

Research Article

Qiang Zhang*

An optimized solution to the course scheduling problem in universities under an improved genetic algorithm

<https://doi.org/10.1515/jisys-2022-0114>

received February 24, 2022; accepted August 02, 2022

Abstract: The increase in the size of universities has greatly increased the number of teachers, students, and courses and has also increased the difficulty of scheduling courses. This study used coevolution to improve the genetic algorithm and applied it to solve the course scheduling problem in universities. Finally, simulation experiments were conducted on the traditional and improved genetic algorithms in MATLAB software. The results showed that the improved genetic algorithm converged faster and produced better solutions than the traditional genetic algorithm under the same crossover and mutation probability. As the mutation probability in the algorithm increased, the fitness values of both genetic algorithms gradually decreased, and the computation time increased. With the increase in crossover probability in the algorithm, the fitness value of the two genetic algorithms increased first and then decreased, and the computational time decreased first and then increased.

Keywords: genetic algorithm, coevolution, scheduling problem, universities

1 Introduction

With the development of the social economy and the introduction of compulsory education, more and more people are able to receive education, the population base receiving basic education has increased, and the number of people willing to receive higher education has also greatly increased [1]. As a result, the enrollment scale of universities becomes increasingly larger, and the limited educational facilities and teachers' resources on the campus can hardly meet the growing demand. Therefore, multi-campus operation and sharing of educational facilities and faculty resources among different campuses have become a way to solve the demand for education [2]. Although the sharing of teaching resources across multiple campuses can meet the growing demand, the increased teaching resources also make the management more difficult and require the coordinated adjustment of the sequence of course arrangements to maximize the use of limited resources [3]. Hossain et al. [4] proposed a particle swarm algorithm to solve the highly constrained university course scheduling problem and added a forceful swap operation with a repair mechanism to the particle swarm algorithm. The experimental results showed the effectiveness of the proposed method. Herawatie et al. [5] solved the course scheduling problem with a genetic algorithm and avoided local optimum with a replacement policy. The experimental results showed that the algorithm functioned well. Yang and Xie [6] proposed a course scheduling algorithm based on a particle swarm optimization algorithm, designed a model of university ideological and political course scheduling system based on this algorithm, and verified the effectiveness and usability of the system by experimental and test results. In this

* **Corresponding author: Qiang Zhang**, School of Hotel Administration, Zhengzhou Tourism College, Unit 1, Building 3, District A, Xingfu Gangwan, Chengdong South Road, Guancheng District, Zhengzhou, Henan, 450000, China, e-mail: qgi2zn@yeah.net

study, the course scheduling problem in universities was briefly introduced. A genetic algorithm was proposed to optimize the scheduling scheme. The genetic algorithm was improved by using the principle of coevolution. The novelty of this study is scheduling courses with a genetic algorithm instead of manpower and using coevolution to improve the performance of the genetic algorithm in finding the best solution. The advantage of using a genetic algorithm for scheduling courses is that it can give multiple scheduling solutions more quickly and select the best solution from them.

2 Course scheduling problem in universities

Higher education institutions teach students according to a course schedule. A scientific and standardized course schedule can maximize the mobilization and utilization of teaching resources, thus improving teaching quality. After excluding various restrictions, the scheduling problem can be regarded as a problem of arranging and combining different courses, and different arrangements and combinations will bring different teaching qualities [7]. However, in the actual scheduling problem, the optimization of the scheduling scheme is subject to various constraints, including the type of courses, the number of classes required for every course, the number of teachers, the size of classrooms, the number and size of classes, and the duration of courses.

In addition to the various restrictions mentioned above, the difficulties of the scheduling problem also include the large number of optional permutations and combinations caused by the large number of courses, class time slots, and teachers. Some of these permutations and combinations violate the restrictions, but even for the remaining combinations, the huge number makes it difficult to find a suitable scheduling solution by human effort. Therefore, this study uses a genetic algorithm to optimize the scheduling solution.

The factors involved in the mathematical description of the scheduling problem are:

$$\begin{cases} T = \{T_1, T_2, \dots, T_t\}, \\ M = \{M_1, M_2, \dots, M_m\}, \\ C = \{C_1, C_2, \dots, C_c\}, \\ R = \{R_1, R_2, \dots, R_r\}, \\ P = \{P_1, P_2, \dots, P_p\}, \end{cases} \quad (1)$$

where T_t refers to the t -th period, T refers to the set of time periods, M_m refers to the m -th course (there are z_m classes for this course), M refers to the set of courses, C_c refers to the c -th class (this class includes k_c students), C refers to the set of classes, R_r refers to the r -th classroom (this class can tolerate y_r people), R refers to the set of classrooms, P_p refers to the p -th teacher (this teacher has x_p courses to teach), and P is the set of teachers [8].

