Research Article

A Double Evolutionary Pool Memetic Algorithm for Examination Timetabling Problems

Yu Lei, Maoguo Gong, Licheng Jiao, Wei Li, Yi Zuo, and Qing Cai

*Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education,
International Research Center for Intelligent Perception and Computation, Xidian University, Xi'an, Shaanxi 710071, China*

Correspondence should be addressed to Maoguo Gong; gong@ieee.org

Received 23 January 2014; Accepted 18 August 2014; Published 25 September 2014

Academic Editor: Albert Victoire

Copyright © 2014 Yu Lei et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A double evolutionary pool memetic algorithm is proposed to solve the examination timetabling problem. To improve the performance of the proposed algorithm, two evolutionary pools, that is, the main evolutionary pool and the secondary evolutionary pool, are employed. The genetic operators have been specially designed to fit the examination timetabling problem. A simplified version of the simulated annealing strategy is designed to speed the convergence of the algorithm. A clonal mechanism is introduced to preserve population diversity. Extensive experiments carried out on 12 benchmark examination timetabling instances show that the proposed algorithm is able to produce promising results for the uncapacitated examination timetabling problem.

1. Introduction

Timetabling problem has great significance in our daily life, which has drawn more and more attention since 1960s. Generally speaking, the timetabling problems include educational timetabling [1, 2], nurse timetabling [3], sports timetabling [4], and transportation timetabling [5]. Examination timetabling problem (ETP), which can be described as the allocation of various examinations into predefined timeslots, is one of the common educational timetabling problems. Great challenges for large scale examination timetabling problem have been presented to all researchers. Therefore, it seems quite difficult to find a general and effective solution for examination timetabling due to the complexity and large number of various examinations. There is no known deterministic polynomial algorithm for the ETP, since it is considered to be an NP-hard combinatorial optimization problem.

It has been a long history in the field of examination timetabling problems. Perhaps the earliest excellent survey of examination timetabling problems was introduced by Carter [6] in 1986 and other meaningful papers were then presented by Carter et al. [7, 8]. Since then a significant number of papers in this field have been published and also lots of systems and tools for examination timetabling problems have

been introduced. The represented techniques include graph based techniques, constrain based techniques, population based techniques, multicriteria techniques, hyperheuristics techniques, and decomposition/clustering techniques [9]. To validate the efficiency of algorithms, several benchmark examination timetabling problems have been established and collected, which have been tested in many research papers [8]. The classical benchmark data sets include University of Nottingham benchmark data, University of Melbourne benchmark data, and some test datasets created in the second international timetabling competition.

In 1994, Corne et al. [10] provided a brief survey on using genetic algorithms to solve general educational timetabling problems, and some issues and future prospects were also discussed in their research. Their contribution was to show that the use of representation in genetic algorithms was incapable of dealing with certain problem structures in some specially generated graph coloring problems. In 1996, Ross et al. [11] showed that there existed transition regions in solvable timetabling problems by experimenting upon specially generated graph coloring problems of different connectivity and homogeneity. In the research, authors indicated that their research was helpful to understand how different algorithms perform on complex timetabling problems. Ross et al. [12] in 1998 indicated that there were some weaknesses of using

direct coding in genetic algorithms. In 1999 Terashima-Marin et al. [13] designed a clique-based crossover on the timetabling problems, in which the problems were transferred into graph coloring problems. They not only pointed out the weaknesses of the direct coding in genetic algorithms as that in Ross et al. [12], but also made some contributions to penalty function on both random and real timetabling problems. In 2001, Erben [14] developed a grouping genetic algorithm where appropriate encoding, special crossover and mutation operators and new fitness function were studied. In 2002, Sheibani et al. [15] built a special mathematical model and developed a standard genetic algorithm for solving examination timetabling problems. Wong et al. [16] in 2002 discussed some issues concerning their implementation of a genetic algorithm for solving the examination timetabling problems, which was modeled as a constraint satisfaction problem. In this research, the tournament selection was applied to choose parents and repairing strategies was incorporated with mutation to produce better solutions. In 2005, Côté et al. [17] investigated a biobjective evolutionary algorithm with the objectives of minimizing the timetable length and spacing out the conflicting examinations. Besides, two new local search operators were introduced to deal with the hard and soft constraints. In 2007, Ülker et al. [18] developed a genetic algorithm where Linear Linkage Encoding was used as the representation method. In his paper, there were different crossover operators in conjunction with the linear linkage representation. The experimental results indicated that their methods were feasible.

In the last decade, more researchers paid their attention to introduce some local search operators to improve the quality of solutions obtained in the examination timetabling problems, which had made remarkable achievements. di Gaspero and Schaerf [20] in 2001 carried out a valuable investigation on a family of Tabu Search based techniques whose neighborhoods concerned those which contributed to the violations of hard and soft constraints. In the following year, di Gaspero [34] improved the approach by applying multiple neighbourhoods based on a token-ring search which circularly employs recolor (change a single examination) and shake (swap groups of examinations) in 2002. White [35] developed a four-stage Tabu search called OTTABU, where solutions were gradually improved by taking into more and more constraints in 2000. After then, in 2004, this approach was extended by White et al. [36], where tabu lists could be dynamically relaxed. Paquete and Stützle [21] developed a Tabu search methodology for examination timetabling problems where ordered priorities were given for the constraints. In 2010, Burke et al. [31] investigated variants of variable neighbourhood search and obtained the best results in the literature across some of the problems in the Toronto datasets. Abdullah et al. [27] in 2007 developed a large neighborhood search based on the methodology of improvement graph construction originally developed by Ahuja et al. [37] for different optimization problems. Caramia et al. [19, 38] developed a fine-tuned local search method where a greedy scheduler assigned examinations into the least timeslots and a penalty decrease improved the timetable without increasing the number of timeslots. In 2009, Pillay and Banzhaf [30]

proposed an informed genetic algorithm for the examination timetabling problems, which takes two-phase approach to optimize one individual in two different stages. In 2011, Turabieh and Abdullah [39] proposed a hybrid approach that incorporates effective heuristic operators within the great deluge algorithm for solving examination timetable problems. Alinia Ahandani et al. [40] proposed a method for solving examination timetabling problems mainly based on swarm optimization and two-phase hill-climbing local search operators in 2012, which had obtained good results. Gunawan et al. [41] developed a hybridized Lagrangian relaxation and simulated annealing method for the timetabling problems, in which the initial solutions are obtained by a mathematical programming approach based on Lagrangian relaxation and then improved by a simulated annealing algorithm. In 2014, Xu et al. [33] investigate the robust graph coloring problem with application to a kind of examination timetabling by using the matrix semitensor product and present a number of new results and algorithm.

