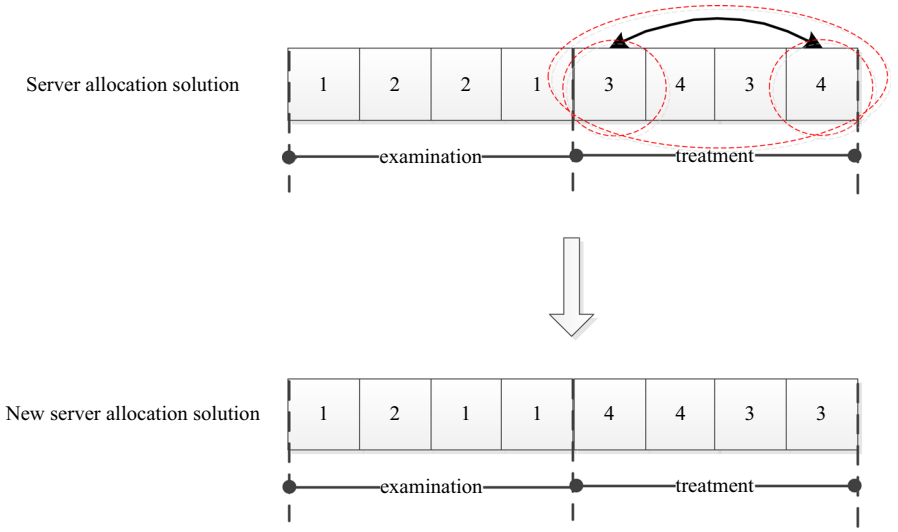



**Fig. 8** A simple example for server-swap neighborhood structure

borhood structure and a modified server-change neighborhood structure (Figs. 7, 8, 9). The server-change neighborhood structure is to randomly select the examination servers or treatment servers and randomly change the one patient's server allocation for this type of server. The server-swap neighborhood structure is to randomly select the examination servers or treatment servers and randomly swap two patients' server allocation of this type of server. The modified server-change neighborhood structure is

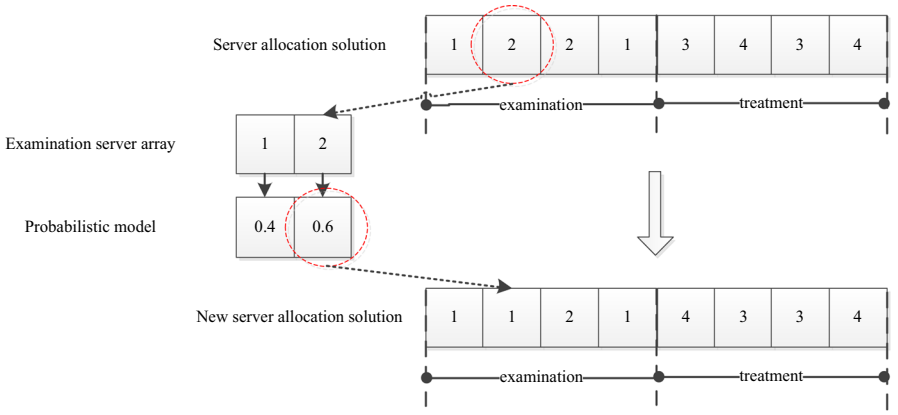



**Fig. 9** A simple example for modified server-change neighborhood structure

**Table 2** Procedures of modified server-change neighborhood structure

| Procedures of modified server-change neighborhood structure |
|---|
| Step 1. Randomly generate a number 0 or 1, 0 represents the examination server solution, 1 represents the treatment server solution |
| Step 2. Randomly extract one position from the solution, for each possible value of this position, obtain the selection probability |
| Step 3. If the selection probability of one possible value exceeds that of all other possible values, record the value and compare it with the value of the position |
| Step 4. If the value is equal to the value of the position, change the value of the position randomly until the two values are not equal |

designed according to the probabilistic models of the EDA algorithm. In the iterative process of probability model, there may be a situation that the probability of some values is increasingly getting larger, so it will fall into local optimum. If the probability of taking a certain value has exceeded the sum of the probabilities of other values, the value will be changed randomly. Because the probability sum of all possible values is 1, when the probability of a certain value is higher than 0.5, the value is randomly changed to other possible values. The main steps of the modified server-change neighborhood structure are as shown in Table 2. Figure 10 shows the pseudocode of VNS.

## 4 Patient sequence

After all the patients' assignments to servers are determined, the sequence of patients should be determined. We merge consultation and re-consultation as one service and add a virtual service to patients who only receive a consultation service. As a result, all patients have two stages. In stage 1, each patient receives consultation service at a doctor, and in stage 2, patients have examination services or treatment services or virtual services. Due to the complexity of the system, it is difficult to sequence all patients of

**The pseudocode of VNS**

Define neighborhood structures $N_k\ (k = 1, \cdots, k_{max})$

Get elite solution $x^0,\ x^0 \in X$ from EDA as the initial solution, $x^{best} \leftarrow x^0$

**While** stopping condition is not met **do**

$k \leftarrow 1$

**While** $k \le k_{max}$ **do**

$x' \leftarrow Shake(x^0), x' \in N_k(x^0)$ /*randomly generate a solution $x'$ from $N_k(x^0)$*/

$x'' \leftarrow LocalSearch(x'), x'' \in X$

**If** $x'' < x^0$

$x^{best} \leftarrow x'', k \leftarrow 1$

**Else**

$k \leftarrow k + 1$

**Endif**

**Endwhile**

**Endwhile**

**Output** $x^{best}$

**Fig. 10** The pseudocode of VNS

all stages at the same time. Firstly, we sort the doctors according to the available time, start with the earliest available doctor and sequence each doctor's patients. Only the first two arriving patients are sequenced at a time. Then the patient sequence on one server is considered. Since patients of different doctors may be assigned to one server, it will affect the server available time which is the basis of the sequence in stage 1. Therefore, after completing stage 2 sequence, the server available time is updated according to the patient ranked first on the server at this time. The first-server patient of each doctor should be reconsidered. After generating the server-allocation solution according to the proposed algorithm, the processes of patient sequence are as follows:

1. Generate the doctors' patient list according to the patients' appointments of each doctor. The patients on the list are ordered by the non-descending sequence of the patients' arrival times.
2. Compare the first and second arrival patients of each doctor and determine the first served patient of each doctor.
3. Compare the first served patients of each doctor who are allocated to the same server to determine the first served patient on the server. Refresh the available time of the server by calculating the occupying duration of the server that the first patients is served.

4. Back to step 2 to reconsider the first served patient of each doctor. Refresh the available time of doctors by calculating the consultation duration of the first served patient.
5. If the patient has an examination, the end examination time is a new patient's arrival time. Insert the patient arrival time into the doctors' patient list.

### 4.1 Patient sequence of one doctor

After consultation, the patient may have a further examination or treatment or release. The release can be regarded as a virtual server, which is available all the time and the duration is zero. The following is how we determine the sequence of patients for one doctor. Start with the earliest available doctor and select two earliest arrival patients $f$ and $k$ of the doctor. Assume it is the $l_f$ th consultation of patient $f$, and the $l_k$ th consultation of patient $k$. The arrival time of the two patients are $A_{f,l_f}$ and $A_{k,l_k}$, $A_{f,l_f} \leq A_{k,l_k}$. The scheduling time is $T$. The consultation durations of the two patients are $d_{f,1,l_f}$ and $d_{k,1,l_k}$, and the next service durations of the two patients are $d_{f,2,l_f}$ and $d_{k,2,l_k}$. The consultation duration is shorter than the examination duration and treatment duration. The next server available time are $X_{f,l_f}$ and $X_{k,l_k}$. The next servers of the two patients may be different or the same. Obviously, the assignment of other doctors' patients may influence the available time of this next server, and then the assignment of these two patients will also be affected. At first, we don't consider other doctor's patients and only consider these two patients which include two cases as follows, and then we extend to more complex situations.

*Case 1* The next servers of the two patients are the same, and no patient is between the two patients on the next server. $X_{f,l_f} = X_{k,l_k}$.

