Optimization Of University Course Scheduling Problem With A Hybrid Artificial Bee Colony Algorithm

Adalet Oner, Sel Ozcan, Derya Dengi
Department of Industrial Engineering
Yasar University
Izmir, Turkey
adalet.oner@yasar.edu.tr

***Abstract*— Course scheduling problem (CSP) is concerned with developing a timetable that illustrates a number of courses assigned to the classrooms. In this study, a hybrid algorithm composed of a heuristic graph node coloring (GNC) algorithm and artificial bee colony (ABC) algorithm is proposed to solve CSP. The study is one of the few applications of ABC on discrete optimization problems and to our best knowledge it is the first application on CSP. A basic heuristic algorithm of node coloring problem takes part initially to develop some feasible solutions of CSP. Those feasible solutions correspond to the food sources in ABC algorithm. The ABC is then is used to improve the feasible solutions. The employed and onlooker bees are directed or controlled in a specific manner in order to avoid the conflicts in the course timetable. Proposed solution procedure is tested using real data from a university in Turkey. The experimental results demonstrate that the proposed hybrid algorithm yields efficient solutions.**

Keywords- course scheduling problem; node coloring; artificial bee colony algorithm

I. INTRODUCTION

Course scheduling problem (CSP) involves developing a timetable that requires assigning courses to timeslots across a week in a conflict-free manner with respect to the sections, instructors and classrooms. The problem includes optimization of the performance criteria with various additional constraints such as curriculum planning policies, proper classroom assignment and available teaching resources.

II. LITERATURE REVIEW

The course scheduling problem (CSP) is one of the combinatorial optimization problems that has been studied over many years. An earlier survey on that problem is reported in [24]. Various meta-heuristic based methods are implemented to solve CSP such as genetic algorithms [8, 9, 12, 15], simulated annealing [2, 14], tabu search [10,17,18,23], ant colony optimization [6] and particle swarm optimization [1]. Mainstream meta-heuristic based methods are implemented and compared in [6]. On the other hand, hybrid algorithms were used for CSP in various studies. For example, a hybrid algorithm was introduced in [3] where integer programming, a greedy heuristic and a modified SA were combined. Other implementations of hybrid algorithms are [4, 11, 21]. Another hybrid algorithm is proposed recently in [1] which is the first application of particle swarm optimization for course scheduling problem where flexible preferences were also considered. As it was stated in [7], with the integration of hybrid heuristics and a combination of heuristics with artificial intelligence, the solution quality will definitely be improved. On the other hand, graph theory, network models, mixed integer programming, and constrained logic programming are other methods used to formulate and solve CSP [16, 20,25,26,28,30,31,32]. A mixed integer model is developed as a special case of fixed charge transportation problem in [17]. On the other hand, goal programming is used to formulate and solve CSP in [13].

III. PROBLEM DEFINITION

The course scheduling problem in Yasar University consists of a set of courses to be scheduled in 40 timeslots across five days and eight periods a day. At the beginning of each semester, several courses are offered to students and each course can be divided into several sections depending on the number of students enrolled. The curriculum imposes the weekly number of lecture hours for each course. It is assumed that the instructors are pre-determined for each course and/or for each course sections. The type and the capacity of each available classroom are known. The task is to distribute the lectures into 40 timeslots associated with proper classrooms. The aim in general is to generate a conflict-free course timetable while optimizing the performance criterion.

A valid timetable has to meet several requirements which can be divided into two categories: "must" requirements and some "preference" requirements. The "must" requirements bear the hard constraints that must be satisfied in order to generate feasible solutions. They can be stated as follows.

- An instructor can not be assigned to more than one lecture in the same timeslot.
- A section (student group) can not be assigned to more than one lecture in the same timeslot.
- Only one lecture can be assigned to a classroom in a specific timeslot.

978-1-4244-7835-4/11/$26.00 ©2011 IEEE

- The number of lecture hours assigned for each course should be the same as the weekly number of lecture hours stated in the curriculum.
- A section (student group) can not be assigned to a classroom such that the number of students in that section exceeds capacity of the classroom.
- Some lectures have to be scheduled in a particular classroom, such as computer lab.