As is presented above, there are two difficult problems that researchers have to face when they want to solve the ETP. One is how to generate feasible individuals, and the other is how to optimize these feasible individuals. Genetic algorithm is a famous and useful tool for this kind of nonlinear optimization problems. It is well known to all that genetic algorithms obtain better and better solutions through proper coding process, crossover, and mutation operators in a series of iterations. Some parameters and operators should be predefined properly with the aim to get an excellent solution. Therefore, there are recently an increasing number of successful algorithms for the examination timetabling problems based on genetic algorithm with local search methods, which are often called memetic algorithms.

In this paper, we also concentrate on solving the examination timetabling problems with the main framework of the evolutionary algorithm. A double evolutionary pool memetic algorithm (DEPMA) for examination timetabling problems is proposed in our paper. There are two evolutionary pools in our algorithm to solve the two difficult problems, respectively. Each evolutionary pool has its own conventional crossover operator, mutation operator, and selection mechanism. The main evolutionary pool with a large scale of population is aiming to make the constructed examination timetables more and more feasible, in which there is one local search operator based on reordering strategy. The clonal strategy is employed to make the feasible individuals produced in the main evolutionary pool export into the secondary evolutionary pool. The reasons that we use the clonal strategy are as follow: For some test instances (e.g., Car91, Ear83, Hec92, Uta92, and York83), enough feasible solutions cannot be acquired from the main evolutionary pool, thus, the secondary evolutionary pool also cannot guarantee optimal solutions, which affects the ultimate performance of the algorithm (see Figure 4). We employ the clonal strategy to guarantee the population size in the secondary evolutionary pool. In addition, the secondary evolutionary has one local operator, and this operator makes the obtained feasible individuals from the main evolutionary pool satisfy more soft constraints. Finally, several newly produced individuals

will replace the feasible individuals produced in the main evolutionary pool to maintain the diversity and constant scale of population. Several benchmark test datasets are then used to validate the efficiency of our algorithm.

The remaining parts of our paper are organized as follows. We will concentrate on descriptions of the examination timetabling problems in Section 2. Section 3 presents all the details of the proposed algorithms including the framework, crossover and mutation operators, and local search operators. In Section 4, some comparison results on benchmark test problems and analysis on the newly proposed strategy are presented. In the end, some conclusions and future work are discussed in Section 5.

2. Examination Timetabling Problems Formulation

As is described in Section 1, since we should insert all examinations into one timetable at the same time, there is no doubt that many constraints are involved in the examination timetabling problems. Constraints are usually divided into two classes: the hard constraints and the soft ones. They are described as follows.

In any case, hard constraints cannot be violated (mainly due to physical restrictions). That is to say, the examinations with the conflict resources cannot be scheduled simultaneously. A timetable which satisfies all the hard constraints is usually called a feasible timetable.

Soft constraints are desirable but are not absolutely critical. In practice, it seems quite hard to find a solution which satisfies all the soft constraints. Soft constraints vary from one institution to another in terms of both the type and their importance [42]. The most common soft constraint is to spread conflicting examinations as much as possible, whose purpose is to make the students have enough time to rest and prepare for the next examination. Another soft constraint may be that the large examinations should be scheduled as early as possible to allow enough time for marking.

As depicted in Section 1, this paper considers an instance of the ETP that was first formulated by Edmund and Ross [42]. In this problem, a set of exams $E = \{e_1, e_2, \dots, e_E\}$ is to be scheduled into a set of periods $P = \{1, 2, \dots, P\}$. The problem can be formally specified by defining the following: a_{ij} is one if exam e_i is allocated to the period p . e_{ij} is the number of students registered for exams e_i and e_{ij} .

The mathematical formulation is as follows:

$$\min \sum_{i=1}^{|E|-1} \sum_{j=i+1}^{|E|} \sum_{p=1}^{|P|-1} a_{ip} a_{jp} c_{ij}, \quad (1)$$

$$\text{sub. } \sum_{i=1}^{|E|-1} \sum_{j=i+1}^{|E|} \sum_{p=1}^{|P|-1} a_{ip} a_{jp} c_{ij} = 0, \quad (2)$$

$$\sum_{p=1}^{|P|} a_{ip} = 1, \quad \forall i \in \{1, \dots, |E|\}. \quad (3)$$

Equation (1) is the objective of minimizing the number of clashes. Constraint (2) is the constraint that no student is to

be scheduled to take two exams at any one time. Constraint (3) indicates that every exam can only be scheduled once in any timetable.

To evaluate the quality of one feasible timetable, one evaluation function based on soft constraints has been proposed. The function is to evaluate the average cost per student. It can be described as follows:

$$\text{fitness}_1 = \frac{\left(\sum_{s=0}^4 \omega_s \times N_s \right)}{S}. \quad (4)$$

For students taking two examinations which are S timeslots apart from each other, $\omega_s = 2^s$, $s = 0, 1, 2, 3, 4$, is the penalty of scheduling these examinations 4, 3, 2, 1, or 0 timeslots away, respectively. N_s is the number of students who take the examinations. S is the total number of students in the timetable.

3. Proposed Double Evolutionary Pools Memetic Algorithm for the ETP

In this section, a double evolutionary pool memetic algorithm is introduced to solve the incapacitated examination timetabling problems. In addition, several local search operators are designed for each evolutionary pool, whose aims are to make the individual become more and more feasible and satisfy as many soft constraints as possible.