If schedule patient $f$ is scheduled before patient $k$ (short for Schedule 1), the completion time of patient $f$ is

$$C_{f,l_f} = max\big(X_{f,l_f}, max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}\big) + d_{f,2,l_f}$$

The completion time of patient $k$ is

$$C_{k,l_k} = max\big(C_{f,l_f}, max\big(max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k}$$

The completion time of the two patients is

$$\begin{aligned} C(1) = C_{f,l_f} + C_{k,l_k} &= max\big(X_{f,l_f}, max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}\big) \\ &+ max\big(max\big(X_{f,l_f}, max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}\big) \\ &+ d_{f,2,l_f}, max\big(max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}, A_{k,l_k}\big) + d_{k,1,l_k}\big) \\ &+ d_{f,2,l_f} + d_{k,2,l_k} \end{aligned}$$

Otherwise if patient $f$ is scheduled after patient $k$ (short for Schedule 2), similar to Schedule 1, the completion time of patient $k$ is

$$C_{k,l_k} = max\big(X_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k}$$

The completion time of patient $f$ is

$$C_{f,l_f} = max\big(C_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + d_{f,1,l_f}\big) + d_{f,2,l_f}$$

The completion time of Schedule 2 is

$$\begin{aligned} C(2) = C_{f,l_f} + C_{k,l_k} &= max\big(X_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k}\big) \\ &+ max\big(max\big(X_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k}\big) \\ &+ d_{k,2,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + d_{f,1,l_f}\big) + d_{k,2,l_k} + d_{f,2,l_f} \end{aligned}$$

If $C(1) \le C(2)$, then Schedule 1 should be adopted, otherwise Schedule 2. In particular, if two patients are released after consultation, then $d_{f,2,l_f} = d_{k,2,l_k} = 0$. The completion time is

$$\begin{aligned} C(1) &= max\big(T, A_{f,l_f}\big) + d_{f,1,l_f} + max\big(max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}, A_{k,l_k}\big) + d_{k,1,l_k} \\ C(2) &= max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + d_{f,1,l_f} \end{aligned}$$

We can quickly get the results in some conditions. For example, if the next server is the virtual server, the scheduling time is not earlier than patient $k$ arrival time $\big(T \ge A_{k,l_k}\big)$, the consultation duration of patient $f$ is no longer than that of patient $k$ $\big(d_{f,1,l_f} \le d_{k,1,l_k}\big)$.

$$\begin{aligned} C(1) &= T + d_{f,1,l_f} + T + d_{f,1,l_f} + d_{k,1,l_k} \\ C(2) &= T + d_{k,1,l_k} + T + d_{k,1,l_k} + d_{f,1,l_f} \end{aligned}$$

$C(1) \le C(2)$, then Schedule 1 should be adopted.

The conditions and corresponding decision-making tree for two patient sequences in Case 1 are listed Table 3 (Fig. 11).

*Case 2* The next servers of the two patients are different.

Similar to Case 1,

$$\begin{aligned} C\big(1^*\big) &= max\big(X_{f,l_f}, max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}\big) + d_{f,2,l_f} \\ &+ max\big(X_{k,l_k}, max\big(max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k} \\ C\big(2^*\big) &= max\big(X_{f,l_f}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + d_{f,1,l_f}\big) + d_{f,2,l_f} \\ &+ max\big(X_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k} \end{aligned}$$

If $C(1^*) \le C(2^*)$, then Schedule 1 should be adopted, otherwise Schedule 2. In particular, if patient $f$ will release after consultation, and patient $k$ will have examination or treatment.

$$\begin{aligned} C\big(1^*\big) &= max\big(T, A_{f,l_f}\big) + d_{f,1,l_f} \\ &+ max\big(X_{k,l_k}, max\big(max\big(T, A_{f,l_f}\big) + d_{f,1,l_f}, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k} \\ C\big(2^*\big) &= max\big(T, A_{k,l_k}\big) + d_{k,1,l_k} + d_{f,1,l_f} \\ &+ max\big(X_{k,l_k}, max\big(T, A_{k,l_k}\big) + d_{k,1,l_k}\big) + d_{k,2,l_k} \end{aligned}$$

**Table 3** The decision conditions of Case 1 of patient sequence for one doctor

| Conditions | |
|---|---|
| ① | The next server is the virtual server. $d_{f,2,l_f} = d_{k,2,l_k} = 0$, $X_{f,l_f} = X_{k,l_k} = 0$ |
| ② | Scheduling time is not earlier than patient $k$ arrival time. $T \geq A_{k,l_k}$ |
| ③ | The consultation duration of patient $f$ is no longer than that of patient $k$. $d_{f,1,l_f} \leq d_{k,1,l_k}$ |
| ④ | The next service duration of patient $f$ is no longer than that of patient $k$. $d_{f,2,l_f} \leq d_{k,2,l_k}$ |
| ⑤ | If patient $f$ consults first, and patient $k$ has not arrived at the end of the consultation time of patient $f$. $A_{k,l_k} > max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}$ |
| ⑥ | No matter who is the first patient, the next server is not idle at the end of the consultation time of the first patient. $X_{f,l_f} > max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}$, $X_{k,l_k} > max\left(T, A_{k,l_k}\right) + d_{k,1,l_k}$, $X_{f,l_f} = X_{k,l_k}$ |
| ⑦ | If patient $f$ consults first, the next server will be idle at the end of the consultation time of patient $f$, while if patient $k$ consults first, and the next server will not be idle at the end of the consultation time of patient $k$. $X_{f,l_f} \leq max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}$, $X_{k,l_k} > max\left(T, A_{k,l_k}\right) + d_{k,1,l_k}$, $X_{f,l_f} = X_{k,l_k}$ |
| ⑧ | If patient $f$ consults first, twice the difference between the completion time and the next server idle time is no longer than the difference between the duration of patient $k$ and the duration of patient $f$ on the next server. $2 * \left[max\left(T, A_{f,l_f}\right) + d_{f,1,l_f} - X_{k,l_k}\right] \leq d_{k,2,l_k} - d_{f,2,l_f}$ |
| ⑨ | Calculate $C(1)$ and $C(2)$, $C(1) \leq C(2)$ |

If patient $k$ will release after consultation, and patient $f$ will have examination or treatment.

$$\begin{aligned} C\left(1^*\right) &= max\left(X_{f,l_f}, max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}\right) + d_{f,2,l_f} \\ &\quad + max\left(max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}, A_{k,l_k}\right) + d_{k,1,l_k} \\ C\left(2^*\right) &= max\left(X_{f,l_f}, max\left(T, A_{k,l_k}\right) + d_{k,1,l_k} + d_{f,1,l_f}\right) + d_{f,2,l_f} \\ &\quad + max\left(T, A_{k,l_k}\right) + d_{k,1,l_k} \end{aligned}$$

The conditions and corresponding decision-making tree for the two patient sequences in Case 2 are listed in Table 4 (Fig. 12).

Table 5 shows the Sequence policy 1. By applying policy 1, all doctors have a first scheduled patient. If the first scheduled patients are assigned to different servers in the next step, then there will be no interference with each other. However, owing to the constraint on the number of servers, patients may be assigned to the same server. Then the patient sequence on one server is considered in the following section.

### 4.2 Patient sequence of one server

After consultation, the patients will be served on one server or release. If the patients of different doctors are served on the same server, the available time of this server may be changed. We should change the consultation sequence according to the updated available time of server.

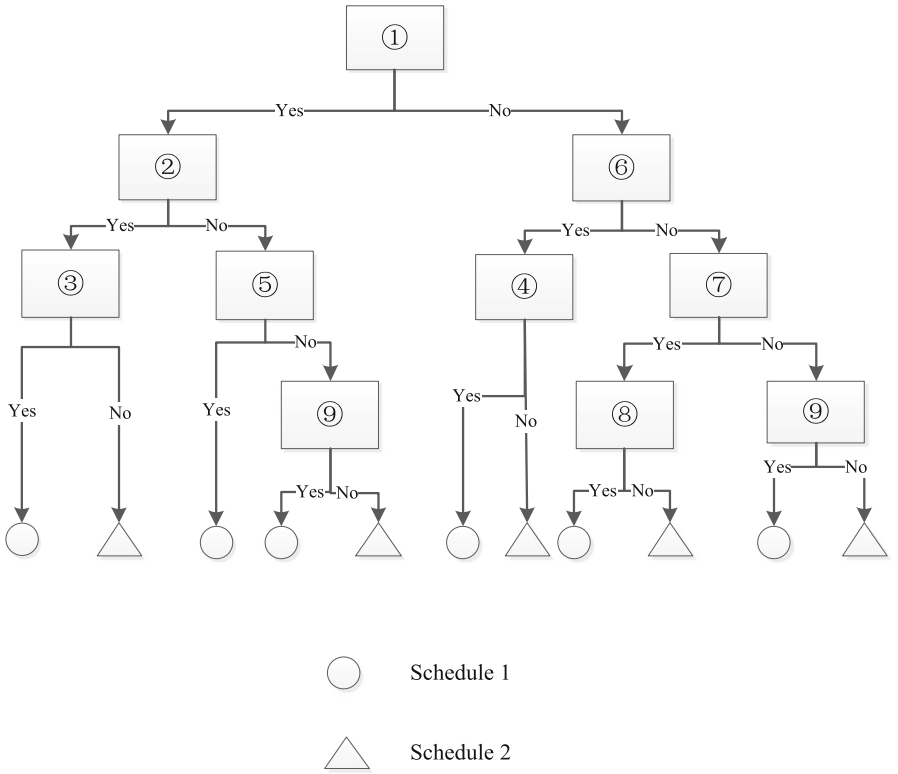



**Fig. 11** The decision-making tree of Case 1 of patient sequence for one doctor

If two patients $k$ and $l$ are assigned to the same server. The end consultation time are $E_k$ and $E_l$. $E_k < E_l$. The initial available time of the server is $X_M$. The service duration on the server are $d_k$ and $d_l$. The Schedule 1 is patient $k$ before patient $l$, and the Schedule 2 is on the contrary.