In the actual scheduling problem, the values of the factors involved in equation (1) will be restricted. To facilitate calculation, we default that every classroom can accommodate students in every class in equation (1). Restrictions for the scheduling problem are:

- (1) only one M_m can be taught by P_p at T_t ;
- (2) C_c can only take one M_m at T_t ;
- (3) only one M_m can be taught in R_r at T_t .

The above restrictions are converted into a mathematical form:

$$\begin{cases}
 V = \{V_1, V_2, \dots, V_q\}, \\
 f = f_1 + f_2 + f_3 = 0, \\
 f_1 = \frac{\sum_{i=0}^q \sum_{j=0}^q T_1}{2}, \\
 f_2 = \frac{\sum_{i=0}^q \sum_{j=0}^q T_2}{2}, \\
 f_3 = \frac{\sum_{i=0}^q \sum_{j=0}^q T_3}{2}, \\
 T_1 = \begin{cases} 1, & M_i \neq M_j, T_i = T_j, P_i = P_j, \\ 0, & \text{else,} \end{cases} \\
 T_2 = \begin{cases} 1, & M_i \neq M_j, T_i = T_j, C_i = C_j, \\ 0, & \text{else,} \end{cases} \\
 T_3 = \begin{cases} 1, & M_i \neq M_j, T_i = T_j, R_i = R_j, \\ 0, & \text{else,} \end{cases}
 \end{cases} \quad (2)$$

where V is the set of allocation events with the teacher, student, and course bound in the scheduling scheme, v_q is the q -th allocation event with the teacher, student, and course bound (teacher is an element of P , student is an element of C , and course is an element of M), T_1 , T_2 , and T_3 are the judgment values for the corresponding three restrictions (the value is 1 if the restriction is not met; otherwise, the value is 0), and f_1 , f_2 , and f_3 are the total judgement values of V for the above three restrictions. When the sum of f_1 , f_2 , and f_3 is 0, it indicates that there is no scheduling conflict [9].

3 Scheduling optimization based on the improved genetic algorithm

As we can see from Section 2, the goal of the course scheduling problem in universities is to make the scheduling scheme more reasonable without scheduling conflicts. Traditionally, manual course scheduling involves scheduling courses with high priorities or scheduling difficulty first, arrange time periods for every course according to the course hours, and arrange courses with lower priorities in the same way, and the goal of no conflict is eventually achieved through continuous backtracking and adjustment [10]. However, in universities with a large number of faculty, students, and course programs, the workload of course scheduling is heavy. Manual course scheduling for universities is tedious and more prone to scheduling conflicts. The traditional genetic algorithm [11] uses a single population for evolution. Small population size is ineffective in finding the optimal solution, while a large population size will lead to too much computation and difficult convergence. In addition, although the iterative process of the genetic algorithm will increase the number of solution candidates by mutation operation, it will still fall into the optimal local solution [12].

The traditional genetic algorithm was improved by multi-population coevolution, and the flow of the improved genetic algorithm for scheduling is shown in Figure 1.

Three initial populations of the same size are generated through coding. Every chromosome in the population represents one scheduling scheme. In general, the teaching schedule is based on a week-long cycle, and Saturday and Sunday are holidays, so the scheduling scheme spans from Monday to Friday, which means that the schemes represented by the chromosomes are for a week. The gene coding pattern of the chromosome is “time period code + classroom code + course code + class code + teacher code,” and non-conflicting plural segments following the above pattern constitute the scheduling scheme, i.e., a chromosome. The choice of gene locus code is not random. There are 25 time periods, so the time period code is chosen from “01, 02, 03, ..., 25.” The classroom code is chosen from the classroom codes owned by the school. The course code, class code, and teacher code are provided by the school [13]. The coding rules