Generally speaking, the standard evolutionary algorithms are a class of general-purpose optimization tools that model the principles of nature evolution. Evolutionary algorithms are population-based heuristic methods, which start from an initially randomly generated population. Then all individuals in the population are evaluated according to one problem-specific function, which is usually called fitness function. After the evaluation process, some good individuals are chosen to undergo the recombination process. In the recombination process, crossover and mutation operators are applied to create new individuals in order to explore the search space. After that, the newly produced individuals replace the old ones, which are usually the worst ones of the population based on the fitness function. This process is repeated until a stopping criterion is reached, which may be the maximum number of generations or a time limit. Maybe the first use of genetic algorithm in timetabling problems appeared in 1990 [8]. Since then, there have been an increasing number of genetic algorithms for the examination timetabling problem.

In our paper, we propose a double evolutionary pool memetic algorithm for examination timetabling problem, which is denoted as DEPMA. Algorithm 1 shows the procedure of our proposed algorithm. The main framework of DEPMA can be described as follows. There are two evolutionary pools (search spaces) in the framework. In examination timetabling problem, there are common crossover and mutation operators in each pool. The population scale in main evolutionary pool is always set from 50 to 100, and then the scale in the secondary evolutionary pool is always set from 20 to 30. The population in the main evolutionary pool will become more and more feasible through crossover, mutation,

Input: PopM: The population in the main evolutionary pool
 NM: The scale of population in the main evolutionary pool
 PopS: The population in the secondary evolutionary pool
 NS: The scale of population in the secondary evolutionary pool
 P_C : The crossover probability in the main evolutionary pool
 P_M : The mutation probability in the main evolutionary pool
 P_{Mmain} : The mutation probability for one single individual in the main evolutionary pool
 P_{Cmain} : The crossover probability for one single individual in the main evolutionary pool
 T_{exc} : The number of exchanged timeslots in the secondary evolutionary pool
 P_{Msec} : The mutation probability for one single individual in the secondary evolutionary Pool
 IterMax: The maximum iteration number
 Itermax: The maximum iteration number for the local search operators

Output: The best individual PopBest in the secondary evolutionary pool

Step 1. Initialization. Randomly create NM individuals in the main evolutionary pool according to what is described in Part B, which is denoted as PopM. Meanwhile, $PopM = \emptyset$ and $Iter = 0$.

Step 2. Crossover. Randomly select $\lceil P_C \times NM \rceil$ individuals in the main evolutionary pool and perform crossover operation described in Part 3.3.

Step 3. Mutate. Randomly select $\lceil P_M \times NM \rceil$ individuals in the main evolutionary pool and perform mutation operation PopM described in Part 3.4.

Step 4. Local optimization. For each individual through crossover and mutation operation, the special local search operator is performed according to what is described in Part 3.5.

Step 5. Selection. Select individuals from the main evolutionary pool to the secondary evolutionary pool. Once the individuals in the main evolutionary pool satisfy all hard constraints, all feasible individuals are put into the secondary evolutionary pool. Meanwhile, create some new individuals replacing those feasible ones in the population for the main evolutionary pool.

Step 6. Clone. Assume that the number of feasible individuals is NF. If $NF \geq NS$, randomly select NS individuals from all feasible individuals. Otherwise, each feasible individual obtained from the main evolutionary pool is copied according to the following rules. The number of each feasible individual after clone is $\lceil NS/NF \rceil$. And then combine original individuals in the secondary evolutionary pool with the cloned individuals. Select the best NS individuals as the population in the secondary evolutionary pool.

Step 7. Local optimization. After clone, the local search operator described in Part 3.6 is performed to make the feasible individuals satisfy more and more soft constraints.

Step 8. Selection. Select potential individuals in the secondary evolutionary pool according to (4). Keep the best individual in the current generation constant in the population in the secondary evolutionary pool. And then combine other individuals in the original secondary evolutionary pool with the best individual to form the population in the secondary evolutionary pool.

Step 9. $Iter = Iter + 1$, and go to Step 2.

Step 10. If Iter reaches IterMax output the best individual PopBest in the secondary evolutionary pool.

ALGORITHM 1: The details of our proposed algorithm.

and one local search operator. That is, the population in the main evolutionary pool will satisfy more and more the hard constraint. The feasible individuals produced in the main pool are put into the secondary evolutionary pool to perform a predefined number of randomly clonal processes, whose aim is to optimize the iterations and make full use of one feasible individual. After that, the individuals in the secondary evolutionary pool will satisfy more and more soft constraints through the common crossover, mutation and one local search operator. To keep the population scale constant, the individuals in the main evolutionary pool who have become feasible will be replaced by several newly created individuals. In addition, we introduce one reordering strategy in the main evolutionary pool. According to clashes in each timeslot in ascending order, the timeslots should be reordered when a new examination is to be inserted the timetable. Our motivation is to give the priority to the timeslots with fewer clashes when inserting a new examination.

Before introducing the details of our algorithm, we should concentrate on the motivation of proposed framework and two new strategies in our algorithm. Conventionally, fitness function obtained by weighted-sum of the hard constraint and the soft constraint cannot reflect which is the primary factor between the hard constraint and soft constraint in the gradual optimization process. Therefore, our double evolutionary pool framework is proposed, whose purpose is to make the individuals become feasible in the main evolutionary pool and to make the feasible individuals satisfy more soft constraints in the secondary evolutionary pool. It should be emphasized that these two evolutionary pools are not independent. Once feasible individuals emerge in the main evolutionary pool, they will be cloned and put into the secondary evolutionary pool. There is one disadvantage without clonal strategy. In the evolutionary process, perhaps the newly produced individuals can be easily obtained from parent just through exchanging different timeslots. Therefore,

TABLE 1: The structure of individuals in our algorithm.