For Schedule 1, the completion time of patient $k$ on the server is

$$C_k = max(X_M, E_k) + d_k$$

The completion time of patient $l$ on the server is

$$C_l = max(C_k, E_l) + d_l$$

Then the completion time of two patients is

$$C(1) = max(X_M, E_k) + max(max(X_M, E_k) + d_k, E_l) + d_k + d_l$$

Similar to Schedule 1, the completion time of two patients of Schedule 2 is

$$C(2) = max(X_M, E_l) + max(max(X_M, E_l) + d_l, E_k) + d_k + d_l$$

**Table 4** The decision conditions of Case 2 of patient sequence for one doctor

| Conditions | |
|---|---|
| ① | No patient will leave the clinic after consultation. $d_{f,2,l_f} \neq 0$, $d_{k,2,l_k} \neq 0$ |
| ② | If patient $k$ consults first, and patient $k$'s next server is not available after patient $f$'s consultation. $X_{k,l_k} > max\left(T, A_{k,l_k}\right) + d_{k,1,l_k} + d_{f,1,l_f}$ |
| ③ | If patient $k$ consults first, and patient $f$'s next server is not available after patient $f$'s consultation. $X_{f,l_f} > max\left(T, A_{k,l_k}\right) + d_{k,1,l_k} + d_{f,1,l_f}$ |
| ④ | If patient $f$ consults first, and patient $k$ has not arrived after patient $f$'s consultation. $A_{k,l_k} > max\left(T, A_{f,l_f}\right) + d_{f,1,l_f}$ |
| ⑤ | If patient $f$ will release after consultation, and if patient $f$ consults first, patient $k$'s next server is not available after patient $k$'s consultation. $d_{f,2,l_f} = 0$, $X_{k,l_k} > max\left(T, A_{f,l_f}\right) + d_{f,1,l_f} + d_{k,1,l_k}$ |
| ⑥ | If patient $k$ will release after consultation, and if patient $k$ consults first, patient $f$'s next server is not available after patient $f$'s consultation. $d_{k,2,l_k} = 0$, $X_{f,l_f} > max\left(T, A_{k,l_k}\right) + d_{k,1,l_k} + d_{f,1,l_f}$ |
| ⑦ | Calculate $C(1^*)$ and $C(2^*)$, $C(1^*) \leq C(2^*)$ |

The conditions and corresponding decision-making tree for two patient sequences on one server are listed Table 6 (Fig. 13).

We can get a Sequence policy 2 to schedule the patients on the same server. Then we turn back to schedule the patients on the consultation phase according to the sequence on the server. Table 7 represents the steps of policy 2.

## 5 Computational experiments and comparisons

In this section, we design the computational experiments and compare the results with EDA (Wang et al. 2013), VNS (Lei and Guo 2016) and PSO (particle swarm optimization) (Taherkhani and Safabakhsh 2016) to evaluate the performance of EDA–VNS algorithm. There are three parameters in EDA and EDA–VNS: $pop$ (population scale), $\theta$ (proportion of elite individuals) and $\varphi$ (learning rate). Parameter settings may affect the results of the experiments. We give three reasonable values of each parameter in Table 8. Each possible value of a parameter is sorted by increasing order, ranking from 1 to 3.

We do 9 small-scale experiments to determine the optimal combination of parameters. The number of patients is 20, the number of doctors is 3, the number of examination servers and treatment servers is both 2. The experiments results are obtained by EDA–VNS algorithm. Each experiment is repeated twenty times and the average minimum value is adopted to evaluate the results. Figure 14 shows the convergence curves for 9 different combinations of parameters. Table 9 represents the average minimum value in twenty experiments.

The results in Table 10 show the population scale is the most influential factor among the three parameters. With the population scale increasing, the final results

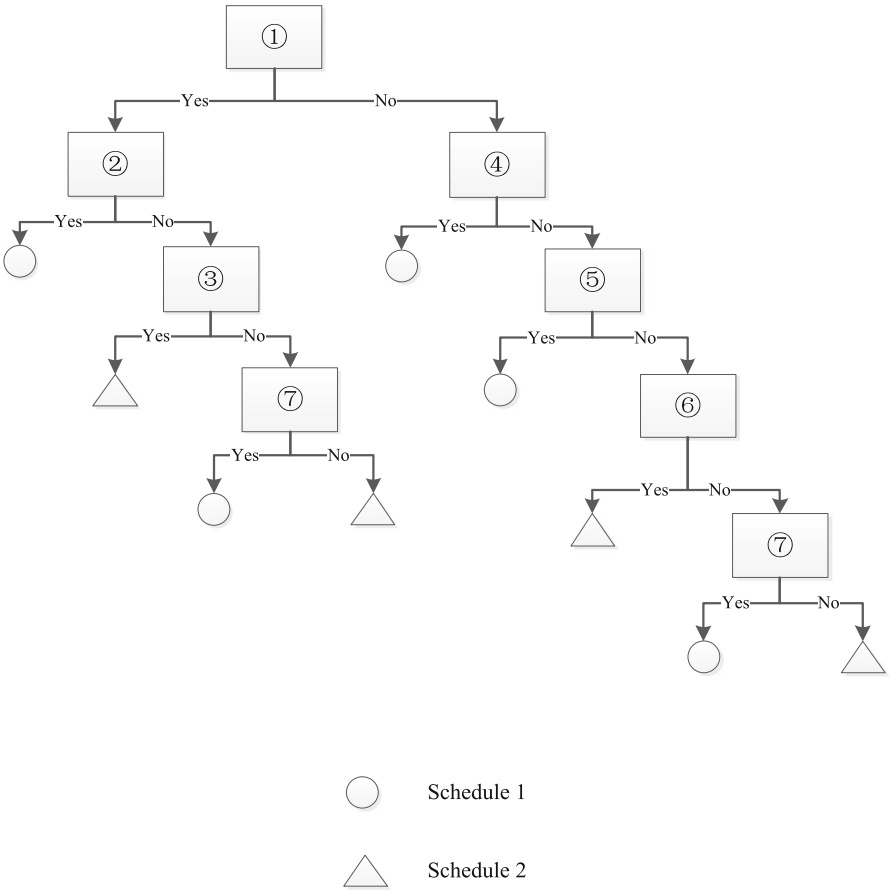



**Fig. 12** The decision-making tree of Case 2 of patient sequence for one doctor

**Table 5** The steps of Sequence policy 1

| Sequence policy 1 |
|---|
| *Step 1*. Search the earliest available doctor, assign the available time to $T$ |
| *Step 2*. Search two earliest arriving patients of this doctor, get the arrival time of $A_{f,l_f}$ and $A_{k,l_k}$ |
| *Step 3*. If the two patients are allocated to the same next server, compare C(1)and C(2), otherwise compare $C(1^*)$ and $C(2^*)$ |
| *Step 4*. Select the second earliest available doctor, repeat the above steps until all doctors have a first schedule patient |

become better. However, a larger population scale needs a longer computational time. Population size of 50 individuals is chosen to balance the quality of solution and the computational time of experiments. The increase in the selection rate of elite solutions causes the solution to improve first and then to deteriorate. Hence the value 0.3 is selected. The parameters settings are as shown Table 11.

