An Improved Genetic Algorithm for College Course Scheduling

Chenle Wang[1(✉)] and Bin Wang[2]

[1] International Engineering College, Xi'an University of Technology, Xi'an 710048, China
3202241003@stu.xaut.edu.cn

[2] School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China
wb@xaut.edu.cn

Abstract. The course scheduling is a NP-complete problem. At present, various intelligent optimization algorithms have provided many feasible solutions to the course scheduling problem in colleges and universities, with differentiated advantages and disadvantages. This work tries to design a general and efficient algorithm to solve the large-scale course scheduling. In particular, we analyze and model the problem of course scheduling in colleges and universities, and put forward the improved genetic algorithm to solve the problem of large course scheduling under various constraints. First, the teaching task number is stored in a two-dimensional time-class matrix, which represents the information of teachers, and a two-dimensional classroom matrix is established to store the classroom information of the class. Next, we apply the improved genetic algorithm to cross and mutate the time-class matrix to obtain new individuals, thereby adjusting the corresponding classroom matrix. Then, the excellent individual is selected from the parent and child generation, and iterated until the optimal individual is produced. Finally, experimental results show that the convergence speed of the proposed algorithm is faster and higher fitness values can be obtained.

Keywords: Genetic Algorithm · Scheduling Problem · Classroom Matrix

1 Introduction

Genetic Algorithm (GA) is a randomized, efficient and adaptive search method inspired by the biological genetic mechanisms of survival of the fittest and survival of the fittest. Since the method was proposed by Professor J. Holland of the University of Michigan in 1975, it has been widely used in many fields, such as optimization problem solving, machine learning, data mining, artificial neural network training and intelligent control system, etc., because of its powerful global search and adaptive adjustment ability. Genetic algorithm is regarded as one of the key technologies in modern intelligent computing.

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 573, pp. 481–493, 2024.
https://doi.org/10.1007/978-3-031-65126-7_40

The core idea of genetic algorithm is to realize efficient search of solution space by simulating natural selection, crossover and variation in the process of biological evolution. Different from traditional numerical optimization methods, genetic algorithm does not depend on the derivative property of the problem and the continuity of the function, so it has higher robustness and applicability. In genetic algorithms, each solution in the solution space is encoded as a chromosome and then evaluated by a fitness function. Based on these evaluation results, the algorithm performs selection, crossover and mutation operations in the solution space to generate a new generation of solutions, and iterates constantly to gradually approach the global optimal solution.

In genetic algorithms, the initial solution population consists of n binary strings, called chromosomes. The binary bits in each chromosome represent genes, which together describe the characteristics of a solution. Genetic algorithm consists of three main operations: selection, crossover and mutation.

2 Preliminaries

In 1963, Gotlibe proposed a mathematical model for the problem of class scheduling [1]. In 1976, S. Even proved for the first time that the scheduling problem is NP-complete [2]. Since scientists do not have an algorithm for solving NP-complete problems, they focus on specific practical problems. In 2000, E. K. Burke and A. J. Smith proposed a hybrid meme algorithm combining local search operators with genetic algorithm, which can better solve scheduling problems such as course scheduling [3]. In 1983, Kirkpatrick et al. proposed the simulated annealing method [4]. Although it has achieved certain results, the simulated annealing algorithm has some shortcomings, such as slow convergence speed, long execution time, algorithm performance is related to the initial value and parameter sensitivity. William Hoiles and Mihaela van der Schaar proposed a personalized course scheduling algorithm based on UCB in 2016, using recorded student data to generate featured course recommendations and course plans [5]. In 2019, Imran Hossain proposed an optimization method for university course scheduling based on particle tour [6]. Indian scholar Arabinda Tripathy proposes to arrange classes based on people and adopt the method of multiple class groups to deal with conflicts [7]. Canadian scholars Jean Aubin, Jacques A. Ferland, Charles Fleutent and others divided the course scheduling problem into schedule problem and grouping problem. Then the SAPHIR course scheduling decision support system, which includes several modules such as data processing, automatic optimization and interactive optimization, is developed [8]. In 1984, Lin Zhang xi and Lin Yao rui published the experimental research result “Application of Artificial Intelligence Technology in Curriculum Scheduling” [9]. They used artificial intelligence technology to conduct heuristic search for and or graphs generated in the curriculum scheduling, and successfully arranged the curriculum according to the two sets of data. Nanjing Institute of Technology developed UTSS (A University Timetable Scheduling System) [10]. The course scheduling system of these universities is designed for the curriculum and teaching resources of the university, and cannot be widely promoted. There are many algorithms to realize the scheduling problem, and there are roughly the following types of algorithms: integer programming algorithm, graph theory based algorithm, heuristic algorithm, genetic algorithm, simulated annealing method and ant colony algorithm.