T_1	T_2	T_3	T_4	T_a
1	0	0	0	0
0	1	0	0	0
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0

to speed up the convergence speed and make full use of one feasible individual, clonal strategy is introduced in the selection process between these two evolutionary pools. In addition, our reordering strategy in the main evolutionary pool is to reorder the timeslots before one new examination is inserted into one timetable, which is beneficial to give the priority to those timeslots with fewer examinations and to make the examinations with more conflicts separate from each other. In this case, the numbers of examinations in each timeslot will become more uniform. Meanwhile, the individuals can become feasible much easier. In the following parts, the whole proceeding from the start to the end of our algorithm is introduced.

3.1. A Binary Coding Representation. In our paper, the incapacitated examination timetabling problem is the focal point of our research. A binary coding representation for examination timetabling is introduced, whose details are shown as follows:

$$T_{ij} = \begin{cases} 1 & \text{if the exam } i \text{ is scheduled in timeslot } j \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

To schedule all the examinations in one timetable, the number of timeslots is always set one more than the given ones. That is to say, there are two kinds of timeslots in our timetable. One is called the regular timeslots and the other is called alternate timeslot.

Table 1 shows the structure of the solution in our proposed algorithm. Five examinations are allocated into four different timeslots. The final column t_a denotes the alternate timeslot. A number of 1 in the timeslot column represents that the examination is allocated in this timeslot. A number of 1 in the final timeslot indicates that the examination cannot be allocated in any timeslot due to hard constraints and the timetable is an unfeasible solution. Therefore, the fitness function in the main evolutionary pool can be shown in (6), where t_a denotes the alternate timeslot. Our motivation is to minimize the clashes in the alternate timeslot. The individual is not feasible until the fitness value reaches

$$\text{fitness}_2 = \sum_{i=1}^n t_a. \quad (6)$$

3.2. Initialization. In our algorithm, examinations are randomly inserted into a binary timetable one by one. In the initialization part, a new reordering strategy is introduced in order to make one feasible timetable as soon

FIGURE 1: The illustration of how to allocate examinations in the initialization.

as possible (see Figure 1). That is, before inserting one new examination into the timetable, we should sort all timeslots in descending order according to some indicators, such as the conflicts in each timeslot or total number of students for each timeslot. After the reordering strategy, some examinations with large conflicts or anticipating students will be moved to the last few timeslots and the first several timeslots with few conflicts can have much space for inserting another new examination. Meanwhile, the new reordering strategy makes each timeslot have nearly the same number of examinations, which is beneficial for teachers and students not to take too many examinations practically.

Consider an example as follows. Assume that there are four timeslots in the timetable with several examinations. 10 examinations have been allocated in this timetable. We will detail that how the 11th examination is allocated into the timetable. According to what is described above, before we insert the 11th examination into the timetable, we should reorder all timeslots except the alternate timeslot according to some indicators. Here consider the number of examinations and the hard constraint in each timeslot as the indicator. Calculate the sum of examinations in each timeslot, which are 3, 2, 4, and 1, respectively. Therefore, the position of each timeslot will be exchanged according to the indicator. When inserting a new examination, our purpose is to give priority to the timeslot which has fewer examinations and the new examination in this timeslot does not violate the hard constraint. After the reordering strategy, the timetable is transformed. Through the reordering strategy, we can insert a new examination without causing any conflict much easily.

3.3. Crossover Operator in Main Evolutionary Pool. Since the fitness function in the main evolutionary pool is to minimize the number of examinations in the alternate timeslot, there are some disadvantages in randomly crossover for this kind of timetabling scheduling problem. The reason is that randomly selected examinations or timeslots to exchange may lead to the increasing of sum of examinations in the alternate timeslot. Consequences are that there are rarely few feasible individuals in the main evolutionary pool and lots of computations are wasted. Meanwhile, through observing many experimental results, there are few examinations in the alternate timeslot after the randomly initialization. Therefore, there exist some timeslots in both parents, which do not contain examinations in alternate timeslots for both parents. Then randomly select some above timeslots to crossover will not make the children worse than their parents as much as

Input: Randomly selected parents P_1, P_2 and the crossover probability P_{Cmain}

Output: Newly produced children PN_1, PN_2

Step 1. Assume that the examinations sets in the alternate timeslot in P_1, P_2 are ES_1, ES_2 and n is the number of total timeslots in our timetable. Randomly select examinations in $[P_{Cmain} \times (n - 1)]$ timeslots, which do not contain any examination in ES_1, ES_2

Step 2. Exchange the selected timeslots in P_1, P_2 , and then delete the duplicate examinations in unselected timeslots in P_1, P_2 , which are denoted as PN_1, PN_2 .

Step 3. Output the newly produced children PN_1, PN_2

ALGORITHM 2: The details of the crossover operator in the main evolutionary pool.

```

P: the original individual
PN: the new individual
Iter: the iteration number
Itermax: the maximum iteration number:
SD: the search depth.
Enum: the number of examinations
r: the increasing step

Iter = 0, r = 1
WHILE Iter < Itermax OR r < 5
    randomly select  $\lceil (E_{num}/SD) \times r \rceil$  examinations and rearrange the in descending order, then obtain  $P_1$ 
    Iter = Iter + 1
    IF fitness1( $P_1$ ) < fitness1(P)
        PN =  $P_1$       BREAK
    END
    IF Iter = Itermax
        r = r + 1
        Iter := 0
    END
END

```

ALGORITHM 3: The pseudocode of the local search approach in the main evolutionary pool.

possible. Therefore, one special crossover operator is applied in our algorithm. Assume that selected parents are P_1, P_2 . The details are described in Algorithm 2.

3.4. Mutation Operator in the Main Evolutionary Pool. In the mutation operator, the reordering strategy is still applied to extend the search space. After randomly selecting some examinations to make the mutation process, a special reordering strategy is applied here. According to some indicators such as the number of participating students or the conflicts for each examination, the examinations are selected to perform mutation operation. Then all selected examinations will be inserted into the timetable one by one again. Examinations that cannot be put into nonalternate timeslots should be ranged into the alternate timeslot. Consider an example that there are 10 examinations in the timetable. The mutation ratio P_{Mmain} equals 0.3. So the total number of examinations selected to make the mutation process is 3. Assume that the selected examinations sets are {1, 4, 5} and the total number of participating students for them are 20,

101, and 50 separately. Therefore, the sorted examinations sets for mutation are {4, 5, 1}. That is to say, the examinations with more students or conflicts should be given the priority to be rearranged firstly.