On the other hand, “preference” requirements bring the soft constraints which can be violated if necessary. The following soft constraints are considered.

- The preferences of the instructors and the students about the assigned timeslots should be taken into consideration. The preferences may be expressed in numeric values denoting the dissatisfaction rate for each timeslot.
- The lecture hours (periods) of any course should be scheduled in a format of consecutiveness determined by the instructor. For example, a course with 4 lecture hours may be scheduled using (4-0) format meaning that all four lecture hours should be scheduled consecutively at the same day as a single session. On the other hand, the format (2+2) represents two distinct sessions in different days and each session consists of two consecutive lecture hours.
- The instructors should have a teaching free day.

It may not be possible to satisfy all the soft constraints, and therefore they may be violated as necessary. However some priority rules may be applied between the constraints and instructors. The instructors indicate their preferences, along with the timeslots across a week using a preference table. A sample preference table is shown in Figure 1.

Each instructor prepares a preference table and the entries in that table are expressed within the range [0 , 10]. Those entries indicate the dissatisfaction degree of the instructor if his/her lecture is assigned to the corresponding timeslots. The lower value is the more desirable timeslot to be assigned. The dissatisfaction values for the instructor i at timeslot t represented by DSV_{it}, t=1….40.

PERIOD	Mon	Tue	Wed	Thu	Fri
1	10	10	10	10	9
2	9	4	2	8	9
3	2	0	0	0	9
4	2	0	0	0	9
5	1	2	1	1	10
6	1	2	1	2	10
7	7	5	6	3	10
8	7	5	6	3	10

Figure 1. Preference table for a instructor

Let C={C_1, C_2, …, C_n} be the set of n courses and LH_j be the number of weekly lecture hours for the corresponding course. A part of the set of courses and corresponding weekly hours are shown in Figure 2.

Course Name	**Code C_j**	**Weekly Hours LH_j**	**The Set of Sections taking this course**
Project Management	C_1	3	NS_1 ={ S_5 , S_6}
System Simulation	C_2	4	NS_2 ={ S_5 , S_6 }
Eng. Mechanics	C_3	4	NS_3 ={ S_3 , S_4 }
……	C_j	….	NS_j ={ …S_p , $S_{r…}$}
……	….	…..	
Stochastic Processes	C_n	3	NS_n ={ S_7 }

Figure 2. Courses and Sections

The set of k sections is represented by S= {S_1, S_2, ., S_k}. A section represents a pre-determined group of regular students at each class of each department. Let NS_j = {…S_p, S_r …} be a subset of S and represents the set of sections that take the course C_j.

Having the definitions above, we may now define Q = {Q_1, Q_2, …., Q_{NQ}} be the set of all single lecture hours to be scheduled. Those single lecture hours may be shown as in Figure 3.

Course Name	**Code C_j**	**Section S_k**	**Single Lecture Hour**	
Project Management	C_1	S_5	Q_1	Section S_5 takes Project Management course. Three lecture hours should be scheduled for this section.
Project Management	C_1	S_5	Q_2	
Project Management	C_1	S_5	Q_3	
Project Management	C_1	S_6	Q_4	Section S_6 also takes Project Management course. Another three lecture hours should be scheduled for this section.
Project Management	C_1	S_6	Q_5	
Project Management	C_1	S_6	Q_6	
……	…..	…..	….	….
……	C_n	S_k	Q_{NQ}	

Figure 3. List of Single Lecture Hours

NQ represents the number of elements in Q, in other words, total number of single lecture hours to be scheduled. It may be defined as follows.

$$NQ = \sum_j (LH_j) * \left| NS_j \right| \qquad (1)$$

where LH_j is weekly lecture hours for course (j) and $|NS_j|$ is the number of sections that take course (j).