**Table 6** The decision conditions of patient sequence on one server

| Conditions | |
|---|---|
| ① | The server is not idle when two patients complete the consultation. $X_M > E_l$ |
| ② | The service duration of patient $k$ is no longer than that of patient $l$ on the server. $d_k \leq d_l$ |
| ③ | The server is idle when patients $k$ completes the consultation. $X_M \leq E_k$ |
| ④ | If the difference between the end consultation time of patient $l$ and that of patient $k$ is not less than the service duration of patients $k$. $E_l - E_k \geq d_k$ |
| ⑤ | If the twice difference between the end consultation time of two patients is not less than the difference between the server duration of patient $k$ and that of patient $l$. $2 * (E_l - E_k) \geq d_k - d_l$ |
| ⑥ | If the twice difference between the end consultation time of patients $l$ and the initial available time of the server is not less than the difference between the server duration of patient $k$ and that of patient $l$. $2 * (E_l - X_M) \geq d_k - d_l$ |

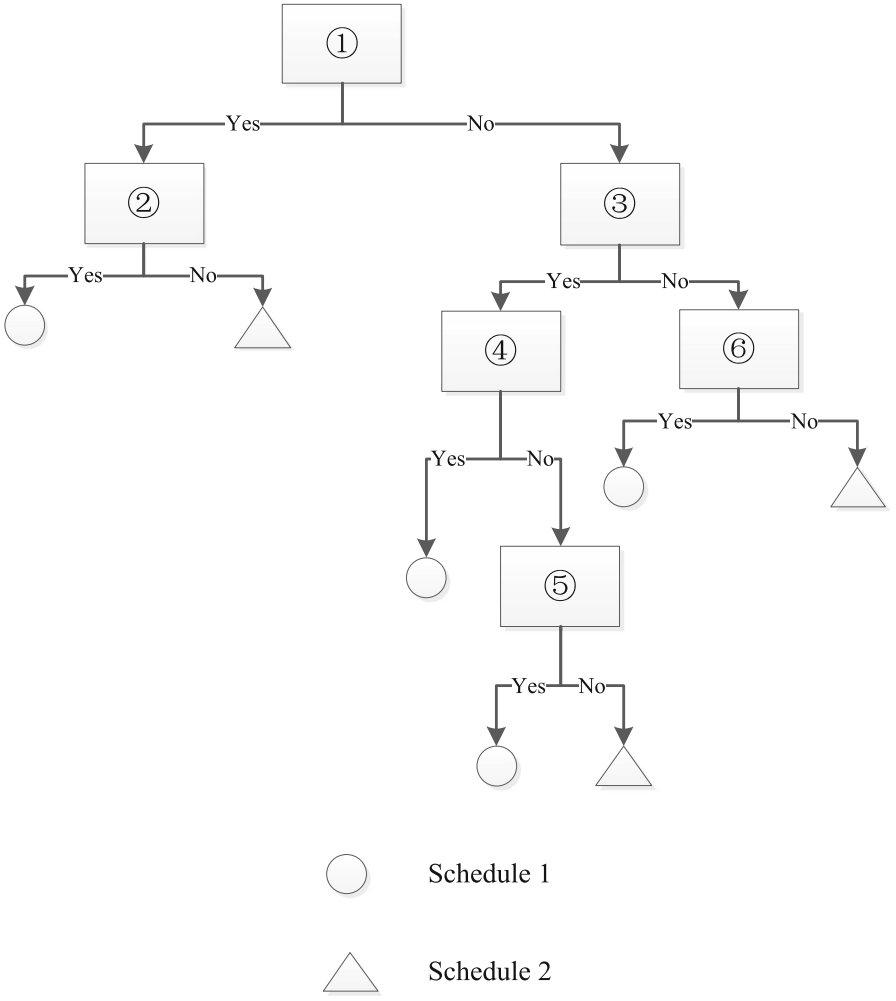



**Fig. 13** The decision-making tree of patient sequence on one server

**Table 7** The steps of Sequence policy 2

| Sequence policy 2 |
|---|
| *Step 1*. Search the first patients of different doctors who are assigned to the same next server. Select one server |
| *Step 2*. Search the first patient on the server, calculate the end service time and refresh the available time of this server |
| *Step 3*. Search the second patient on the server; go back to the policy 1 step3 according to the new available time of this server |
| *Step 4*. If the schedule does not change, then refresh the available time of this server |
| *Step 5*. Repeat step 3 and step 4, until the patients of different doctors who are assigned to this same next server is determined |
| *Step 6*. Repeat the above steps until all the next servers satisfied the condition are selected once |

**Table 8** Combination of parameter values

| Parameters | Reasonable values | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| $pop$ | 10.0 | 20.0 | 50.0 |
| $\theta$ | 0.1 | 0.3 | 0.5 |
| $\varphi$ | 0.1 | 0.3 | 0.5 |

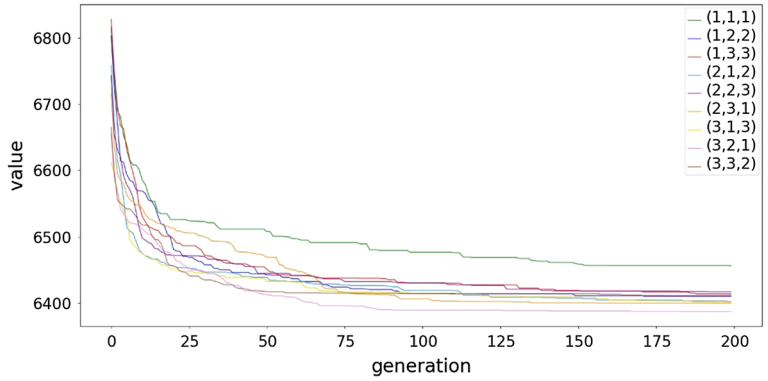



**Fig. 14** Convergence curves for 9 different parameter settings

Table 12 shows the 27 instances and the experiment results. Each instance is tested operated by four algorithms. We code four algorithms in python and run on an Acer computer running Windows 10 with i5-8500U CPU@3.0 GHz and 16 GB RAM. The maximum iteration is 200. Each experiment is repeated 20 times. The combinations represent the number of patients, the number of doctors and the number of servers. Because the number of examination servers and the number of treatment servers are set to be the same, the number of servers can represent both the number of examination servers and the number of treatment servers. The average value of each generation in 20 experiments (Ave) and the minimum value of all experiments (Min) are also recorded in Table 12. The relative percent deviation (RPD) is used to measure the ability of an algorithm to find the best solution:

**Table 9** Orthogonal array and average minimum value

| Experiment number | Parameter settings rank | | | Average minimum value |
|---|---|---|---|---|
| | $pop$ | $\theta$ | $\varphi$ | |
| 1 | 1 | 1 | 1 | 6456.15 |
| 2 | 1 | 2 | 2 | 6411.3 |
| 3 | 1 | 3 | 3 | 6413.9 |
| 4 | 2 | 1 | 2 | 6402.35 |
| 5 | 2 | 2 | 3 | 6417.1 |
| 6 | 2 | 3 | 1 | 6399.95 |
| 7 | 3 | 1 | 3 | 6400.95 |
| 8 | 3 | 2 | 1 | 6387.25 |
| 9 | 3 | 3 | 2 | 6410.0 |

**Table 10** Response values and ranks of each parameter

| Rank | $pop$ | $\theta$ | $\varphi$ |
|---|---|---|---|
| 1 | 6427.12 | 6419.82 | 6414.45 |
| 2 | 6406.47 | 6405.22 | 6407.89 |
| 3 | 6399.40 | 6407.95 | 6406.47 |
| The difference between maximum and minimum value | 27.72 | 14.60 | 7.98 |
| Impact level | 1 | 2 | 3 |

**Table 11** Parameters setting

| Definition | Value |
|---|---|
| The number of patients | 50,100, 200 |
| The number of doctors | 3, 6, 9 |
| The number of examination servers | 2, 3, 4 |
| The number of treatment servers | 2, 3, 4 |
| Consultation/re-consultation duration | U[5,10] |
| Examination duration | U[60,120] |
| Treatment duration | U[15,20] |
| Population scale | 50 |
| Selection rate of elite solutions | 0.3 |
| Learning rate of EDA and EDA–VNS | 0.5 |

$$\mathrm{RPD}(ALG) = \frac{Efit(ALG) - bestfit}{bestfit} * 100$$

The value of denominator is the minimum value of four algorithms' minimum value. The numerator is the difference between the expected value of an algorithm with the best value of four algorithm. In this paper, the RPD is:

**Table 12** Computational results by algorithms EDA–VNS, EDA, VNS, and PSO

| No. | Combinations | EDA–VNS | | EDA | | VNS | | PSO | |
|---|---|---|---|---|---|---|---|---|---|
| | | Ave. (min) | RPD | Ave. (min) | RPD | Ave. (min) | RPD | Ave. (min) | RPD |
| 1 | $50 \times 3 \times 2$ | 16,167.85 (16,097.0) | 0.44 | 16,222.2 (16,102.0) | 0.78 | 16,476.95 (16,351.0) | 2.36 | 16,530.65 (16,173.0) | 2.69 |
| 2 | $50 \times 3 \times 3$ | 15,308.1 (15,145.0) | 1.08 | 15,483.85 (15,299.0) | 2.24 | 15,909.8 (15,694.0) | 5.05 | 15,923.65 (15,460.0) | 5.14 |
| 3 | $50 \times 3 \times 4$ | 12,263.8 (12,179.0) | 0.7 | 12,317.95 (12,227.0) | 1.14 | 12,528.85 (12,396.0) | 2.87 | 12,537.05 (12,444.0) | 2.94 |
| 4 | $50 \times 6 \times 2$ | 18,646.35 (18,452.0) | 1.05 | 19,013.0 (18,472.0) | 3.04 | 19,921.2 (19,328.0) | 7.96 | 19,940.75 (18,956.0) | 8.07 |
| 5 | $50 \times 6 \times 3$ | 17,256.8 (17,033.0) | 1.31 | 17,497.75 (17,209.0) | 2.73 | 17,863.3 (17,512.0) | 4.87 | 17,849.7 (17,496.0) | 4.79 |
| 6 | $50 \times 6 \times 4$ | 14,258.3 (14,095.0) | 1.16 | 14,360.85 (14,190.0) | 1.89 | 14,695.65 (14,452.0) | 4.26 | 14,635.65 (14,354.0) | 3.84 |
| 7 | $50 \times 9 \times 2$ | 18,878.9 (18,672.0) | 1.11 | 18,973.35 (18,784.0) | 1.61 | 19,159.6 (19,028.0) | 2.61 | 19,168.5 (18,902.0) | 2.66 |
| 8 | $50 \times 9 \times 3$ | 17,193.55 (17,014.0) | 1.06 | 17,344.55 (17,097.0) | 1.94 | 18,052.8 (17,641.0) | 6.11 | 18,020.25 (17,505.0) | 5.91 |
| 9 | $50 \times 9 \times 4$ | 12,814.75 (12,609.0) | 1.63 | 12,920.0 (12,743.0) | 2.47 | 13,378.55 (13,200.0) | 6.1 | 13,321.5 (12,921.0) | 5.65 |
| 10 | $100 \times 3 \times 2$ | 74,765.5 (73,497.0) | 1.73 | 75,502.2 (73,998.0) | 2.73 | 76,757.65 (76,473.0) | 4.44 | 76,942.95 (76,483.0) | 4.69 |
| 11 | $100 \times 3 \times 3$ | 42,101.8 (41,568.0) | 1.28 | 42,602.5 (42,162.0) | 2.49 | 44,161.45 (43,580.0) | 6.24 | 43,936.65 (42,856.0) | 5.7 |
| 12 | $100 \times 3 \times 4$ | 33,891.05 (33,508.0) | 1.14 | 34,217.5 (33,886.0) | 2.12 | 34,899.25 (34,579.0) | 4.15 | 34,834.85 (34,200.0) | 3.96 |
| 13 | $100 \times 6 \times 2$ | 65,837.95 (64,679.0) | 1.79 | 66,993.7 (65,453.0) | 3.58 | 68,697.75 (67,431.0) | 6.21 | 68,602.7 (66,725.0) | 6.07 |
| 14 | $100 \times 6 \times 3$ | 37,941.65 (37,480.0) | 1.23 | 38,551.7 (37,689.0) | 2.86 | 39,972.45 (39,126.0) | 6.65 | 39,918.2 (38,994.0) | 6.51 |
| 15 | $100 \times 6 \times 4$ | 41,728.7 (41,177.0) | 1.34 | 42,476.35 (41,598.0) | 3.16 | 44,228.55 (43,330.0) | 7.41 | 43,965.65 (42,879.0) | 6.77 |
| 16 | $100 \times 9 \times 2$ | 48,892.6 (48,033.0) | 1.79 | 49,368.3 (48,657.0) | 2.78 | 51,284.95 (50,411.0) | 6.77 | 51,076.5 (49,467.0) | 6.34 |
| 17 | $100 \times 9 \times 3$ | 46,196.85 (45,770.0) | 0.93 | 46,790.5 (45,870.0) | 2.23 | 49,529.2 (48,816.0) | 8.21 | 49,163.55 (47,899.0) | 7.41 |
| 18 | $100 \times 9 \times 4$ | 48,011.95 (47,338.0) | 1.42 | 49,186.5 (47,753.0) | 3.9 | 51,904.6 (50,963.0) | 9.65 | 51,989.1 (50,400.0) | 9.83 |

**Table 12** continued

| No. | Combinations | EDA–VNS | | EDA | | VNS | | PSO | |
|---|---|---|---|---|---|---|---|---|---|
| | | Ave. (min) | RPD | Ave. (min) | RPD | Ave. (min) | RPD | Ave. (min) | RPD |
| 19 | $200 \times 3 \times 2$ | 277,998.2 (274,284.0) | 1.35 | 280,244.9 (275,695.0) | 2.17 | 282,987.05 (281,135.0) | 3.17 | 282,854.1 (281,792.0) | 3.12 |
| 20 | $200 \times 3 \times 3$ | 179,544.75 (177,740.0) | 1.02 | 181,861.45 (179,308.0) | 2.32 | 184,263.95 (183,438.0) | 3.67 | 184,205.45 (183,326.0) | 3.64 |
| 21 | $200 \times 3 \times 4$ | 156,487.65 (155,424.0) | 0.68 | 157,747.9 (156,268.0) | 1.5 | 159,948.1 (159,233.0) | 2.91 | 159,636.4 (158,328.0) | 2.71 |
| 22 | $200 \times 6 \times 2$ | 250,168.15 (247,787.0) | 0.96 | 253,273.2 (248,563.0) | 2.21 | 258,827.35 (256,852.0) | 4.46 | 257,894.65 (255,511.0) | 4.08 |
| 23 | $200 \times 6 \times 3$ | 167,092.5 (164,540.0) | 1.55 | 170,121.65 (166,385.0) | 3.39 | 174,776.2 (173,179.0) | 6.22 | 175,516.7 (172,545.0) | 6.67 |
| 24 | $200 \times 6 \times 4$ | 140,160.95 (138,370.0) | 1.29 | 142,362.1 (140,044.0) | 2.89 | 151,139.3 (148,401.0) | 9.23 | 150,256.35 (146,501.0) | 8.59 |
| 25 | $200 \times 9 \times 2$ | 252,962.6 (250,593.0) | 0.95 | 257,954.4 (251,783.0) | 2.94 | 266,389.45 (261,933.0) | 6.3 | 266,129.85 (260,063.0) | 6.2 |
| 26 | $200 \times 9 \times 3$ | 169,285.85 (167,624.0) | 0.99 | 172,046.65 (169,530.0) | 2.64 | 179,640.9 (177,678.0) | 7.17 | 179,745.05 (175,285.0) | 7.23 |
| 27 | $200 \times 9 \times 4$ | 112,278.35 (111,522.0) | 0.68 | 114,260.7 (112,623.0) | 2.46 | 119,339.4 (117,420.0) | 7.01 | 119,079.75 (116,929.0) | 6.78 |