3 Mathematical Model of Scheduling Problem

The Various Entities That Appear in the Scheduling Problem

(1) Class: In college education, classes of different majors may be arranged to take the same course.
(2) College: A university college is a teaching and research institution within a university divided by discipline and specialty. Teachers, courses and classes are all affiliated to the college.
(3) Room: There are many types of classrooms, such as multimedia classrooms, speech classrooms and outdoor venues. The description of the classroom must include the classroom name, building, classroom capacity, and classroom type.
(4) Teacher: A teacher is a lecturer who imparts knowledge to students. The description of teachers includes teacher number, teacher rank, teacher name, teacher gender and other factors.
(5) Task: A course is a series of teaching tasks arranged by the school. Courses have attributes such as course number, course name, teaching period, and school and course type.
(6) Teaching Schedule: Teaching schedule is one of the core elements of course scheduling, which involves the course arrangement required for all classes in a semester, including course name, course number, starting college, grade major and other information.
(7) Time period: The class time is divided into a period according to certain rules. For example, 8:00–10:00 on Monday is the first period, and 16:00–18:00 on Friday is the 20th period.

Hard Constraints That Must be met for Scheduling Problems

(1) A teacher can only arrange one course at a time;
(2) A student can only take one course at a time;
(3) A classroom can only arrange one course at a time;
(4) There is only one type of classroom required for a course;
(5) Must conform to the school's teaching plan, and complete a certain class hour and examination or examination within the specified time.
(6) The number of students in the classroom should be less than the capacity of the classroom.

Soft Constraints

(1) Each course should be arranged at appropriate intervals during the week;
(2) The courses of high difficulty will be arranged in a better time period, and the courses of low difficulty will be arranged in other time periods. The physical education class will be arranged in 3–4 classes every day, or in the afternoon, and there will be no other classes after that;

(3) The teaching of each course should be arranged in the same classroom every week;
(4) The number of students in the classroom should be as close as possible to the capacity of the classroom.

Mathematical Models of Constraints

This article will class - class, the teacher Teaching events (would Event), the Teaching events according to the Teaching schedule. $TE = \{(c_a, s_b, t_c\}) \mid c \in C, s \in S, t \in T, a \in cnum, b \in snum, c \in tnum$, (c,s,t) satisfies the "teaching arrangement"}. Will constitute the classroom - time corresponding space-time events (Spatiotemporal Event), namely, $SE = R \times M = (r_1, m_1), (r_1, m_1), (r_1, m_1), (r_1, m_1), \ldots, (r_{rnum}, m_{mnum})$. Cnum is the total number of classes, gnum is the total number of colleges, rnum is the number of classrooms, tnum is the number of teachers, snum is the total number of courses, mnum is the total number of time ranges.

The space-time event is set as a random queue, and when a teaching event needs to be processed, a teacher-time is selected from the space-time event as the class time and place, that is, $TE \cap SE = 1$ should be guaranteed, and if there is no such correspondence, $TE \cap SE = 0$.

The hard constraints include classroom capacity, teacher conflict, curriculum conflict, etc., while the soft constraints include course time allocation, etc. For these constraints, the corresponding mathematical model is established:

(1) Each teacher may only teach in one classroom during a specified period of time.

$$\forall t \in T, m \in M, |\{r \in R : (t, m, r)\}| \leq 1 \tag{1}$$

For any teacher t belongs to set T, and any time period m belongs to set M, that is, given teacher t and time period m, the number of times that teacher t teaches in different classrooms r cannot exceed 1, that is, each teacher can only teach in one classroom within a time period.