Let R= {R_1, R_2, …..,R_w } be the set of *w* classrooms to be scheduled. The problem may now be expressed as distributing NQ single lecture hours into (40*w) boxes while ensuring any conflict with respect to instructors, sections and classrooms is avoided. The objective is minimizing total dissatisfaction value.

$$Min.Z = \sum_{i \in L} \sum_{t}^{40} DSV_{it} * X_{it} \qquad (2)$$

where X_{it} is the binary variable representing whether the single lecture hour taught by the instructor *i* is scheduled in timeslot *t* and the set L = {L_1, L_2, …..,L_i } represents the set of lecturers (instructors). Remember that the instructor and the section are the pre-determined attributes of each single lecture hours.

IV. HYBRID (GNC-ABC) ALGORITHM

The proposed hybrid algorithm is composed of heuristic graph node coloring (GNC) and artificial bee colony (ABC) algorithms. Thus it is referred as GNC-ABC algorithm. A basic graph-node coloring algorithm is used to generate some feasible solutions that fully satisfy hard constraints and partly satisfy the consecutiveness constraint. Afterwards, the ABC algorithm is incorporated for massaging the feasible solutions in order to get improvement in the objective function.

A. Graph Node Coloring Algorithm (GNC)

The graph-node coloring problem involves assigning colors to nodes in a graph such that "neighbor" nodes have distinct colors. If a node is connected to another node, then those nodes are regarded as neighbors. The problem is often to determine the minimum cardinality (the number of different colors) to color all the nodes in the graph.

GNC algorithm were used in modeling and solving the course scheduling problem in the past [19,22,27,29]. In this study, a graph is generated with the nodes representing distinct single lecture hours to be scheduled. Each node contains the associated information consisting of the course code, instructor code and section code. If any one of those three attributes is the same for two nodes, then they are connected to each other with an arc which makes them neighbor nodes. Once the graph is generated, you need to color all the nodes, but the same color is prohibited for the neighbor nodes. The colors represent the timeslots herein. Using different timeslots for neighbor nodes ensures to satisfy the hard constraints of the course scheduling problem.

Although graph-node coloring problem is known to be NP-complete, various efficient algorithms are developed to solve it. A widely-used general greedy based approach involves getting an ordered node enumeration and then sequentially assigning a color for each node in the list with the smallest possible color code inside the current color set. In other words, there are two sets, the first one is the ordered list of nodes and the latter one is the ordered list of colors. The list of colors is ordered by their code. Procedure starts with choosing the first node from the node list and it tries to assign the first color to that node. If that color is not used for a neighbor node before, it is allowed to assign it for that node. Otherwise it tries the next color and so on. Once a node is associated with a color, the same process is repeated sequentially for other nodes until all nodes are colored.

However, the ordering of the nodes dramatically affects the coloring. The arbitrary order may perform very poorly while another ordering may produce an optimal coloring. The projection of a poor solution on the course scheduling problem is that it would require more than 40 timeslots (40 different colors) to develop a course schedule. Several ordering algorithms have been studied to help the greedy coloring heuristics including largest degree-first ordering, largest saturation degree-first ordering and incidence degree ordering. The degree of a node is the number of arcs connected to that node. The saturation degree, on the other hand, represents the total number of degrees of the neighbor nodes.

In this study, a variation of general approach is used. The nodes (lecture hours) are ordered by their degree and largest degree-first policy is implemented. Once a node (lecture hour) is colored (assigned to a timeslot), it is associated to one of the available classrooms randomly depending on the desired classroom type and capacity. If the timeslot of that classroom is used before, the color is erased and the next color (timeslot) is tried from the set of colors.

It is tested that the minimum cardinality (minimum number of different colors required) is always smaller than 40. Therefore, the solution coming out of this approach always delivers a feasible solution for the CSP which satisfies the hard constraints. In order to generate different feasible solutions, the colors are ordered randomly for each time instead of ordering them by their code. This procedure delivers different feasible solutions, however soft requirements may not be met. In order to get some improvement in the consecutiveness requirement, the structure of the nodes is changed a little bit. A node is designed such that each one represent a block of two consecutive lecture hours of the same course. The attributes (instructor, course and section) do not change. Having this modification provides partial improvement in consecutiveness requirement, but more important than that, it brings the advantage of decreasing the number of nodes. The solutions obtained from GNC algorithm provide inputs for the second phase of the proposed GNC-ABC algorithm.

