STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **4**, June 2016, pp 147–153.
Published online in International Academic Press (www.IAPress.org)

Memetic Algorithm and its Application to the Arrangement of Exam Timetable

Wenhua Huang *, Guisheng Yi, Sulan He

College of Mathematics and Information Science,JiangXi Normal University, China

(Received: 13 January 2016; Accepted: 15 February 2016)

Abstract This paper looks at Memetic Algorithm for solving timetabling problems. We present a new memetic algorithm which consists of global search algorithm and local search algorithm. In the proposed method, a genetic algorithm is chosen for global search algorithm while a simulated annealing algorithm is used for local search algorithm. In particular, we could get an optimal solution through the .NET with the real data of JiangXi Normal University. Experimental results show that the proposed algorithm can solve the university exam timetabling problem efficiently.

Keywords Memetic Algorithm, Timetabling, Genetic Algorithm, Simulated Annealing Algorithm

AMS 2010 subject classifications 68W01

DOI: 10.19139/soic.v4i2.190

1. Introduction

Timetabling problems is often encountered in university education institutions, such as university course and exam timetabling. It is a typical combinatorial optimization problem of a non-polynominal-time complicity [1]. Because of its practicability and complexity, it has become one of the hottest issues that scholars focus on. University exam timetabling problem is a typical example of the timetabling problem [2]. The constraint conditions are mainly composed of five objects, which are students in the class, invigilators, exam rooms, exam subjects and exam timeslots. Each object has a corresponding attribute, which reacts with each other from the constraint conditions. In general, the constraints are classified into two types: hard constraints and soft constraints [3]. The exam timetabling problem requires a set of examinations to be allocated to timetable slots. In making these allocations, the hard constraints of the problem must be satisfied and the number of soft constraints violated minimized [4].

Hard constraint condition: The conditions that must be met in the process of arranging the exam time. The feasible solution to the corresponding exam arrangement which can satisfy all hard constraints can be executed smoothly and will not lead to conflict of resources. i.e. It can meet the following basic conditions:

1) Only arrange one test subject to every student in a certain period.

2) One test subject in the exam time interval only for one timeslot of exam.

3) The number of students in the exam room can not exceed the total number of the exam room at the same exam timeslot.

4) All the students' test subjects have been arranged for the exam timeslot.

Soft constraint condition: The constraints that should be met in the exam. If not satisfied with these conditions, the solution may be still a feasible solution, but not preferred, whether a solution is feasible depends on the hard

*Correspondence to: Wenhua Huang (Email: huangwenhua-612@163.com). College of Mathematics and Information Science, JiangXi Normal University. 99 Ziyang Road, NanChang, Jiangxi Province, China (210093).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

constraint condition. Usually, soft constraint conditions including students' exam time interval (each student's exam time interval is stretched as far as possible, and tries not to squeeze together) and the invigilators requirements (the number of teachers in each period is as uniform as possible, so as to facilitate arrangements) etc. In this paper, the main consideration is given to the students' exam time interval.

What makes it even harder to arrange exam is the wide variety of courses which are contained in the college exam, the complexity of students' elective courses, as well as the freedom of students to choose their appropriate elective course. With so many constraints to take into consideration, it is rather difficult to find a better arrangement scheme only by manual scheduling. The usual methods of solving timetabling problems are mainly *genetic algorithm* (GA), heuristic graph coloring method, constraint logic programming, simulated annealing algorithm, integer programming, branch and bound techniques and tabu search algorithm [5, 6, 7, 8, 9, 10, 11]. These algorithms of low success rate, time-consuming and difficult to get a better solution. However, the genetic algorithm is a global search algorithm and also improves the success rate, but easy to fall into local optimum resulting in premature. Researchers and practitioners have shown that a combination of global and local search is almost always beneficial. So we propose to use memetic algorithm to solve timetable problem.

In this paper, we use the GA to carry out the *global search* (GS) to explore the outstanding areas of the solution space and combined with simulated annealing algorithm for *local search*(LS) to jump out of local optimal solution, so as to obtain the global optimal solution quickly. As experimental results show, this method can solve the problem of university exam schedule quickly and efficiently. This paper is organized as follow: section 2 explains details of proposed algorithm, section 3 describes the details of the exam course arrangement, section 4 presents the experimental results, and section 5 shows the conclusion.

2. The design of memetic algorithm

Memetic algorithm is one of the metaheuristic techniques commonly used for solving timetabling problems. In this paper, a new memetic algorithm is proposed for the problem of university exam schedule. We suppose that the number of exam subject is m , the exam timeslot n , student class p , exam room q , and student s .