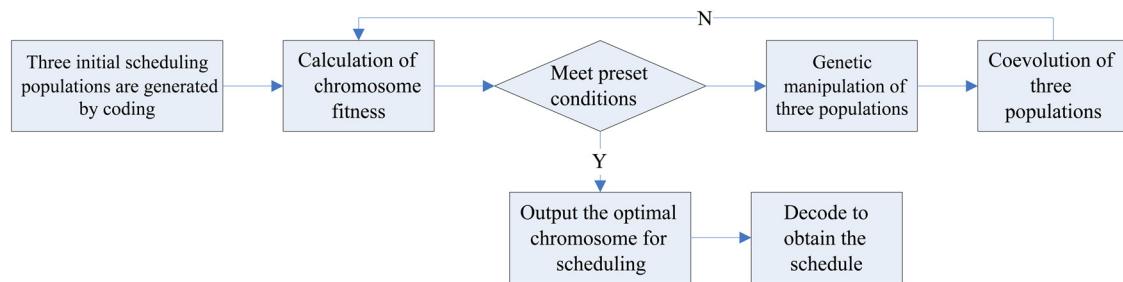


Figure 1: The improved process.

for chromosomes within the initial population are as described in Section 2. Under the premise of following the coding rules, the initial population is generated according to the custom of manual course scheduling. Step 1, the corresponding teacher codes of one class are added to fixed time periods. Then, the codes of other teachers of that class are added to different time periods randomly to obtain the initial course schedule of the class, and the initial class schedule of all classes is combined to form an initial chromosome. The plural chromosomes obtained as above constitute the initial population.

Step 2 The fitness of chromosomes in the population is calculated. Since the random generation of chromosomes, i.e., scheduling schemes, has already taken into account the problem of course conflicts and actively excluded conflicting schemes in the subsequent genetic operation, the main consideration in the calculation of fitness is to make the existing conflict-free scheduling schemes better, so the calculation formula of fitness is:

$$\begin{cases} g = \omega_1 g_1 + \omega_2 g_2 + \omega_3 g_3, \\ g_1 = \frac{1}{mn} \sum_{j=1}^m \sum_{i=1}^n \alpha_j t_i, \\ g_2 = \frac{1}{m} \sum_{j=1}^m \alpha_j d_j, \\ g_3 = \frac{1}{rc} \sum_{j=1}^r \sum_{i=1}^c \frac{k_i}{y_j}, \end{cases} \quad (3)$$

where g is the chromosome fitness, g_1 , g_2 , and g_3 are the course sequence priority, course duration discrete uniformity, and classroom utilization, ω_1 , ω_2 , and ω_3 are weights of g_1 , g_2 , and g_3 , α_j is the weight value (importance) of the j -th course, t_i is the sequence weight of the i -th period, and d_j is the schedule discrete degree of the j -th course.

Step 3 Whether the predefined conditions (reaching the maximum number of iterations or population fitness converging to stability) are met is determined based on the calculated population chromosome fitness. If the predefined conditions are met, the optimal chromosome is selected, and the chromosome is decoded to obtain the schedule. As shown in Figure 2, every row in this two-dimensional schedule represents a classroom, every column represents a course time period, and the intersection of the rows and columns is an array of courses, classes, and teachers, (M_m, C_c, P_p) .

	T ₁	T ₂		T ₂₅
Classroom 1				
Classroom 2				
.....				
Classroom r				

Figure 2: Two-dimensional course schedule.

The genetic operation is conducted when the population chromosome fitness does not meet the pre-defined conditions.

Step 4 In genetic iteration [14], crossover means randomly selecting two chromosomes according to the crossover probability and exchanging the gene fragment of the same position. In this study, the single-point crossover was used, i.e., only one gene at one gene locus was exchanged in one crossover. In addition, the conflict between the two chromosomes after crossover is determined using equation (2), and if a conflict occurs, another gene locus is randomly selected again for crossover until there is no conflict. Mutation means randomly selecting a chromosome according to the mutation probability and exchanging genes at the same type of gene locus until the mutated chromosome is conflict-free.

Step 5 After all three populations have undergone one genetic operation, the three populations are subjected to a coevolutionary [15] operation, as shown in Figure 3. The updated populations are arranged in ascending order by the value of the fitness function, and half of the worst chromosomes are replaced by half of the best chromosomes in the other population (order: population 1 $\rightarrow$ population 2 $\rightarrow$ population 3 $\rightarrow$ population 1). Then, the best and worst chromosomes in every population are selected, and the worst chromosomes are replaced by the best chromosomes.