3.5. Local Search Operator in the Main Evolutionary Pool. To make the individual satisfy more and more hard constraints, a local search operator will be introduced. Conventionally, the simulated annealing algorithm is always used as the local search operator. The simulated annealing algorithm makes the individuals change in the annealing process, whose purpose is to enable the individuals to become better than the initial state as much as possible and accept bad individuals with a small probability. However, there are few feasible individuals in the main evolutionary pool in our algorithm. We should enhance the strength of the local search operator. Therefore, in our local search operator the small probability to accept bad individual is removed. We only receive better individuals after annealing process. Otherwise, the individual should keep constant. Additionally, during the annealing

FIGURE 2: The illustration of the crossover operator in the secondary evolutionary pool.

FIGURE 3: The illustration of the mutation operator in the secondary evolutionary pool.

process, if the individuals cannot become better through a certain number of iteration, the sum of examinations to be rearranged should be increased by a constant step. For instance, the step in our algorithm is set as 1, which means that the sum of examinations to be rearranged will be increased by 10 percent. (see Algorithm 3).

3.6. Local Search Operator in the Secondary Evolutionary Pool.

In the secondary evolutionary pool, one special crossover operator and mutate operator will be integrated into the local search operator in the secondary evolutionary pool. Since all individuals are feasible in the secondary evolutionary pool and all timeslots are independent, random exchange of some timeslots cannot change the feasible properties for individuals in the secondary evolutionary pool. Therefore, the crossover operator will randomly exchange T_{exc} timeslots in the timetable for individuals in the secondary evolutionary pool. Consider an example as follows. Assume that the timeslots sets selected to make the crossover process is $\{1, 4\}$, we exchange all examinations in these two timeslots and then the newly produced individual can be obtained in Figure 2.

The mutation operator is more complicated. Randomly select a small number of examinations and set their original position as 0. Then insert them into the timetable from the first timeslot to the last one without causing any conflict; otherwise, the individual should be constant. Therefore, details of mutation operator are described as Figure 3.

Our local search operator in the second evolutionary pool is based on the crossover operator and the mutation operator above. After a certain number of iteration for crossover and mutation process, the selected individuals will be replaced by the newly produced better individuals. Otherwise, if the individuals cannot become better, they should be kept constant. Therefore, the details are described in Algorithm 4.

After describing all in details, we should concentrate on how to validate the efficiency of our algorithm. Therefore, in next section 12 benchmark examination timetabling problems are introduced and tested on our DEPMA.

4. Experimental Study

In this section, some experimental results are presented to validate our algorithm. Firstly, 12 benchmark test problems

TABLE 2: The characteristics of benchmark exam timetabling problems.

Date set	Timeslots	Exams	Students	Conflict density
Car 91	35	682	16925	0.13
Car 92	32	543	18419	0.14
Ear 83	24	190	1125	0.27
Hec 92	18	81	2823	0.42
Kfu 93	20	461	5349	0.6
Lse 91	18	381	2726	0.6
Rye 92	23	486	11483	0.08
Sta 83	13	139	611	0.14
Tre 92	23	261	4360	0.18
Ute 92	10	184	2750	0.08
Uta 93	35	622	21266	0.13
York 83	21	181	941	0.29

are briefly introduced. Secondly, we will show the efficiency of our proposed strategy in boxplots. Finally, our best results will be compared with the other ones obtained in another 16 algorithms. Note that the DEPMA is implemented on a 2.8 GHz core personal computer.

The high level of research interest in examination timetabling has led to the establishment of a variety of different benchmark problems which have been widely studied. Carter et al. [8] in 1996 introduced a set of several real world examination timetabling problems from three Canadian high schools, five Canadian universities, one American university, one British university, and one university in Saudi Arabia. Over the past few years, they were widely employed as test beds in examination timetabling research.

In all the test problems, to indicate the density of the conflicting examinations in each of the instances, a conflict matrix C was defined, where each element $c_{ij} = 1$ if exam i conflicts with exam j (i.e., they have common students), or $c_{ij} = 0$ otherwise. The conflict density represents the ratio between the numbers of elements of value “1” to the total number of elements in the conflict matrix. The enrolments are the total number of examinations that each student takes part in. All the details of these test problems are shown in Table 2.

```

P: the original individual
PN: the new individual
Iter: the iteration number
Itermax: the maximum iteration number:
 $E_{\text{num}}$ : the number of examinations
 $T_{\text{exc}}$ : the exchange number of timeslots

Iter = 0;
WHILE Iter < Itermax
    Selecte  $\lceil P_{\text{Msec}} \times E_{\text{num}} \times (\text{Itermax}/\text{Iter} + 1) \rceil$  examinations randomly from individual  $P$ .
    Perform the mutation and crossover process, and then obtain the new individual  $P_1$ ;
    Iter = Iter + 1;
    IF  $\text{fitness}_1(P_1) < \text{fitness}_1(P)$ 
        PN =  $P_1$ ;
    ELSE
        PN =  $P$ ;
    END
END

```

ALGORITHM 4: The pseudocode of the local search approach in the secondary evolutionary pool.

TABLE 3: The parameter setting of our algorithm.

Parameter	Values
NM/NS: the scale of the population in the main/secondary evolutionary pool	100/30
$P_C/P_{C\text{main}}$: the same crossover probability in these two evolutionary pools	0.8
$P_M/P_{M\text{main}}$: the same mutation probability in these two evolutionary pools	0.2
IterMax: the maximum iteration number of the whole algorithm	500
Itermax: the maximum iteration number of the local search approach	10
T_{exc} : the exchange number of timeslots	2

Since we introduce the reordering strategy to make the examinations with large conflicts separate from each other and the clonal strategy to make full use of one feasible individual and to accelerate the convergence speed, we have made some experiments to compare the performance between DEPMA with and without reordering strategy and DEPMA with and without clonal strategy. All parameters are set in Table 3. There are 30 independent runs for each algorithm. The fitness values are shown in boxplots in Figure 4.