2.1. Auxiliary Information

Considering that every university has their own elective course exam, so we can not judge whether there is a conflict among the subjects simply according to the corresponding relationship among classes and subjects, The way to determine whether the mutual conflicts between the exam subjects is according to the students' elective course. If one of the students chooses both subject A and subject B, then subject A and subject B are conflicting subjects, and so on. Then, check all the students' selective course and store the conflict information in the $Subject\_Subject$ matrix. Defined as (1):

$$Subject\_Subject[i,j] = \begin{cases} 1, If\ the\ subjects\ i\ and\ the\ subjects\ j\ are\ conflict \\ 0, If\ the\ subjects\ i\ and\ the\ subjects\ j\ are\ not\ conflict. \end{cases} \tag{1}$$

It is similar to the following matrix (2):

$$Subject\_Subject_{mxm} = \begin{pmatrix} 0 & 1 & \cdots & 1 & 0 & 0 \\ 1 & 0 & \cdots & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 1 & \cdots & 1 & 0 & 1 \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix} \tag{2}$$

2.2. Encoding

When it comes to encode the timetable problem, Matrix is a simple and effective way to encode timetable problem. Here, we will present a feasible solution intuitively in the form of sparse matrix, therefore every feasible solution in the solution space can be expressed as (3):

$$Timetable_{(m+1)X(n+1)} = \begin{pmatrix} 0 & 1 & \cdots & 0 & 0 & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & 0 & 1 \\ 0 & 1 & \cdots & 0 & 0 & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 1 \\ 12 & 2 & \cdots & 5 & 10 & 3 & 0 \end{pmatrix} \tag{3}$$

Where the number of subjects is represented as m, n is expressed as the number of exam timeslots.

$$Timetable[i,j] = \begin{cases} 1, If\ the\ exam\ (i)\ is\ arranged\ in\ the\ time\ (j)\ , i \in [1,m], j \in [1,n] \\ 0, If\ the\ exam\ (i)\ is\ not\ arranged\ in\ the\ time\ (j),\ i \in [1,m], j \in [1,n]. \end{cases} \tag{4}$$

In (3), row $m+1$ is used to store the number of exam rooms currently occupied, and column $n+1$ is used to identify whether the corresponding subject has been arranged for the exam timeslot, as can be seen from (4), if arranged, the value is 1, otherwise it is 0.

We can define the exam arrangement for each timeslot as a genetic locus, thus, each chromosome consists of n genetic locuses, as Figure 1:

Figure 1. the definition of chromosome

2.3. Initializing the population

In the process of arranging subjects, we need to determine whether one exam subject can be arranged in the specified timeslot. Three basic conditions need to meet as following:

1) The current test subjects have not yet been scheduled for exam timeslot, that is $Timetable[i, n+1] = 0$.

2) The test subjects arranged in the same timeslot do not conflict with each other, that is, the exam subjects needs to be arranged in the exam timeslot have no conflict with all exam subjects that have been arranged in the timeslot ,that is, $Subject\_Subject[i,j] = 0$.

3) The number of vacant rooms in the current exam timeslot is larger than the number of exam subjects required to be arranged in the exam subjects c, that is, $c \leq q - Timetable[m+1, t]$.

In the process of population initialization, the algorithm generates the initial population with heuristic method based on the random initial feasible solution, which reduces the risk of conflict in the exam subjects greatly and improve the efficiency and quality of the initial population. For each individual, the following steps are included in the generation of the initial population:

1. Set the starting index for the timeslot to be minPos (Initialization to 0). The final index is maxPos (initialized to n). For each test subject randomly generating an arrangement of timeslot during the exam time can be arranged.

2. Determine whether the termination condition is satisfied, if satisfied, Initialize population end. Otherwise, the minPos t traversal, determine whether the subject has a schedule of timeslot, If it can be arranged, the corresponding value of Timetable is set to 1, At the same time, the corresponding row of the last column is also set to 1, and update the number of exam time in the exam time, Otherwise continue.

3. Set $minPos = t + 1$, A random number is generated within the [minPos,maxPos] and the value is assigned to the variable t, go to the step 2.

2.4. Global search strategy

The global search includes two operators, namely, crossover operator and mutation operator.

Crossover operator

Crossover operator is the process of combining the two generations of the chromosomes in the specified rules to produce a new generation. The sequential crossover strategy, which can inherit some genes of the original parent as well as integrate into another parent to help add diversity of next generation,thus improving the cross efficiency.

If the two parent individuals are X, Y, the concrete steps are:

Randomly generated two positions i, j, where $i, j \in [1, n]$, The cross section of the exam time between the first $i + 1$ and the j position of the parent individual. For each subject have been arranged in the timeslot is in the cross section t_1, Check which timeslot is arranged in the exam subjects t_2, And then determine whether the X individual exam subjects can change the exam timeslot from t_1 to t_2, if can transform, otherwise try to the next subject until all the subjects in the cross domain is traversed.