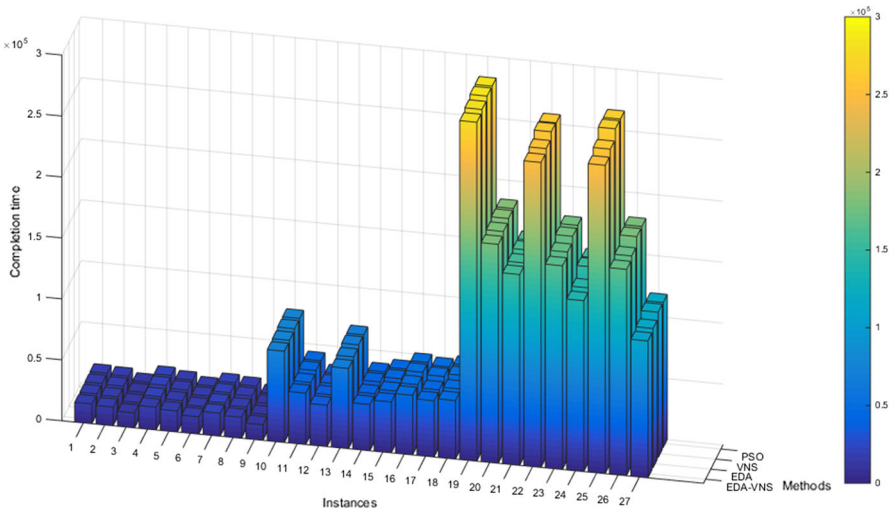



**Fig. 15** Completion time for different numerical instances

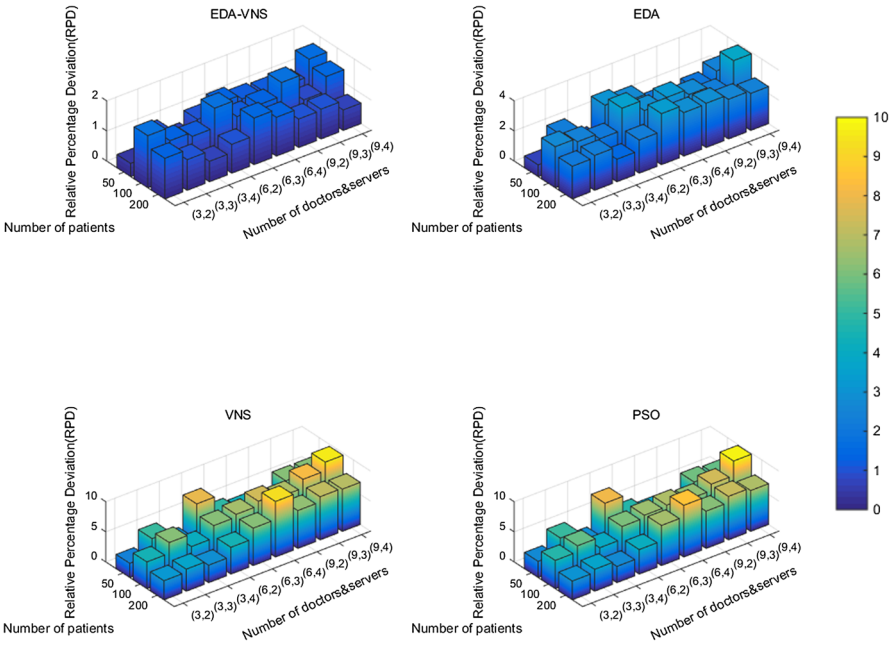



**Fig. 16** RPD results with different numbers of patients, doctors and servers

$$\mathrm{RPD}(ALG) = \frac{Ave\_min(ALG) - Min(Four\_ALG)}{Min(Four\_ALG)} * 100$$

Table 12 shows the results of four algorithms. The average value, the minimum value, and the RPD of four algorithms are used to measure the effectiveness of four

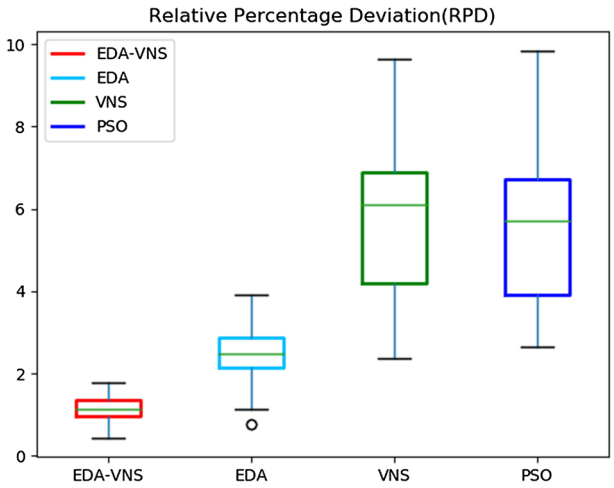
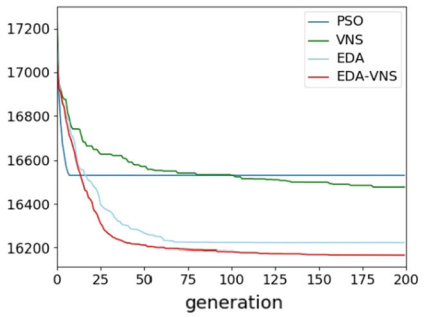



**Fig. 17** Box plots of RPD for different instances

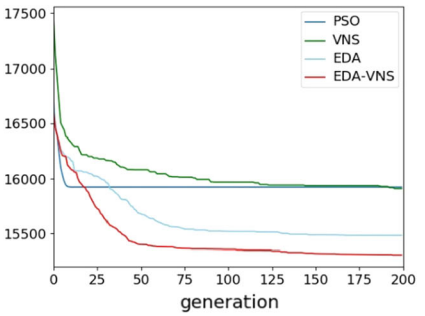
algorithms. It is obviously that EDA–VNS can obtain the best values in all instances among the four algorithms under all measurement criteria.

Figure 15 shows the completion time of four algorithms for 27 instances. The completion time increases with the number of patients increasing. The completion time decreases when the number of doctors and servers increase. The EDA–VNS algorithm can obtain the best objective value among the four algorithms for 27 instances. The solutions of EDA are better than VNS and PSO. Figure 16 represents the RPD of four algorithms. We can easily find that EDA–VNS has the smallest RPD. The RPD of VNS and PSO are large and the overall trend of RPD of these two algorithms is similar. We can obtain more information of RPD from Fig. 17, which demonstrates the boxplots of four algorithms. The boxplots can show the maximum, minimum, median, upper and lower quartiles of a set of data. VNS has smaller maximum and minimum RPD than PSO, while the median, upper and lower quartiles are larger than PSO. The five values of EDA are better than VNS and PSO. The EDA–VNS is the best among the four algorithms.

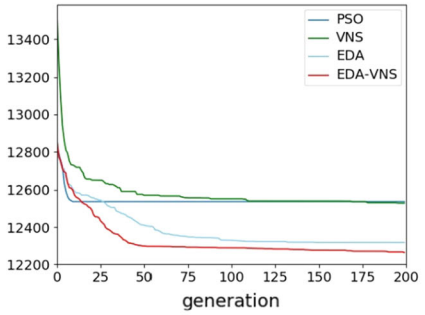
The convergence curves of four algorithms for 27 instances are represented in Fig. 18. Points on the convergence curves are the average value of each generation. The differences between PSO and VNS are close in all instances. The differences among EDA–VNS and EDA, VNS and PSO become larger with the increase of number of patients. Especially when the number of patients is 50, the differences between EDA–VNS and EDA is small. With the increase of the number of doctors, the differences between EDA–VNS and EDA basically become large at first, then smaller. But when the number of patients is 200, the number of servers is 2, the doctor number is changed from 6 to 9, and the differences become larger. EDA–VNS basically converges after 200 iterations. EDA algorithm basically converges when the number of patients is less than 200. PSO has always shown good convergence. The convergence of VNS is worse than EDA–VNS. Obviously, the experimental results show EDA–VNS algorithm outperforms among four algorithms. What's more, the effective-



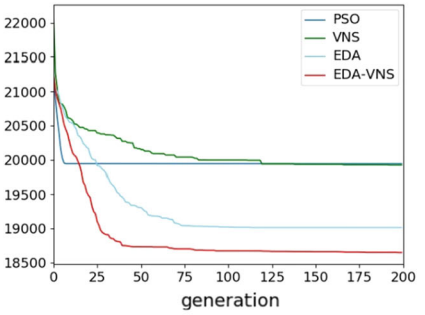

(1) Convergence curve for (50,3,2)



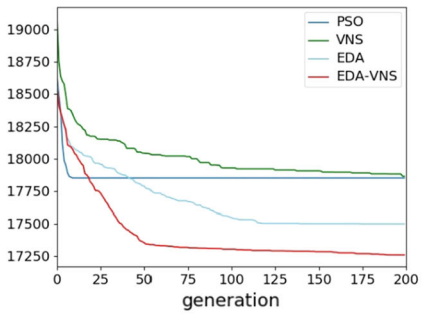

(2) Convergence curve for (50,3,3)



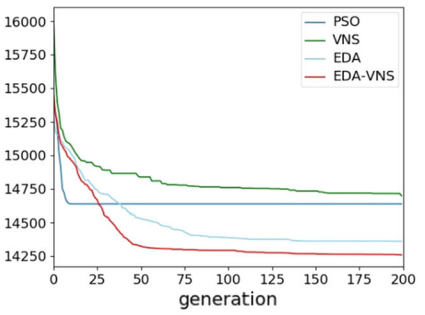

(3) Convergence curve for (50,3,4)



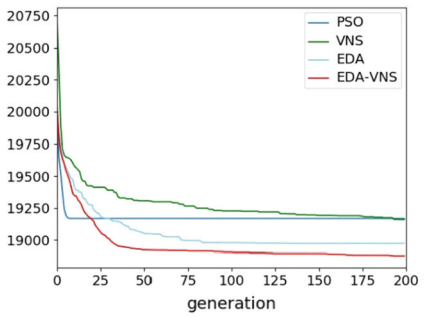

(4) Convergence curve for (50,6,2)