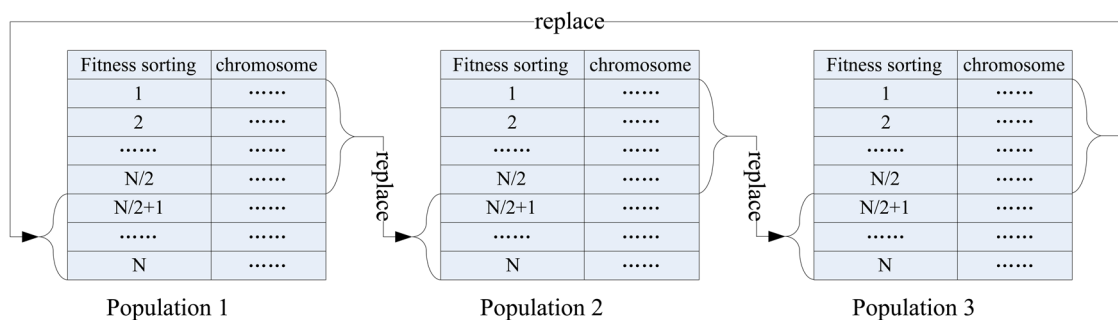


Figure 3: The operation flow of coevolutionary operation of three populations.

Step 6 Return to step 2 until one of the termination conditions is satisfied. There are two termination conditions. The first termination condition is that the number of iterations reaches the preset maximum number, avoiding the unnecessary computation time caused by difficult convergence. The second termination condition is that the fitness value of the algorithm converges to stable. The difference between the fitness value after the current time of iteration and the fitness value after the last time of iteration is calculated. The algorithm is considered as converging to stable when the difference reduces to within the preset range and no longer exceeds the preset range in the following 100 times of iterations.

4 Simulation experiments

4.1 Experimental setup

The actual university scheduling data are huge. Not only the number of courses, teachers, and classes is large [16], but also the variety of courses that teachers can teach is limited, which increases the difficulties in practice. Therefore, in this study, some of the data were selected to facilitate simulation experiments. The relevant experimental data were 20 classes, 25 classrooms, 27 teachers, 50 courses, 5 days a week, and 5 teaching time units per day.

The relevant parameters of the traditional genetic algorithm are as follows: the population size was set as 60, $\omega_1 = 0.5$, $\omega_2 = 0.3$, $\omega_3 = 0.2$, and the crossover and mutation probabilities were set as 0.6 and 0.1, respectively.

The relevant parameters of the improved genetic algorithm are as follows: the size of all 3 populations was 20, $\omega_1 = 0.5$, $\omega_2 = 0.3$, $\omega_3 = 0.2$, and the crossover and mutation probabilities were 0.6 and 0.1, respectively. The crossover and mutation probabilities used in the above genetic algorithm are parameters obtained through orthogonal experiments.

In addition to the above basic comparison experiments, this study also adjusted the crossover and mutation probabilities of the two genetic algorithms and compared the performance under different crossover and mutation probabilities. When the mutation probability was fixed at 0.1, the crossover probabilities were 0.4, 0.5, 0.6, 0.7, and 0.8, respectively; when the crossover probability was fixed at 0.6, the mutation probabilities were 0.1, 0.2, 0.3, 0.4, and 0.5, respectively.

4.2 Experimental results

Figure 4 shows the change in the average fitness value of population chromosomes during iterations of the traditional genetic algorithm and the improved genetic algorithm. It is seen from Figure 4 that the average fitness value of the populations of both genetic algorithms increased with the increase in the number of iterations and converged gradually. The improved genetic algorithm converged to a stable fitness value (48) after about 750 iterations, and the traditional genetic algorithm converged to a stable fitness value (44) after about 900 iterations. It was found that the improved genetic algorithm converged faster and yielded better chromosomes, i.e., better scheduling solutions.