Mutation operator

Mutation operator is the operation of some of the genes that are randomly changed. According to the specified variation probability P_m, which is usually greater than 0 and less than 0.1. Mutation operator, through which will increase the diversity of population, remove the bad individuals, as well as enhance the fitness of individuals. The concrete steps are:

For each individual in the population to generate a random number between 0 and 1 named r; If $r < P_m$ then generate two random numbers t_1, t_2 randomly, where $t_1, t_2 \in [1, n]$. Determine whether the current individual exam timeslot t_1 in the exam subjects can be arranged in the exam timeslot t_2, if so, the exam timeslot transform from t_1 to t_2; otherwise, do not do any operation.

2.5. Local search strategy

There are lots of frequently-used local search algorithms, including simulated annealing method, mountain climbing method, conjugate gradient method, guided local search, greedy method, simplex optimization method, Newton and Newton method, etc. After trying many kinds of methods, the simulated annealing method is selected finally, which help adding the diversity of population, producing more excellent individuals. Moreover, it could help to speed up the convergence rate. The concrete steps are listed as follows:

1. Set the initial temperature, temperature variation coefficient and tolerance range.

2. Determine whether the current temperature is within the tolerance range, if yes, then the end; otherwise, generate two random number t_1, t_2 randomly, where $t_1, t_2 \in [1, n]$. Determine whether the current individual exam timeslot t_1 in the exam subjects can be arranged in the exam timeslot t_2, If can, the exam timeslot transform from t_1 to t_2.

3. Calculate the individual fitness value before and after the change of exam timeslot. If after change fitness value is higher or meet the Metropolis criterion, the after change individual will replace the before change individual. Otherwise cool down, and go to step 2.

3. Course arrangement based on memetic algorithm

3.1. The establishment of fitness function and its calculation method

The soft constraints which considered in this algorithm is that each students should avoid taking exams in the same day or every other day. Therefore, the method of conflict function were used here, to measure the fitness of individuals. Conflict function definitions are as (5):

$$Conflict(i) = w_1 * Sameday(i) + w_2 * Overday(i) \tag{5}$$

In (5), Conflict i is the conflict value of individual i (the greater the conflict value, the smaller the fitness value), w_1 said the conflict of values that many tests occur at the same day for a student, w_2 said the conflict of values that many tests is occur at the next day for a student. Sameday i said the certain number of students at the same day to participate in many exam subjects, Overday i said the certain number of adjacent days to take an exam of exam subjects.

The fitness function is defined as (6):

$$Fitness(i) = s/Conflict(i) \tag{6}$$

The students' exam subjects whose arrangement are conflict , as can be seen from (5), the more conflict subjects, the greater the conflict, yet the smaller the degree of adaptation. The goal of algorithm is to try to find a feasible solution, which has smaller values of conflict yet with higher adaptation.

3.2. Algorithm description

The flow chart of the memetic algorithm is as Figure 2:

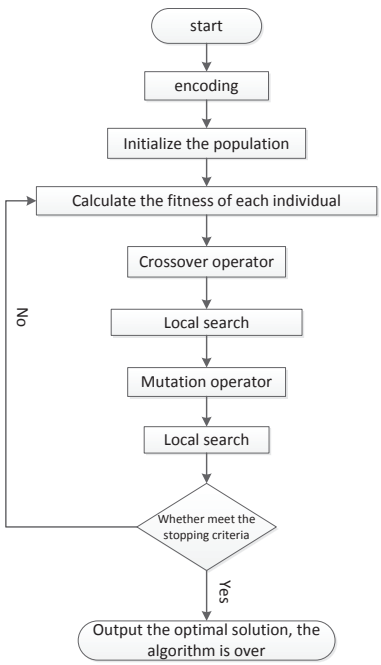

Figure 2. the flow chart of memetic algorithm

Among them, the stopping criteria can be a maximum number of iterations or fitness error range.

The main steps of the algorithm are:

1. A given population size M, the number of iterations N and m, n, Pc, Pm, $w1$, $w2$, p, q, generating initial population $G(t)$.

2. Calculate the fitness of each individual in the population, evaluate the fitness of each individual, and save the best individual to the next generation.

3. Crossover operation was carried out on the population, generating new individuals and calculating the value of fitness.

4. Carry out the local search operation to the population, using the simulated annealing algorithm to update the population.

5. Carry out the mutation operation to the population and generate new individuals, and calculate the fitness value of the population;

6. The same as step 4.

7. Determine whether satisfy the conditions precedent, if satisfied, then output the optimal solution and algorithm over, or turn to step 2.