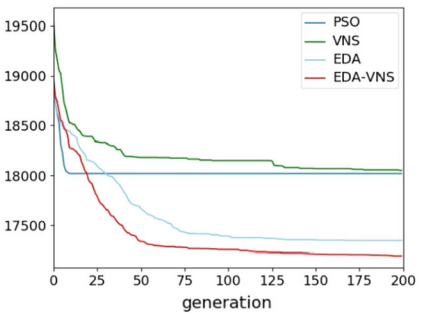

(5) Convergence curve for (50,6,3)




(6) Convergence curve for (50,6,4)

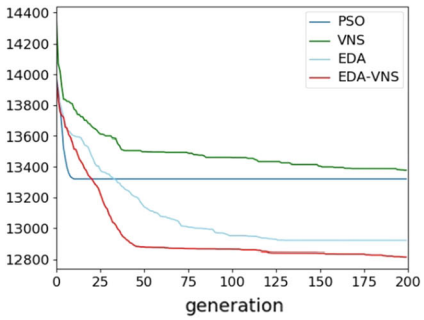



(7) Convergence curve for (50,9,2)

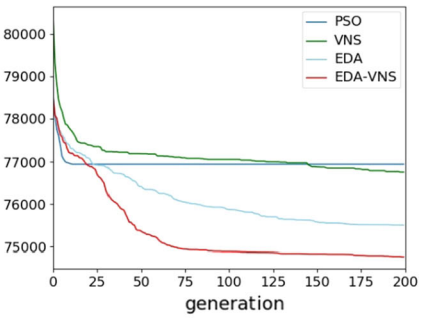



(8) Convergence curve for (50,9,3)

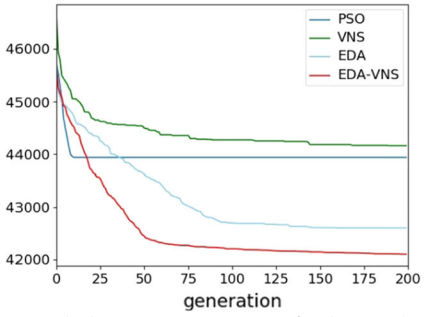
**Fig. 18** Convergence curves of four algorithms for different instances



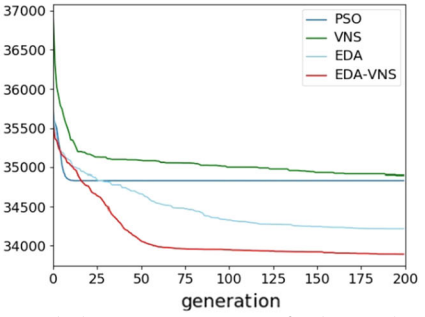

(9) Convergence curve for (50,9,4)



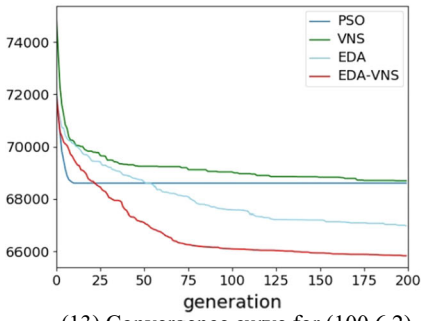

(10) Convergence curve for (100,3,2)



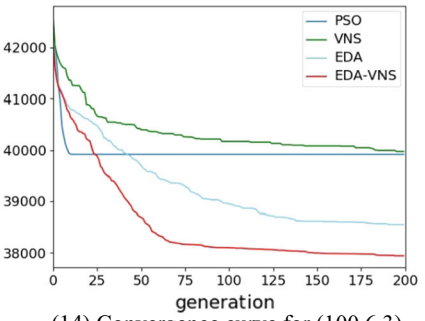

(11) Convergence curve for (100,3,3)



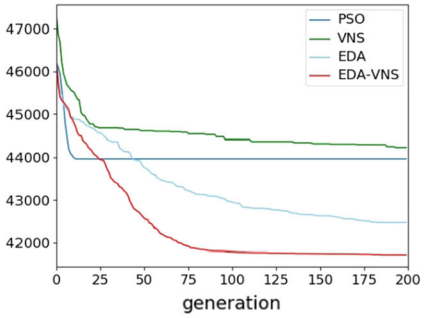

(12) Convergence curve for (100,3,4)



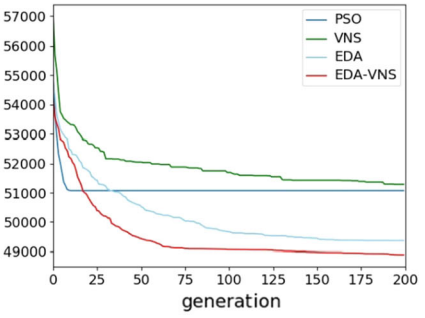

(13) Convergence curve for (100,6,2)




(14) Convergence curve for (100,6,3)

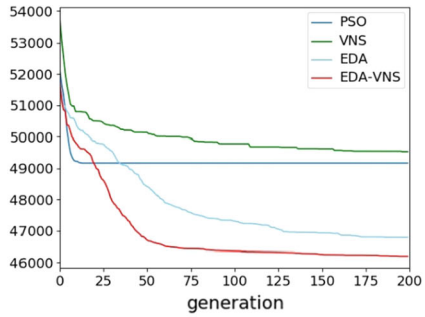



(15) Convergence curve for (100,6,4)

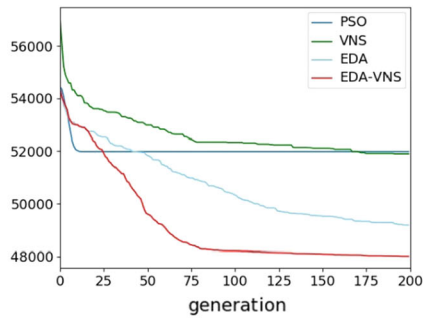



(16) Convergence curve for (100,9,2)

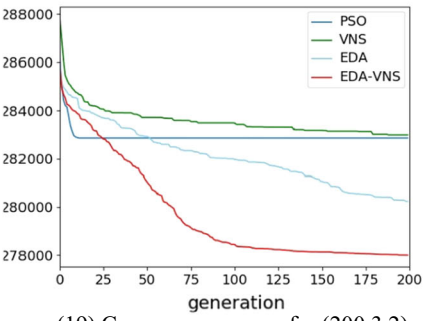
**Fig. 18** continued



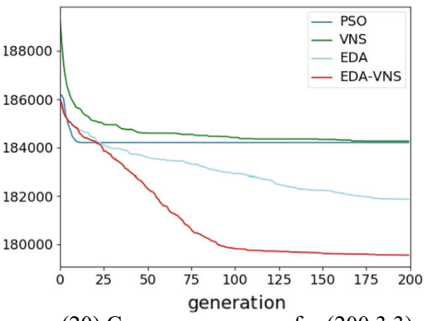

(17) Convergence curve for (100,9,3)



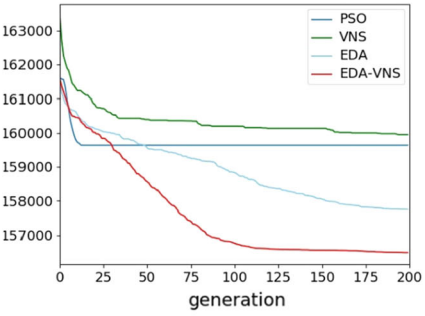

(18) Convergence curve for (100,9,4)



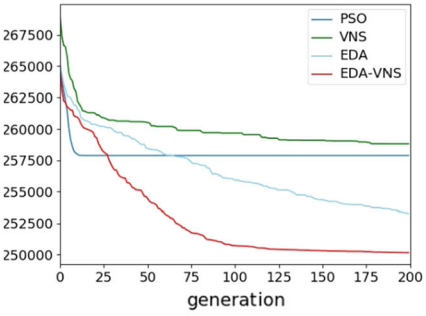

(19) Convergence curve for (200,3,2)



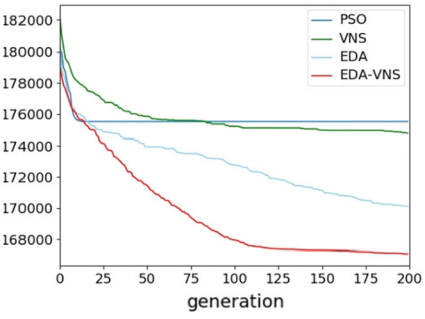

(20) Convergence curve for (200,3,3)



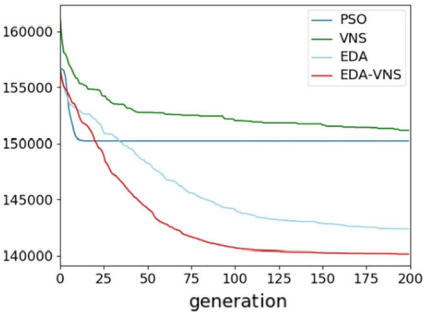

(21) Convergence curve for (200,3,4)