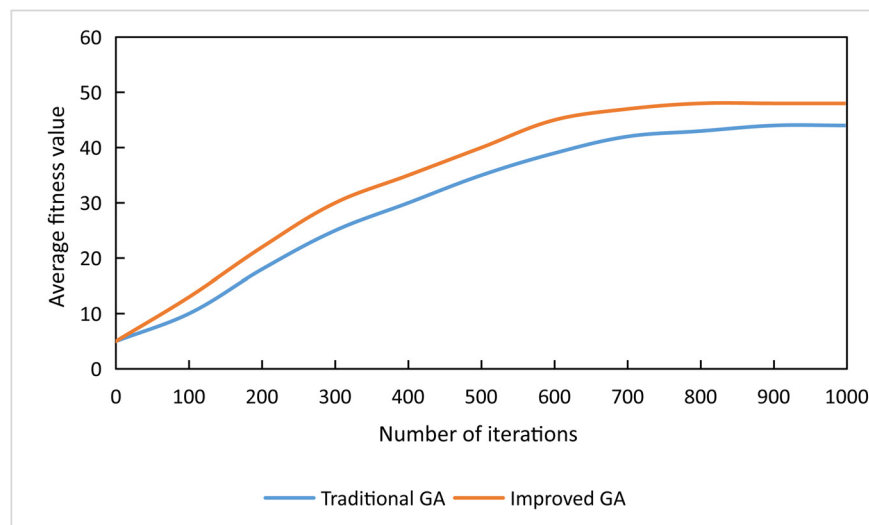


Figure 4: Comparison of traditional and improved genetic algorithms.

Figure 5 shows the average population fitness values and computational time of the two genetic algorithms under 0.4, 0.5, 0.6, 0.7, and 0.8 crossover probabilities when the mutation probability was 0.1. It is seen from Figure 5 that as the crossover probability increased from 0.4 to 0.8, the average fitness value of the two genetic algorithms tended to increase and then decrease, and the computational time of both algorithms tended to decrease and then increase. The comparison of the two genetic algorithms under the same crossover probability showed that the improved genetic algorithm had a larger fitness value and shorter computational time. The reason for the trend of the average fitness value and computational time of

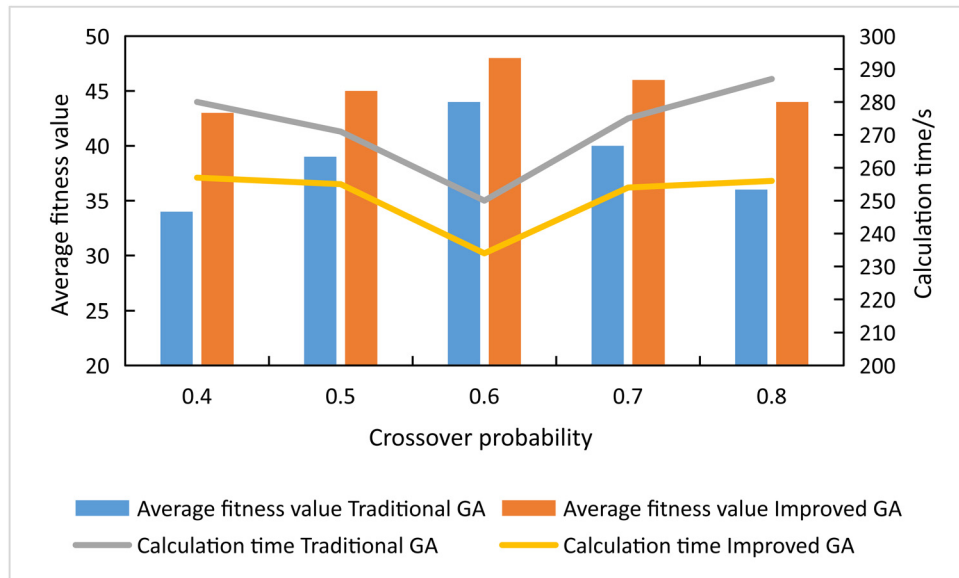


Figure 5: Performance of two genetic algorithms under different crossover probabilities when the mutation probability is 0.1.

the two genetic algorithms with the crossover probability was that when the crossover probability was low, fewer new chromosomes were generated, leading to a slow search, and when it was high, the excellent chromosomes were easily split, leading to a slow search.

Figure 6 shows the average population fitness values and computational time of the two genetic algorithms at 0.1, 0.2, 0.3, 0.4, and 0.5 mutation probabilities when the crossover probability was 0.6. It is seen from Figure 6 that as the variation probability increased from 0.1 to 0.5, the average fitness value of the two genetic algorithms tended to decrease and the detection time tended to increase. The improved genetic algorithm showed a higher fitness value and shorter computational time than the traditional algorithm under the same variance probability. The reason for the trend of the average fitness value and computational time of the two genetic algorithms with the mutation probability was that the variation in chromosomes generated new gene fragments to increase the diversity of population genes, and when the