A sample feasible solution delivered by GNC algorithm is given in Figure 4. Each entry is composed of two different attributes. The first one is the code of the course, and the latter one is the instructor's nick.

Classroom 001

Mon	Tue	Wed	Thu	Fri
		IENG206 demir		
	IENG304 kozan			

Classroom 002

Mon	Tue	Wed	Thu	Fri
IENG354 guldogan				
MATH112 gurkan				MATH232 gurkan
IENG212 bulut	EENG242 ungan			

Classroom 003

Mon	Tue	Wed	Thu	Fri
	EENG112 yilmaz	IENG202 sarman		EENG112 yilmaz
EENG112 yilmaz	EENG242 ungan	IENG202 sarman		IENG462 kozan
EENG242 ungan				IENG212 bulut
IENG306 tasgetiren	IENG206 demir	MATH112 gurkan		MATH112 gurkan

Classroom 004

Mon	Tue	Wed	Thu	Fri
	CENG132 zincir	IENG204 kozan	IENG204 kozan	
				IENG102 sarman
IENG102 sarman			IENG102 sarman	
		CENG132 zincir		IENG204 kozan

Classroom 005

Mon	Tue	Wed	Thu	Fri
	MATH232 gurkan	MATH112 gurkan		IENG306 tasgetiren
	IENG206 demir	IENG304 kozan		IENG302 oner
IENG458 demir	MATH232 gurkan			IENG306 tasgetiren
IENG402 gurkan	IENG312 oner			IENG458 demir

Classroom 006

Mon	Tue	Wed	Thu	Fri
		IENG302 oner		MATH232 gurkan
	IENG354 guldogan	EENG112 yilmaz	MATH112 gurkan	EENG242 ungan
IENG354 guldogan	IENG206 demir		IENG212 bulut	MATH112 gurkan
IENG354 guldogan	IENG402 gurkan		IENG212 bulut	IENG306 tasgetiren

Classroom 007

Mon	Tue	Wed	Thu	Fri
	IENG462 kozan	IENG464 guldogan		
		IENG304 kozan		
			IENG312 oner	IENG202 sarman

Figure 4. Sample Feasible Solution Delivered by GNC Algorithm

Classroom 008

Mon	Tue	Wed	Thu	Fri
	IENG312 oner	BUSN485 tasgetiren		
IENG304 kozan		IENG464 guldogan		CENG132 zincir
IENG312 oner	BUSN485 tasgetiren			IENG302 oner
		IENG202 sarman	CENG132 zincir	IENG302 oner

Classroom 009 (Physics Lab)

Mon	Tue	Wed	Thu	Fri
		EENG112 yilmaz		
		EENG112 yilmaz		

Classroom 010 (CAD Lab)

Mon	Tue	Wed	Thu	Fri
			IENG104 kocaman	
	IENG104 kocaman	IENG104 kocaman	IENG104 kocaman	

Classroom 011

Mon	Tue	Wed	Thu	Fri

Classroom 012 (CAM Lab.)

Mon	Tue	Wed	Thu	Fri
	IENG406 sarman			
	IENG406 sarman			
IENG204 kozan			IENG204 kozan	

Classroom 013

Mon	Tue	Wed	Thu	Fri
CENG132 zincir		IENG102 sarman		
		IENG204 kozan		
	CENG132 zincir			

Classroom 014 (Eln Lab)

Mon	Tue	Wed	Thu	Fri
			EENG242 ungan	
EENG242 ungan				

Figure 4. (continued) Sample Feasible Solution Delivered by GNC Algorithm

B. *Artificial Bee Colony Algortihm (ABC)*

Karaboga [5] proposed the artificial bee colony (ABC) algorithm which is an optimization algorithm based on the intelligent foraging behavior of honey bee swarm. ABC is a population based algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The general scheme of the ABC algorithm is as shown in Figure 5.

```
Initialization Phase
Cycle =1
 REPEAT
      Employed Bees Phase
      Onlooker Bees Phase
      Scout Bees Phase
      Memorize the best solution achieved so far
      Cycle =Cycle+1
UNTIL(Cycle=Maximum Cycle Number)
```