(2) Teachers may only teach courses related to their respective faculties.

$$\forall t \in T, s \in S : \text{ Teaching } s \Rightarrow G(t) \in G(s) \tag{2}$$

For any teacher t to belong to set T, and any course s to belong to set S, if a teacher t teaches course s (represented by Professor t s), then the faculty to which the teacher t belongs must be in the set G(s) corresponding to the course s. This means that teachers can only teach courses related to their school.

4 Improve the Genetic Algorithm of College Course Scheduling Algorithm

4.1 Improvement Genetic Algorithms Course Scheduling Problem

This article sets up a matrix where each row represents a time period and each column represents a class. The class-course-teacher combination is regarded as a "teaching event", and each teaching event is represented by a teaching task number, which is

inserted into the matrix, which is called the "curriculum matrix". At the same time, another unique time-class matrix is used to store the classroom numbers, which is called "classroom matrix" in this paper. The two matrices above are called an individual in a population. Since the above matrix can only store the class-course-teacher arrangement, which is called the "classroom matrix" in this paper, the matrix does not contain the time-classroom arrangement, so another matrix is needed to store the time-classroom information, and the position relationship in the matrix corresponds to the course matrix.

In the optimization process of course scheduling by genetic algorithm, the concepts of Population and Individual need to be introduced, and the individual is a class schedule, and the population is a collection of multiple classes with different contents and the same structure. By definition, the curriculum matrix and classroom matrix correspond to individuals in the genetic algorithm, and the matrix set corresponds to the population. (hereinafter referred to as a class of matrix schedule individual population1, population2, … Populationi), a classroom scheduling matrix called classroom individuals (room1, room2, … Roomi), the set of course matrices is called the class schedule population, and the set of classroom scheduling matrices is called the room population.

Algorithm 1: Framework of the course scheduling algorithm.

Input: N, population size; CR, cross factor; F, mutation operator, GEN ,The maximum number of generations ;
Class_num, number of class; T, Time period; teaching_task_data, Teaching task information;
Room_data, Classroom information

Output: A series of best solutions

```
Initialize population ;
while the termination conditions are not met do
    Teacher teaching conflict handling;#Algorithm 2
    Classroom conflict handling;#Algorithm 3
    Mutation;#Algorithm 4
    Crossover;#Algorithm 5
    Selection Strategy;#Algorithm 6
end while
```

4.2 Teacher Teaching Conflict Management and Classroom Conflict

In the process of teacher teaching conflict processing, each row of the individual population is read to find the teacher number corresponding to each teaching task number in the row, and the teacher number is read successively. When there is a duplicate teacher number, the teaching task number corresponding to the duplicate teacher number is exchanged with the teaching task number randomly selected in this column, and the classroom number corresponding to the two teaching task numbers is exchanged at the same time.

In the classroom conflict processing, each line of the individual room is read to find whether there is the same classroom number in the row. If there is a duplicate classroom number, the new classroom number is re-assigned to the following number to ensure

that the capacity of the new classroom number is greater than or equal to the number of teachers.

Algorithm 2: Teacher Teaching Conflict Management

Input: N ,population size; Class_num, number of class; T, Time period; population, Class schedule population; room, classroom population

Output: population and room

```
flag=1; #If flag is equal to 1, there is a teaching task number conflict. Otherwise, no conflict
while flag==1
  flag=0;
  for i=1:N
    for row=1:T
      If a teacher repeats the class in the same period, adjust the individual population's teaching
      time and adjust the Classroom number of individual room at the same time;
      flag=1;
    end for
  end for
end while
```

Algorithm 3: Classroom Conflict Handling Algorithm

Input: N, population size; Class_num ,number of class; T, Time period; room, classroom population

Output: room

```
flag=1;#If flag is equal to 1, there is a classroom number conflict. Otherwise, no conflict
while flag==1
  flag=0;
  for i=1:N
    for row=1:T
      Find if the row has the same classroom number, if there is a duplicate classroom number,
      the next number is reassigned to a new classroom number;
      flag=1;
    end for
  end for
end while
```

The Crossover Operation

The value of CR is 0.2. When col column of the course individual is accessed, the current row is row, the random value mask [0, 1] is set, the random number [1, 20] is taken and $i \neq$ row is taken. If mask $<$ CR, Then, the course individuals population (row, col) and population (i, col) are exchanged, and the above exchange operation is performed for each data in the col column in turn. After the individual mutation is completed, the rows of population and room matrix are re-processed for conflict.