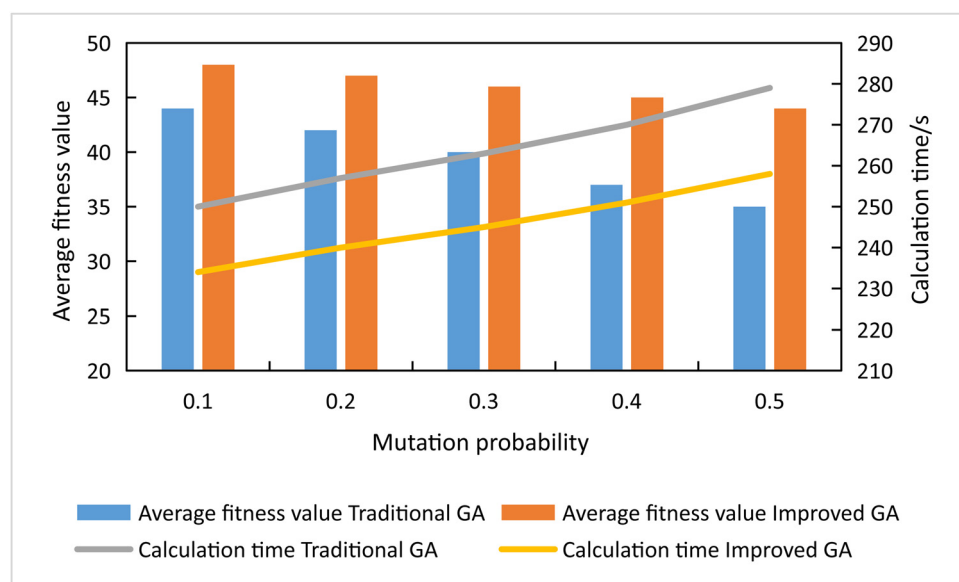


Figure 6: Performance of two genetic algorithms under different variation probabilities when the crossover probability is 0.6.

variation probability increased, the diversity of population genes increased, but it also resulted in the unstable inheritance of good genes.

Both Figures 5 and 6 reflect that the improved genetic algorithm outperformed the traditional genetic algorithm in terms of average fitness value and computational time under the same crossover and mutation probabilities. The reason for the above result was that the improved genetic algorithm used the principle of multi-population coevolution to make the three populations evolve independently and replaced poor chromosomes with excellent chromosomes to enhance the diversity of high-quality chromosomes and jump out of the locally optimal solution.

5 Discussion

For higher education institutions, it is an important task to arrange courses for teachers and students in a reasonable manner. A reasonable scheduling plan can effectively improve the teaching management efficiency of colleges. As the education reform progresses and the importance of higher education is emphasized, the enrollment scale of colleges has increased. However, the teaching resources of universities are limited, and a reasonable scheduling plan is needed to meet the growing demand of students. The increase in the number of students makes the traditional manual scheduling methods increasingly difficult to cope with more complex scheduling problems.

The optimization of the scheduling scheme can be regarded as the search for an optimal scheduling scheme in different candidates, so this study used a genetic algorithm to optimize the scheduling scheme and also used coevolution to improve the genetic algorithm. Finally, simulation experiments were conducted to verify the performance difference between the traditional and improved genetic algorithms under different mutation and crossover probabilities, and the final experimental results are shown above.

Under the same crossover and variation probabilities, the coevolution-improved genetic algorithm converged to stability faster, and the corresponding scheduling scheme was better when convergence was stable. In addition, under the same mutation probability, with the increase in the crossover probability, the fitness values of the scheduling solutions of both traditional and improved genetic algorithms increased first and then decreased, and the computation time decreased first and then increased. Under the same crossover probability, with the increase in the mutation probability, the fitness values of the scheduling schemes of both traditional and improved genetic algorithms tended to decrease, while the computational time tended to increase. The reasons for these results are as follows. Under the same crossover and mutation probabilities, due to the introduction of coevolution, the three populations executed crossover and mutation operations independently and replaced poor chromosomes with excellent ones in turn. The independent evolution of the three populations made it possible to get rid of one of the populations even if it fell into a local optimum by the other population, and the parallel optimization search of the three populations also accelerated the convergence. Therefore, the improved genetic algorithm not only converged faster but also had a larger fitness value after the convergence was stable. Under a fixed mutation probability, a low crossover probability might reduce the efficiency of optimization search due to insufficient chromosome “exchange” within the population, but a high crossover probability might result in too much “exchange” and loss of good chromosomes, which would also reduce the efficiency of optimization search. Therefore, increasing the crossover probability made the fitness value of the two genetic algorithms increase first and then decrease and the computational time decrease first and then increase. If the crossover probability was fixed, the increase in mutation probability would make the chromosomes more likely to produce new genes, but it would also lead to the unstable inheritance of good genes, which would eventually result in slow convergence and lower fitness value in the iterative results.