Figure 5. The general scheme of the ABC algorithm

At the first step, a randomly distributed initial population (food source positions) is generated. After initialization, the population is subjected to repeat the cycles of the search processes of the employed, onlooker, and scout bees, respectively. An employed bee produces a modification on the source position in her memory and discovers a new food source position (local search). Provided that the nectar amount of the new one is higher than that of the previous source, the bee memorizes the new source position and forgets the old one. Otherwise she keeps the position of the one in her memory. After all employed bees complete the search process, they share the position information of the sources with the onlookers on the dance area. Each onlooker evaluates the nectar information taken from all employed bees and then chooses a food source depending on the nectar amounts of sources. As in the case of the employed bee, she produces a modification on the source position in her memory and checks its nectar amount. Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one. The sources abandoned are determined and new sources are randomly produced to be replaced with the abandoned ones by artificial scouts.

In this study, graph-node algorithm is used to produce initial population. They correspond to the food source positions in the initialization phase of the ABC algorithm. A food source position (a feasible solution of CSP) is represented by a one-dimensional array. That array is the converted form a feasible solution. The process of converting a feasible solution into a one-dimensional array may be described as follows. Consider the timetables of the classrooms of a course schedule. The positions of timeslots in the classrooms timetables are enumerated sequentially and shown in Figure 6.

Mon	Tue	Wed	Thu	Fri
Classroom 001				
1	5	9	13	17
2	6	10	14	18
3	7	11	15	19
4	8	12	16	20
Classroom 002				
21	25	29	33	37
22	26	30	34	38
23	27	31	35	39
24	28	32	36	40
Classroom 003				
41	45	49	53	57
42	46	50	54	58
..	…	…	…	…

Figure 6. Enumarating timeslots of classrooms

The result of the GNC algorithm fills some entries the in the classroom timetables and the outcome may look like as in Figure 7. Remember that Q_i is an element of ordered set of lecture hours (nodes) to be scheduled. Then, the nodes' positions in Figure 7 are matched with the position numbers shown in Figure 6. Resulting array shown in Figure 8 is the converted form of a feasible solution.

Mon	Tue	Wed	Thu	Fri
Classroom 001				
Q10	Q6			
	Q5			
				Q4
Classroom 002				
Q7	Q1			
				Q2
	Q6			
Classroom 003				
Q3				
Q8	Q9			

Figure 7. Sample timetable for classrooms

Node	Q1	Q2	Q3	…	Q7	…	Q10	…	Q_{NQ}
Position	25	38	41	…	21	…	1	…	..

Figure 8. Array representation of course schedule

The GNC algorithm is run several times with random orders of color (timeslot) set in order to get a population of those arrays. These arrays are the position vectors of foods resources in ABC algorithm. Let X_m be such a vector, m=1,2,…,SN where SN is the population size. Since each food source, X_m, is a feasible solution vector to the course scheduling problem, each X_m vector holds QN variables, ($X_{mi,}$ i=1,2,…..QN), which are to be optimized so as to minimize the objective function stated in (2).

1) Employed Bees Phase

The employed bees search for new food sources (V_m) having more nectar within the neighborhood of the food source (X_m) in their memory. They find a neighbor food source and then evaluate its profitability (fitness). For example, they can determine a neighbor food source V_m as follows.

$$v_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki}) \tag{3}$$

where X_k is a randomly selected food source, (i) is a randomly chosen parameter index and Φ_{mi} is a random number within the range [-1,1]. At that point, a detail of the process needs to be explained. The value of V_{mi} is obviously a real number which is converted to an integer number by simply rounding it. On the other hand, the new value should be in the allowed range [1, 20**w*] where *w* is the total number of classrooms. If it is not, then V_{mi} is calculated again using another random Φ_{mi} value. This situation is a modification of "employed bee phase" because of the special structure of course scheduling problem. After producing the new food source, its fitness is calculated and a greedy selection is applied between X_m and V_m . The fitness value of the solution is calculated using (2).