Algorithm 4: The crossover operation

Input: population ,Class scheduling individual ;roomi, Classroom individual ;Class_num, Number of classes; T, Time period; CR, cross factor;

Output: populationi and roomi

```
for col=1:class_num
    for row=1:T
        mask = rand(0,1);
        r = rand(1, 20)and r≠row;
        if mask < CR then
            Swaprow(populationi, row, r, col);
            Swaprow(roomi, row, r, col);
        end if
    end for
end for
Conflict handling for populationi; #Algorithm 2
Conflict handling for roomi; #Algorithm 3
```

The Mutation Optimization

We choose F with a value of 0.5, and when accessing an individual, when traversing the columns, randomly generate a number rand data between [0, 1]. If randnum $<$ F, the current column is swapped with the same position column of a randomly selected individual; If randnum $\geq$ F, no exchange operation is performed. At the same time, the same position of the classroom matrix room is exchanged to maintain the consistency of the course arrangement and the classroom arrangement.

Algorithm 5: The mutation operation

Input: N, population size; F, mutation operator, Class_num, number of class; p, Parent class schedule population; r, Parent classroom population;

Output: p_new, Offspring Class schedule population ;r_new, Offspring classroom population

```
for i=1:N
    for col=1:class_num
        p_new = rand(p) % N;
        while p_new == i
            p_new = rand() % N;
        end while
        randnum = rand(0,1);
        if randnum < F then
            swap(p[i][j], p_new[i][j]);
            swap(r[i][j], r_new[i][j]);
        end if
    end for
end for
```

4.3 The Fitness Function

The fitness function plays a key role in evaluating the strengths and weaknesses of individuals in a population. According to the soft constraints related to the course scheduling problem in colleges and universities, this paper adopts the weighted method to construct the fitness function to ensure the balance performance of the algorithm under various conditions. The fitness function F(s) is designed as follows:

$$F(x_i) = \sum_{i=1}^{n} \alpha_i s_i \tag{3}$$

where, x_i represents an individual (that is, a course scheduling solution), n represents the number of soft constraints and the weight coefficient of a certain soft constraint in the fitness function, reflecting the relative importance of the condition in the optimization process. It represents the score of the individual under the soft constraint condition, which measures the performance of the individual in meeting the soft constraint condition, and s should also satisfy the following constraints:

$$\begin{cases} \sum_{i=1}^{n} \alpha_i = 1 \\ s \in [0, 100] \end{cases} \tag{4}$$

In practical application, we can adjust the weight coefficient α and the scoring method according to the characteristics of the problem and the actual demand, so as to better meet the goal of course scheduling optimization. The weights assigned to the constraints in this paper are 0.4, 0.4, 0.1 and 0.1 respectively.

The evaluation function of soft constraints is defined as follows:

Algorithm 6: Selection Strategy

Input: p, Parent class schedule population; r, Parent classroom population ;p_new, Offspring class schedule population ; r_new, Offspring classroom population ;F(),Evaluation function

Output: p_new and r_new

```
p =p∪ p_new;
index=Sort(p, F(p));
index= 1;
for i in index
    p_new[i]=p[i];
    r_new[i]=r[i];
end for
```

Course Preference Evaluation

Courses in a day should be arranged as far as possible at a time with a high expectation of the time period.

This article assigns class Time Expectations to different time periods. Morning time slots are assigned higher course expectations, while afternoon time slots are assigned lower course expectations. In order to evaluate the degree to which a schedule satisfies this constraint, the expected values corresponding to the time periods of all the courses

in the schedule are summed. Finally, this value is used as the grading index of the class schedule under the constraint conditions.

$$S_1(x) = \frac{1}{n}\sum_{i=1}^{n} a_m, \quad m \Rightarrow x_i \tag{5}$$

$S_1(x)$ is the evaluation function of the soft constraint condition, and x is a definite class schedule, $m \Rightarrow x_i$ representing the period m corresponding to the class i of the class schedule x. That is, the weights corresponding to the time periods of all courses in a certain class schedule are added successively and then divided by the total number, and the average value obtained is the result of the evaluation function.