4. Experimentation

In order to study the efficiency of the algorithm, we have carried out a simulation experiment on the college students' exam arrangement in $C\sharp.NET$ platform.This timetabling system has been tested on the real data from JiangXi Normal University. The college has 456 classes, 20081 students, 1397 courses. The total exam time is 10 days and each day contains four identical exam timeslots. Run the algorithm 10 times under different parameters. The configuration of the computer is INTEL CORE i5 2.6 GHZ, 4 GB of memory.

Experimental results show, it takes about 32 seconds to create initial population. With population size of 20, it takes about 58 seconds for each evaluation when hill climbing is used as local search algorithm. With the same population size, it takes about 43 seconds for each evaluation when simulated annealing algorithm is used. Therefore, we employ simulated annealing algorithm in our memetic algorithm. when $Pc = 0.8$ and $Pm = 0.10$, the optimal solution can be obtained, and the average fitness value of the best individual is 0.4803. Table 1 presents information related to the schedule arrangement under different parameters by using the memetic algorithm proposed in this paper, and Table 2 presents the same information by using GA. Compared the data of Table 1 with that of Table 2, the memetic algorithm has better solved the exam subjects conflict problem and a perfect exam time arrangement scheme has been obtained.

Table 1. Information related to the schedule arrangement under different parameters by using the memetic algorithm.

	Pc=0.7		Pc=0.8		Pc=0.9	
	Pm=0.05	Pm=0.10	Pm=0.05	Pm=0.10	Pm=0.05	Pm=0.10
Best conflict value	50961	50444	50246	41811	50125	48172
Best fitness value	0.3940	0.3981	0.3997	0.4803	0.4006	0.4169
Iteration number	3	1	8	7	10	9

Table 2. Information related to the schedule arrangement under different parameters by using GA

	Pc=0.7		Pc=0.8		Pc=0.9	
	Pm=0.05	Pm=0.10	Pm=0.05	Pm=0.10	Pm=0.05	Pm=0.10
Best conflict value	61232	60521	61025	58263	62146	59846
Best fitness value	0.3279	0.3318	0.3291	0.3447	0.3231	0.3355
Iteration number	10	10	10	10	10	10

5. Conculsion and future works

The experimental results prove that the memetic algorithm developed in this paper helps solving the problem of the university exam arrangement effectively, and reducing the number of iterations at the same time. The global search and local search strategy not only could increase the breadth of the search, but also increase the depth of the search, which showing a certain advantage in solving the university exam timetabling problem.

REFERENCES

1. Q. Nguyen, Q. Ta, T. Duong, *A Memetic Algorithm for Timetabling*, Research Informatics Vietnam-Francophony, Can Tho, Vietnam,289-294, February, 2005.
2. E .Burke, D. Elliman, P. Ford, R. Weare, (1995). *Examination timetabling in british universities*: a survey.. Lecture Notes in Computer Science, 1153, 76-90.
3. S. Yang, S. Jat, (2011). *Genetic algorithms with guided and local search strategies for university course timetabling*. IEEE Transactions on Systems Man & Cybernetics Part C, 41(1), 93-106.
4. R. Qu, E. Burke, B. Mccollum, L. Merlot, S. Lee, (2009). *A survey of search methodologies and automated system development for examination timetabling*. Journal of Scheduling, 12(1), 55-89.
5. P. Boizumault, Y. Delon, L. Peridy, (1996). *Constraint logic programming for examination timetabling*. Journal of Logic Programming, 26(2), 217-233.
6. E. Burke, Y. Bykov, J. Newall, S. Petrovic, (2003). *A time-predefined local search approach to exam timetabling problems*. IIE Transactions, volume 36(6), 509-528.
7. J. Thompson, K. Dowsland, (1998). *A robust simulated annealing based examination timetabling system*. Computers & Operations Research, 25(7-8), 637-648.
8. R. Raghavjee, N. Pillay, (2011). *Using genetic algorithms to solve the South African school timetabling problem*. Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on (pp.286-292). IEEE.
9. S. Abdullah, H. Turabieh, B. McCollum, and E. K. Burke, *An Investigation of a Genetic Algorithm and Sequential Local Search Approach for Curriculum-based Course Timetabling Problems, in Proc*. Multildisciplinary International Conference on Scheduling:Theory and Applications (MISTA 2009), Dublin, Ireland (2009), pp. 727-731.
10. S. Abdullah, S. Ahmadi, E. Burke, & Dror, M. (2004). *Investigating Ahuja-Orlin's Large Neighbourhood Search for Examination Timetabling*. OR Spectrum (Vol.volume 29, pp.351-372(22)).
11. J.R. Koza, R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.