It can be found that, for car91 test instance, car92 test instance, hec92 test instance, sta83 test instance, tre92 test instance, uta92 test instance, ute92 test instance, and yor83 test instance, our algorithm with both clonal strategy and reordering strategy obtains the best results. Our algorithm with either clonal strategy or reordering strategy has nearly the same performance for these 8 problems. However, for ear83 test instance, the algorithm without reordering strategy gets the best performance. For kfu83 test instance, the DEPMA without clonal strategy gets the best experimental results. And then the DEPMA without reordering strategy gets the best performance for lse91 test instance, which is in the same case for rye 92 test problems. Overall considering what is said above, the newly introduced clonal strategy in selection process and reordering strategy are benefit for our algorithm.

In our following experiments, we will test the overall performance of the algorithm. There are also 30 independent runs for each test problem. To validate the efficiency of our algorithm, three indicators are introduced here. The generation when the first feasible individual comes out is used to represent the search efficiency in the main evolutionary pool. Meanwhile, the best fitness value and run time is introduced to show the efficiency of the secondary evolutionary pool. All experimental results are represented in boxplots in Figure 5. The parameters are set in Table 3.

Some conclusions can be obtained from Figure 5. Firstly, our algorithm is robust in performance. There are few outliers in all boxplots, which means that our local search operators in both main evolutionary pool and secondary evolutionary pool have steady search efficiency. Secondly, for the key step that we sort timeslots after inserting each examination in the initialization process, the examinations with large conflicts are not ranged into the same timeslot, which is benefit for us to obtain robust performance. There are nearly no outliers in boxplots for best fitness values of all twelve test problems. The difference between the best fitness value and the worst ones is no more than 3. Even in some test instance, the difference reaches no more than 0.3. Finally, the run times for our algorithm are ranged from several hundred to tens of thousands, which means that the algorithm is so low in

FIGURE 4: Boxplots for best fitness values obtained from DEPMA (a), DEPMA without clonal strategy (b), and DEPMA without reordering strategy (c).

speed that needs too much optimization. Overall considering what are presented above, we can conclude that our algorithm is efficient and robust in solving examination timetabling problems.

In Table 4, the best fitness values obtained from other 17 famous algorithms are compared with those obtained in our algorithm. Meanwhile, the best results obtained in our algorithm are shown in Table 4. All the best results

represented in the following table are obtained from the best ones of 30 independent runs.

It can be seen that the results obtained in our algorithm are neither the best nor the worst. The best ranking of examination instance is that for sta83 test problem, which is 2nd in all algorithms. It seems quite different with different test instances. That is to say, our algorithm is worthwhile and general for solving examination tabling problem.

FIGURE 5: Boxplots for the best fitness value (a), the run time (b), and the generation when the first feasible individual comes out in the main evolutionary pool (c).

5. Conclusions and Future Work

This paper briefly overviews methods for solving incapacitated examination timetabling problems, which are based on graph coloring strategy or based on genetic algorithm. And then a double evolutionary pool memetic algorithm for solving incapacitated timetabling problems is introduced. The mechanism of this method is that, in the main evolutionary pool, we use common crossover operator, mutation operator, and one local search operator to make the individuals satisfy more and more hard constraints. Once one feasible individual comes out, the clonal strategy is applied in the selection process from the main evolutionary pool to the secondary evolutionary pool. In the secondary evolutionary pool, one

newly local search operators is introduced based mainly on exchanging strategy to make the individuals more and more outstanding. Finally, some experiments for proving the efficiency of our clonal strategy and reordering strategy in local search operators are presented. Meanwhile, the best fitness values obtained in our algorithm are compared to those obtained by the other 16 algorithms. The results show that the examination tabling problems.

However, there are still some disadvantages in our algorithm. Our future work will focus on introducing more reasonable local search operators to improve the performance of our algorithm. And then how to analyse the correlation between examinations is still a huge challenge for us. What

TABLE 4: The fitness values of our algorithm compared against the best fitness values of current approaches.

Data sets	Our result	Carter et al. [8]	Caramia et al. [19]	Di Gaspero and Schaefer [20]	Paquete and Stützle [21]	Burke and Newall [22]	Burke et al. [1]
Car 91	6.4	7.1	6.6	6.2	—	4.65	5.0
Car 92	5.2	6.2	6.0	5.2	—	4.1	4.3
Ear 83	39.8	36.4	29.3	45.2	38.2	37.05	36.2
Hec 92	11.8	10.8	9.2	12.2	11.2	11.54	11.6
Kfu 93	16.2	14.0	13.2	18.2	16.2	13.9	15.0
Lse 91	14.5	10.5	9.6	15.2	13.2	10.82	11.0
Rye 92	12.3	7.3	6.8	—	—	—	—
Sta 83	157.2	161.5	158.2	160.2	168.2	168.73	161.9
Tre 92	9.5	9.6	9.2	10.2	9.2	8.35	8.4
Uta 92	4.3	3.5	3.2	4.2	—	3.2	3.4
Ute 92	28.6	25.8	24.4	27.8	29.2	25.83	27.4
York 83	40.5	41.7	36.2	41.2	38.2	37.28	40.8
Data sets	Burke et al. [23]	Asmuni et al. [24]	Côté et al. [17]	Kendall and Hussin [25]	Yang and Petrovic [26]	Abdullah et al. [27]	Burke et al. [28]
Car 91	4.8	5.19	5.2	5.37	4.5	5.2	5.36
Car 92	4.2	4.51	4.2	4.67	3.93	4.4	4.53
Ear 83	35.4	36.64	34.2	40.18	33.7	34.9	37.92
Hec 92	10.8	11.6	10.2	11.86	10.83	10.3	12.25
Kfu 93	13.7	15.34	14.2	15.84	13.82	13.5	15.2
Lse 91	10.4	11.35	11.2	—	10.35	10.2	11.33
Rye 92	8.9	10.05	8.8	—	8.53	8.7	—
Sta 83	159.1	160.79	157.2	157.38	158.35	159.2	158.19
Tre 92	8.3	8.47	8.2	8.39	7.92	8.4	8.92
Uta 92	3.4	3.52	3.2	—	3.14	3.6	3.88
Ute 92	25.7	27.55	25.2	27.6	25.39	26.0	28.01
York 83	36.7	39.79	36.2	—	36.35	36.2	41.37
Data Sets	Eley [29]	Qu et al. [9]	Pillay and Banzhaf [30]	Burke et al. [31]	Burke et al. [32]	Xu et al. [33]	
Car 91	5.2	5.16	4.9	4.6	5.19	6.5	
Car 92	4.3	4.16	4.2	4.0	4.31	5.2	
Ear 83	36.8	35.86	35.9	32.8	35.79	39.6	
Hec 92	11.1	11.94	11.5	10.0	11.19	11.63	
Kfu 93	14.5	14.79	14.4	13.0	14.51	16.3	
Lse 91	11.3	11.15	10.9	10.0	10.92	13.4	
Rye 92	9.8	—	10.89	—	—	14.7	
Sta 83	157.3	159.0	157.8	159.9	157.18	160.2	
Tre 92	8.6	8.6	8.4	7.9	8.49	9.5	
Uta 92	3.5	3.59	3.4	3.2	3.44	3.2	
Ute 92	26.4	28.3	27.2	24.8	26.7	27.9	
York 83	39.4	41.81	39.3	37.28	39.47	—	