Algorithm 7: Course preference evaluation algorithm

Input: population ,class schedule individual ;roomi, Classroom individual ;Class_num ,Number of classes ; T, Time period ;Ce ,Curriculum expectation

Output: result ,Class schedule evaluation value

```
result = [0,0,...];
for col=1:Class_num
    s=0;
    for row=1:T
        if populationi[row][col]!=0
            s+=Ce[row];
        end if
    end for
end for
result. append(s);
```

Course Interval Weight

For the same course, it should be reasonably distributed within a week to avoid crowded or distant courses.

$$S_2(x) = \frac{1}{snumber_x}\sum_{i=1}^{snumber} b_s \tag{6}$$

$S_2(x)$ is the evaluation function of the soft constraint condition, x is a definite curriculum schedule, *snumber* is all the courses in the curriculum schedule, and note that the courses that appear multiple times are counted as one course, which is counted according to the course number. The value of the evaluation function is calculated by adding the weights of all courses according to the contents of the table and then calculating the mean value.

In order to measure the uniformity of the weekly course distribution, the phenomenon of course crowding on several days of the week is avoided. In this paper, the standard deviation of course arrangement is used to judge the degree of dispersion of courses. The higher the standard deviation, the more crowded the courses, the lower the fitness

function score, which is

$$\overline{c} = \frac{1}{wday} \sum_{i=1}^{wday} c_i \tag{7}$$

$$\sigma_c = \sqrt{\frac{\sum_{i=1}^{wday} (c_i - \overline{c})^2}{wday}} \tag{8}$$

σ_c is the standard deviation of the dispersion of an individual course, c_i is the sum of the number of courses on the i day, $\overline{c}$ is the average of the number of courses on the day, $wday$ corresponds to the sum of the working days of a week, which in this paper refers to Monday to Friday, so $wday = 5$.

For example, when a class has 5 classes a week, if the 5 classes are evenly distributed between Monday and Friday, $\overline{c} = 1, \sigma_c = 0$, which means that the class schedule is evenly distributed; if 4 classes are scheduled on Monday and 1 class is scheduled on Tuesday, $\overline{c} = 1$, $\sigma_c \approx 1.55$, which means that the class schedule is overcrowded. Therefore, the smaller the standard deviation, the higher the excellence of the individual, so a fixed value can be subtracted from the standard deviation and then amplified, that is, the score of this curriculum under this soft constraint condition.

Algorithm 8: Course Interval Weight Algorithm

Input: pi, class schedule individual ;ri ,Classroom individual ;Class_num ,Number of classes;

Output: result ,Evaluation value

```
result = [0,0,...];
for col  1:Class_num
    s=0;
    taskno=pi(:,col);       #Read the teaching task number of each class
    timeno=find(taskno);#Read the non-0 teaching task number
    ut=unique(taskno(timeno)); #ut is a non-repetitive teaching task number
    m=length(ut); #m is the number of classes per week per teaching task number
    for   k1=1:m
            tib=find(ut(k1)==taskno);#tib is the number of classes per week
            if length(tib)==1          #The course meets once a week
                Add the corresponding weights to s
             else if length(tib)==2#The course meets two times a week
                Add the corresponding weights to s
            else if length(tib)==3#The course meets three times a week
                Add the corresponding weights to s
            end if
    end for
   result[col]=result[col]+s;
end for
```

5 Experimental Results

In this paper, there are 220 classes, 300 classrooms, 700 teachers and 550 courses in four grades in a college. There are 20 classes per week, and each time period is 2 h. The initial population was set to 50. The traditional genetic algorithm and the improved genetic algorithm were used to iteratively evolve 1000 generations, and 10 experiments were conducted. The average value of the optimal individual fitness value was obtained for every 100 generations of evolution. The experimental comparison results were shown in the Fig. 1. The average fitness value of the improved genetic algorithm is higher than that of the traditional genetic algorithm, and the convergence is good. The course scheduling algorithm in this paper is effective in solving the course scheduling problem.