6 Conclusion

This study briefly introduced the course scheduling problem in universities, proposed the use of a genetic algorithm to optimize the scheduling scheme, improved the genetic algorithm with the principle of coevolution, and finally conducted simulation experiments on the traditional and improved genetic algorithms. The results are as follows. The improved genetic algorithm converged to a stable fitness value (48) after about 750 iterations, and the traditional genetic algorithm converged to a stable fitness value (44) after about 900 iterations. When the mutation probability was fixed at 0.1 and the crossover probability was between 0.4 and 0.8, the average fitness value of the two genetic algorithms tended to increase first and then decrease, and the computation time tended to decrease first and then increase. When the crossover probability was fixed at 0.6 and the crossover probability was between 0.1 and 0.5, the average fitness value of the 2 genetic algorithms tended to decrease, and the detection time tended to increase. The improved genetic algorithm had a higher average fitness value and shorter computation time than the traditional one under the same crossover probability and mutation probability.

Conflict of interest: Author states no conflict of interest.

References

- [1] Altunay H, Eren T. A literature review for course scheduling problem. *Pamukkale Univ J Eng Sci.* 2017;23:55–70.
- [2] Ma YH, Tian GL, Li X. Genetic algorithm for the capacitated chinese postman problem on mixed networks. *Appl Mech Mater.* 2015;701–702:44–9.
- [3] Stephan JJ. Speaker identification using evolutionary algorithm. *Res J Appl Sci, Eng Technol.* 2016;13:717–21.
- [4] Hossain SI, Akhand MAH, Shuvo MIR, Siddique N, Adeli H. Optimization of university course scheduling problem using particle swarm optimization with selective search. *Expert Syst Appl.* 2019;127:9–24.
- [5] Herawatie D, Wuryanto E, Jie F. Course scheduling using modified genetic algorithm in vocational education. *Int J Oper Quant Manag.* 2018;24:203–10.
- [6] Yang L, Xie C. Research on model of course scheduling system for ideological and political teaching in colleges and universities based on particle swarm optimization. *Rev Fac Ing.* 2017;32:657–63.
- [7] Maya NR, Flores JJ, Rangel HR. Performance comparison of evolutionary algorithms for university course timetabling problem. *Comput Sist.* 2016;20:623–34.
- [8] Saptarini N, Suasnawa IW, Ciptayani PI. Senior high school course scheduling using genetic algorithm. *J Phys Conf Ser.* 2018;953:1–7.
- [9] Awad FH, Al-Kubaisi A, Mahmood M. Large-scale timetabling problems with adaptive tabu search. *J Intell Syst.* 2022;31:168–76.
- [10] Gozali AA, Fujimura S. Reinforced island model genetic algorithm to solve university course timetabling. *TELKOMNIKA Indonesian J Electr Eng.* 2018;16:2747.
- [11] Ossai E, Souley B. Resolving timetable scheduling problem based on bio-inspired genetic algorithm. *Brit J Math Comput Sci.* 2017;21:1–19.
- [12] Nahavandi N, Zegordi SH, Abbasian M. Solving the dynamic job shop scheduling problem using bottleneck and intelligent agents based on genetic algorithm. *Int J Eng Trans B Appl.* 2016;29:347–58.
- [13] Zhang S, Wang X. Intelligent course scheduling system based on case-based reasoning. *J Phys Conf Ser.* 2021;1920:1–8.
- [14] Assi M, Halawi B, Haraty RA. Genetic algorithm analysis using the graph coloring method for solving the university timetable problem. *Proc Comput Sci.* 2018;126:899–906.
- [15] Gozali AA, Fujimura S. Solving university course timetabling problem using multi-depth genetic algorithm: Solving UCTP Using MDGA. *SHS Web Conf.* 2020;77:01001.
- [16] Jie A, Yanmei T. Discussion on the strategy and mode of sports optional course based on student satisfaction. *Int J Eng Model.* 2018;31:8–14.