is more important is that we will make some efforts to apply our algorithms in solving the other timetabling problems, which may be more difficult than incapacitated examination timetabling problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] E. Burke K, G. Kendall, M. Mısıır et al., “Applications to timetabling,” in *Handbook of Graph Theory*, vol. 445, 1st edition, 2004.
- [2] A. Colorni, M. Dorigo, and V. Maniezzo, “Metaheuristics for high school timetabling,” *Computational optimization and applications Journal*, vol. 9, no. 3, pp. 275–298, 1998.
- [3] E. K. Burke, P. de Causmaecker, G. V. Berghe, and H. van Landeghem, “The state of the art of nurse rostering,” *Journal of Scheduling*, vol. 7, no. 6, pp. 441–499, 2004.

- [4] K. Easton, G. Nemhauser, and M. Trick, "Sports scheduling," in *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, J. Leung, Ed., vol. chapter 52, CRC Press, Boca Raton, Fla, USA, 2004.
- [5] K. C. Tan, Y. H. Chew, and L. H. Lee, "A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows," *Computational Optimization and Applications*, vol. 34, no. 1, pp. 115–151, 2006.
- [6] M. W. Carter, "A survey of practical applications of examination timetabling algorithms," *Operations Research*, vol. 34, no. 2, pp. 193–202, 1986.
- [7] M. W. Carter, G. Laporte, and J. W. Chinneck, "A general examination scheduling system," *Interfaces*, vol. 24, no. 3, pp. 109–120, 1994.
- [8] M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: algorithmic strategies and applications," *Journal of Operational Research Society*, vol. 47, no. 3, pp. 373–383, 1996.
- [9] R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, "A survey of search methodologies and automated system development for examination timetabling," *Journal of Scheduling*, vol. 12, no. 1, pp. 55–89, 2009.
- [10] D. Corne, P. Ross, and H. L. Fang, *Fast Practical Evolutionary Timetabling*, Springer, Berlin, Germany, 1994.
- [11] P. Ross, D. Corne, and H. Terashima-Marin, "The phase transition niche for evolutionary algorithms in timetabling," in *Practice and Theory of Automated Timetabling I: Selected Papers from the 1st International Conference*, E. K. Burke, Ed., vol. 1153 of *Lecture Notes in Computer Science*, pp. 309–324, Springer, Berlin, Germany, 1996.
- [12] P. Ross, E. Hart, and D. Corne, "Some observations about GA based exam timetabling," in *Proceedings of the 2nd International Conference on Practice and Theory of Automated Timetabling*, E. K. Burke and M. W. Carter, Eds., vol. 1408 of *Lecture Notes in Computer Science*, pp. 115–129, Springer, Berlin, Germany, 1998.
- [13] H. Terashima-Marin, P. Ross, and M. Valenzuela-Rendon, "Clique-based crossover for solving the timetabling problem with GAs," in *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 1200–1206, Washington, DC, USA, 1999.
- [14] W. Erben, "A grouping genetic algorithm for graph colouring and exam timetabling," in *Practice and Theory of Automated Timetabling III*, E. K. Burke and W. Erben, Eds., vol. 2079 of *Lecture Notes in Computer Science*, pp. 132–156, Springer, Berlin, Germany, 2001.
- [15] K. Sheibani, E. K. Burke, and P. de Causmaecker, "An evolutionary approach for the examination timetabling problems," in *Proceeding of the 4th International Conference on Practice and Theory of Automated Timetabling*, E. K. Burke and P. de Causmaecker, Eds., pp. 387–396, KaHo Sint-Lieven, Gent, Belgium, 2002.
- [16] T. Wong, P. Côté, and P. Gely, "Final exam timetabling: a practical approach," in *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (CCECE '02)*, pp. 726–731, 2002.
- [17] P. Côté, T. Wong, and R. Sabouri, "Application of a hybrid multiobjective evolutionary algorithm to the uncapacitated exam proximity problem," in *Practice and Theory of Automated Timetabling V: Selected Papers from the 5th International Conference*, E. K. Burke and M. Trick, Eds., vol. 3616 of *Lecture Notes in Computer Science*, pp. 151–168, Springer, Berlin, Germany, 2005.
- [18] O. Ülker, E. Özcan, and E. E. Korkmaz, "Linear linkage encoding in grouping problems: applications on graph colouring and timetabling," in *Practice and Theory of Automated Timetabling VI*, E. K. Burke and H. Rudova, Eds., vol. 3867 of *Lecture Notes in Computer Science*, pp. 347–363, Springer, Berlin, Germany, 2007.
- [19] M. Caramia, P. Dell'Olmo, and G. F. Italiano, "New algorithms for examination timetabling," in *Proceedings of the 4th International Workshop on Algorithm Engineering*, S. Naher and D. Wagner, Eds., vol. 1982 of *Lecture Notes in Computer Science*, pp. 230–241, Springer, Berlin, Germany, 2001.
- [20] L. Di Gaspero and A. Schaerf, "Tabu search techniques for examination timetabling," in *Practice and Theory of Automated Timetabling III*, E. K. Burke and W. Erben, Eds., vol. 2079 of *Lecture Notes in Computer Science*, pp. 104–117, Springer, Berlin, Germany, 2001.