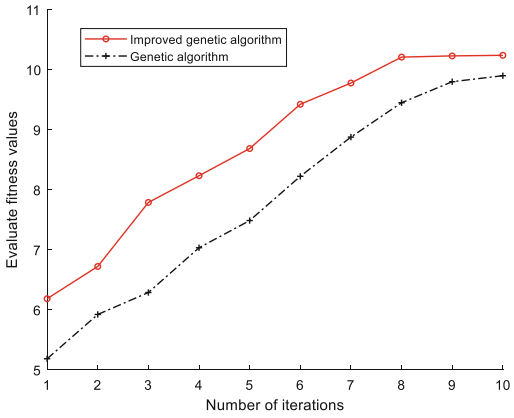

Fig. 1. Optimal course schedule

As shown in the Table 1, basic professional courses are usually arranged in the morning, there are two classes a week with reasonable intervals, and the classes held three times a week are distributed on Monday, Wednesday and Friday. In short, the course distribution is even and reasonable.

Table 1. A Class schedule.

Weekly/session	Monday	Tuesday	Wednesday	Thursday	Friday
The first class (Lesson 1–2) 08:00–09:50	Operating system 6-310 Teacher 10	Software architecture and framework 6-408 Teacher 12	Principles of Compiler Construction 6-301 Teacher 8	Software testing technique 6-310 Teacher 15	Principles and applications of microcomputer 6-401 Teacher 9
The second class (Lesson 3–4) 10:10–12:00	Principles of Compiler Construction 6-301 Teacher 8	Software testing technique 6-310 Teacher 15	Operating system 6-310 Teacher 10	Software project management 6-412 Teacher 17	Principles of Compiler Construction 6-301 Teacher 8
The third class (Lesson 5–6) 14:10–16:00	Principles and applications of microcomputer 6-401 Teacher 9	Software project management 6-412 Teacher 17	Principles and applications of microcomputer 6-401 Teacher 9		Software architecture and framework 6-408 Teacher 12
The fourth class (Lesson 7–8) 16:10–18:00					

6 Conclusion

On the basis of studying the problem of college course scheduling, this paper establishes a mathematical model of college course scheduling, and uses improved genetic algorithm to solve the problem of college course scheduling. In this paper, a time-class "course matrix" and a "classroom matrix" are taken as an individual. In the "course matrix", a class-course-teacher "teaching event" is stored, which is represented by the teaching task number. Meanwhile, another unique time-class matrix is used to store the classroom number. The genetic variation operation is carried out on N individuals of the "course matrix", the fitness function is used to select, and the optimal solution is obtained after many iterations. The experiment verifies that the improved genetic course scheduling algorithm proposed in this paper can effectively solve the problem of efficient course scheduling.

References

1. Gotlieb, C.C.: The construction of class-teacher time-tables. J. Commun. ACM **5**(6), 73–77 (1962)
2. Even, S., Itai, A., Shamir, A.: On the complexity of timetable and multicommodity flow problems. SIAM J. Comput. **5**(4), 691–703 (1976)

3. Burke, E.K., Smith, A.J.: Hybrid evolutionary techniques for the maintenance scheduling problem. J. IEEE Trans. Power Syst. **15**(1), 122–128 (2000)
4. Kirpatrick, S, Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. In: Readings in Computer Vision, pp. 606–615 (1987)
5. Hoiles, W., Schaar, M.V.D.: Bounded off-policy evaluation with missing data for course recommendation and curriculum design. In: The 33rd ICML, New York, pp. 1596–1604 (2016)
6. Hossain, S.I., Akhand, M.A.H., Shuvo, M.I.R., et al.: Optimization of university course scheduling problem using particle swarm optimization with selective search. J. Expert Syst. Appl. **127**(9), 24 (2019)
7. Tripathy, A.: Computerized decision aid for timetabling—a case analysis. J. Discret. Appl. Math. **35**(3), 313–323 (1992)
8. Ferland, A.J.A.: A large scale timetabling problem. J. Comput. Oper. Res. **16**(1), 67–77 (1989)
9. Lin, Z., Lin, Y.: The application of techniques of artificial intelligence to timetable scheduling. J. Tsinghua Univ. **24**(2), 1–9 (1984)
10. Nengbin, W.: UTSS – a university timetable scheduling system. Chin. J. Comput. (1984)