(22) Convergence curve for (200,6,2)

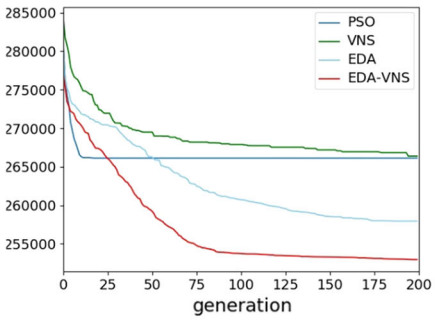



(23) Convergence curve for (200,6,3)

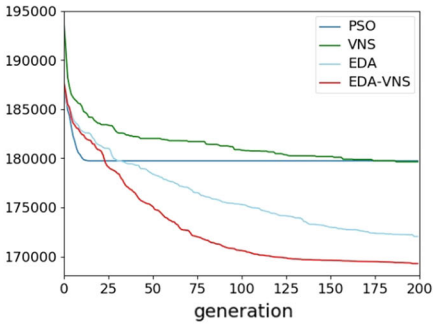



(24) Convergence curve for (200,6,4)

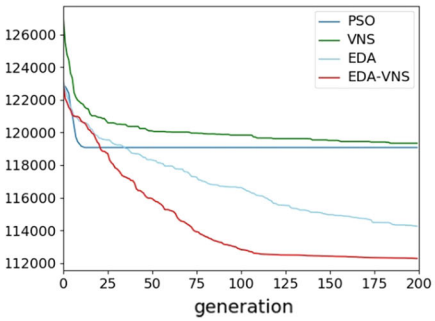
**Fig. 18** continued




(25) Convergence curve for (200,9,2)




(26) Convergence curve for (200,9,3)




(27) Convergence curve for (200,9,4)

**Fig. 18** continued

ness of EDA–VNS algorithm is remaining with the increase of the scale of problem. The proposed algorithm can obtain results within reasonable time. For example, the average running time is 17.72 s in the instance with 100 patients, 9 doctors, 2 examination servers and 2 treatment servers.

## 6 Conclusion

In this work, we investigate the patient flow in a typical ophthalmology clinic. We first build four patient routes to describe the general patient flow. Next, we develop a hybrid EDA–VNS algorithm which combines estimation of distribution algorithm and variable neighborhood search to allocate patients to servers. Patient-sequence rules are also proposed to sequence the allocated patients. Then, the comparisons of the algorithm with EDA, VNS, and popular algorithm PSO show the efficiency of the proposed algorithm.

In the future research, we can enrich our model and take complicated situations into account. The patient routes are known in advance in this paper. The uncertain of patient routes is also a significant problem. We can make more effort to forecast the patient routes, also we can set up a triage room to classify the patients to ensure their routes. In addition, we can try to solve the problem optimally or nearly optimally by using more efficient methods, such as bi-level optimization and optimal structural features.

**Acknowledgements** This work is supported by the Key research and development Projects in Anhui (1804b06020377), the Basic scientific research Projects in central colleges and Universities (JZ2018HGTB0232), the National Natural Science Foundation of China (Nos. 71601065, 71690235 and 71690230), and Innovative Research Groups of the National Natural Science Foundation of China (71521001). This paper is submitted to the special issue (CSoNet2018).

## References

Adibi MA, Shahrabi J (2014) A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem. Int J Adv Manuf Technol 70:1955–1961

Ahmadi-Javid A, Jalali Z, Klassen KJ (2017) Outpatient appointment systems in healthcare: a review of optimization studies. Eur J Oper Res 258:3–34

Armony M, Israelit S, Mandelbaum A et al (2015) On patient flow in hospitals: a data-based queueing-science perspective. Stoch Syst 5:146–194

Chakraborty S, Muthuraman K, Lawley M (2013) Sequential clinical scheduling with patient no-show: the impact ofpre-defined slot structures. Socioecon Plann Sci 47:205–219

Chen X, Wang L, Ding J, Thomas N (2016) Patient flow scheduling and capacity planning in a smart hospital environment. IEEE Access 4:135–148

Chern CC, Chien PS, Chen SY (2008) A heuristic algorithm for the hospital health examination scheduling problem. Eur J Oper Res 186:1137–1157

Choi SW, Kim YD (2008) Minimizing makespan on an m-machine re-entrant flowshop. Comput Oper Res 35:1684–1696

Danping L, Lee CKM (2011) A review of the research methodology for the re-entrant scheduling problem. Int J Prod Res 49:2221–2242

Hansen P, Mladenović N (2001) Variable neighborhood search: principles and applications. Eur J Oper Res 130:449–467

Huang J, Carmeli B, Mandelbaum A (2015) Control of patient flow in emergency departments, or multiclass queues with deadlines and feedback. Oper Res 63:892–908

Kim HW, Lee DH (2009) Heuristic algorithms for re-entrant hybrid flow shop scheduling with unrelated parallel machines. Proc Inst Mech Eng Part B J Eng Manuf 223:433–442

Kong M, Zhou J, Pei J et al (2019) A modified variable neighborhood search algorithm in distributed virtual manufacturing network. Optim Lett. https://doi.org/10.1007/s11590-019-01450-9

Larranaga P, Lozano JA (2001) Estimation of distribution algorithms: a new tool for evolutionary computation. Kluwer Press, Boston

Leeftink AG, Vliegen IMH, Hans EW (2017) Stochastic integer programming for multi-disciplinary outpatient clinic planning. Health Care Manag Sci 22:1–15

Lei D, Guo X (2016) Variable neighborhood search for the second type of two-sided assembly line balancing problem. Comput Oper Res 72:183–188

Li X, Wang J, Fung RYK (2018) Approximate dynamic programming approaches for appointment scheduling with patient preferences. Artif Intell Med 85:16–25

Liang B, Turkcan A, Ceyhan ME, Stuart K (2015) Improvement of chemotherapy patient flow and scheduling in an outpatient oncology clinic. Int J Prod Res 53:7177–7190

Lin CKY (2015) An adaptive scheduling heuristic with memory for the block appointment system of an outpatient specialty clinic. Int J Prod Res 53:7488–7516

Liu X, Lu S, Pei J, Pardalos PM (2018) A hybrid VNS-HS algorithm for a supply chain scheduling problem with deteriorating jobs. Int J Prod Res 56:5758–5775

Lu Y, Xie X, Jiang Z (2018) Dynamic appointment scheduling with wait-dependent abandonment. Eur J Oper Res 265:975–984

Pei J, Liu X, Fan W et al (2019) A hybrid BA-VNS algorithm for coordinated serial-batchingscheduling with deteriorating jobs, financial budget, and resource constraint in multiple manufacturers. Omega (UK) 82:55–69

Pei J, Wang X, Fan W et al (2018) Scheduling step-deteriorating jobs on bounded parallel-batching machines to maximise the total net revenue. J Oper Res Soc 5682:1–18

Puerto J, Pérez-brito D, García-gonzález CG (2014) A modified variable neighborhood search for the discrete ordered median problem. Eur J Oper Res 234:61–76

Salemi Parizi M, Ghate A (2016) Multi-class, multi-resource advance scheduling with no-shows, cancellations and overbooking. Comput Oper Res 67:90–101

Saremi A, Jula P, Elmekkawy T, Wang GG (2013) Appointment scheduling of outpatient surgical services in a multistage operating room department. Int J Prod Econ 141:646–658

Sayah A, Lai-Becker M, Kingsley-Rocker L et al (2016) Emergency department expansion versus patient flow improvement: impact on patient experience of care. J Emerg Med 50:339–348

Shen JN, Wang L, Zheng HY (2016) A modified teaching-learning-based optimisation algorithm for bi-objective re-entrant hybrid flowshop scheduling. Int J Prod Res 54:3622–3639

Taherkhani M, Safabakhsh R (2016) A novel stability-based adaptive inertia weight for particle swarm optimization. Appl Soft Comput J 38:281–295

Wang S, Wang L, Xu Y, Liu M (2013) An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time. Int J Prod Res 51:3778–3793

Yan C, Tang J, Jiang B, Fung RYK (2015) Sequential appointment scheduling considering patient choice and service fairness. Int J Prod Res 53:7376–7395

Zhang XY, Chen L (2018) A re-entrant hybrid flow shop scheduling problem with machine eligibility constraints. Int J Prod Res 56:5293–5305

Zhou B, Hu L, Zhong Z (2018) A hybrid differential evolution algorithm with estimation of distribution algorithm for reentrant hybrid flow shop scheduling problem. Neural Comput Appl 30:193–209