- [21] L. Paquete and T. Stützle, "An experimental investigation of iterated local search for colouring graphs," in *Applications of Evolutionary Computing*, S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, and G. Raidl, Eds., vol. 2279 of *Lecture Notes in Computer Science*, pp. 122–131, Springer, Berlin, Germany, 2002.
- [22] E. K. Burke and J. P. Newall, "Enhancing timetable solutions with local search methods," in *Practice and Theory of Automated Timetabling IV*, pp. 195–206, Springer, Berlin, Germany, 2003.
- [23] E. K. Burke, Y. Bykov, J. Newall, and S. Petrovic, "A time-predefined local search approach to exam timetabling problems," *IIE Transactions*, vol. 36, no. 6, pp. 509–528, 2004.
- [24] H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, "Fuzzy multiple heuristic orderings for examination timetabling," in *Practice and Theory of Automated Timetabling*, pp. 334–353, Springer, Berlin, Germany, 2005.
- [25] G. Kendall and N. M. Hussin, "An investigation of a tabu-search-based hyper-heuristic for examination timetabling," in *Multidisciplinary Scheduling: Theory and Applications*, pp. 309–328, Springer, New York, NY, USA, 2005.
- [26] Y. Yang and S. Petrovic, "A novel similarity measure for heuristic selection in examination timetabling," in *Practice and Theory of Automated Timetabling*, pp. 247–269, V. Springer, Berlin, Germany, 2005.
- [27] S. Abdullah, S. Ahmadi, E. K. Burke, and M. Dror, "Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling," *OR Spectrum*, vol. 29, no. 2, pp. 351–372, 2007.
- [28] E. K. Burke, B. McCollum, A. Meisels, and R. Qu, "A graph-based hyper-heuristic for educational timetabling problems," *European Journal of Operational Research*, vol. 176, no. 1, pp. 177–192, 2007.
- [29] M. Eley, "Ant algorithms for the exam timetabling problem," in *Practice and Theory of Automated Timetabling VI*, E. K. Burke and H. Rudova, Eds., vol. 3867 of *Lecture Notes in Computer Science*, pp. 364–382, Heidelberg, The Netherlands, 2007.
- [30] N. Pillay and W. Banzhaf, "An informed genetic algorithm for the examination timetabling problem," *Applied Soft Computing*, vol. 10, no. 2, pp. 457–467, 2010.
- [31] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Hybrid variable neighbourhood approaches to university exam timetabling," *European Journal of Operational Research*, vol. 206, no. 1, pp. 46–53, 2010.
- [32] E. K. Burke, R. Qu, and A. Soghier, "Adaptive selection of heuristics for improving exam timetables," *Annals of Operations Research*, vol. 218, no. 1, pp. 129–145, 2014.
- [33] M. Xu, Y. Wang, and A. Wei, "Robust graph coloring based on the matrix semi-tensor product with application to examination

- timetabling,” *Control Theory and Technology*, vol. 12, no. 2, pp. 187–197, 2014.
- [34] L. di Gaspero, “Recolour, shake and kick: a recipe for the examination timetabling problem,” in *Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling*, E. K. Burke and P. de Causmaecker, Eds., pp. 404–407, KAHO Sint-Lieven, Gent, Belgium, 2002.
 - [35] G. M. White, “Constrained satisfaction, not so constrained satisfaction and the timetabling problem,” in *Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling*, E. K. Burke and W. Erben, Eds., pp. 32–47, Konstanz, Germany, August 2000.
 - [36] G. M. White, B. S. Xie, and S. Zonjic, “Using tabu search with longer-term memory and relaxation to create examination timetables,” *European Journal of Operational Research*, vol. 153, no. 1, pp. 80–91, 2004.
 - [37] R. K. Ahuja, J. B. Orlin, and D. Sharma, “Multi-exchange neighbourhoodsearch algorithm for capacitated minimum spanning tree problem,” *Mathematical Programming B*, vol. 91, no. 1, pp. 71–97, 2001.
 - [38] M. Caramia, P. Dell’Olmo, and G. F. Italiano, “Novel local search-based approaches to university examination timetabling,” *INFORMS Journal of Computing*, vol. 20, no. 1, pp. 86–99, 2008.
 - [39] H. Turabieh and S. Abdullah, “An integrated hybrid approach to the examination timetabling problem,” *Omega*, vol. 39, no. 6, pp. 598–607, 2011.
 - [40] M. Alinia Ahandani, M. T. Vakil Baghmisheh, M. A. Badamchi Zadeh, and S. Ghaemi, “Hybrid particle swarm optimization transplanted into a hyper-heuristic structure for solving examination timetabling problem,” *Swarm and Evolutionary Computation*, vol. 7, pp. 21–34, 2012.
 - [41] A. Gunawan, K. M. Ng, and K. L. Poh, “A hybridized Lagrangian relaxation and simulated annealing method for the course timetabling problem,” *Computers & Operations Research*, vol. 39, no. 12, pp. 3074–3088, 2012.
 - [42] B. Edmund and P. Ross, “Examination Timetabling in British Universities: A Survey,” in *Practice and Theory of Automated Timetabling*, Springer, Berlin, Germany, 1996.