The projection of employed bees phase on the course scheduling problem is the process of changing the scheduled timeslot and probably the classroom of a lecture hour. The feasibility of that change is checked first. If a conflict occurs, it is not accepted and X_{mi} remains unchanged.

2) Onlooker Bees Phase

An onlooker bee chooses a food source depending on the probability values calculated using the fitness values provided by employed bees. For this purpose, a fitness based selection technique can be used, such as the roulette wheel selection method. The probability value (p_m) with which X_m is chosen by an onlooker bee can be calculated as follows.

$$p_m = \frac{fit_m(\vec{x_m})}{\sum_{m=1}^{SN} fit_m(\vec{x_m})} \tag{4}$$

After a food source for an onlooker bee is probabilistically chosen, a neighborhood source V_m is determined by using (3), and its fitness value is computed. As in the employed bees phase, a greedy selection is applied between X_m and V_m. Hence, more onlookers are recruited to richer sources and positive feedback behavior appears.

3) Scout Bees Phase

The unemployed bees which choose their food sources randomly are called scouts. Employed bees whose solutions cannot be improved through a predetermined number of trials, specified by the user of the ABC algorithm and called "limit" or "abandonment criteria" herein, become scouts and their solutions are abandoned. Then, the converted scouts start to search for new solutions using GNC.

V. EXPERIMENTAL RESULTS

The proposed hybrid algorithm is tested using real data from Yasar University in Turkey. Actually it is the course scheduling problem of Industrial Engineering Department. The number of nodes 86 and total number of classrooms is 14. The experimental results demonstrate that the proposed hybrid algorithm yields efficient solutions to course scheduling problem. The following control parameters are found to be effective in solving the problem:

- swarm size : 40
- number of employed bees : 20
- number of onlooker bees : 20
- number of scout bees : 1
- abandonment criteria : 86
- cycle number : 1500

A sample solution delivered by proposed algorithm are given in Figure 9.

Mon	Tue	Wed	Thu	Fri
Classroom 001				
	IENG202 sarman			
			CENG132 zincir	
IENG304 kozan		CENG132 zincir	IENG302 oner	BUSN485 tasgetiren
	BUSN485 tasgetiren		IENG464 guldogan	
Classroom 002				
	IENG206 demir			
BUSN306 tasgetiren		IENG402 gurkan		
	IENG304 kozan	IENG306 tasgetiren		

Mon	Tue	Wed	Thu	Fri
Classroom 003				
	IENG354 guldogan			
IENG206 demir		IENG212 bulut	IENG212 bulut	
IENG312 oner	IENG354 guldogan			
Classroom 004				
		IENG302 oner	MATH112 gurkan	
	IENG462 kozan			
MATH112 gurkan	IENG212 bulut			
		IENG458 demir		
Classroom 005				
		IENG306 tasgetiren		
	MATH232 gurkan	EENG242 ungan	IENG202 sarman	
				MATH232 gurkan
	IENG206 demir			
Classroom 006				
	EENG112 yilmaz	IENG202 sarman	IENG212 bulut	
			IENG304 kozan	EENG112 yilmaz
EENG112 yilmaz	MATH232 gurkan	EENG242 ungan		
IENG354 guldogan	EENG242 ungan			
Classroom 007				
	IENG462 kozan	MATH112 gurkan	IENG302 oner	MATH112 gurkan
		IENG206 demir		IENG458 demir
IENG354 guldogan		IENG402 gurkan		IENG312 oner
	MATH112 gurkan	IENG458 demir	MATH232 gurkan	
Classroom 008				
MATH112 gurkan				
	IENG204 sarman			IENG306 tasgetiren
CENG132 zincir		CENG132 zincir		
Classroom 009 (Physics Lab)				
			EENG112 yilmaz	
				EENG112 yilmaz

Figure 9. Sample Solution Developed by Proposed Algorithm

Mon	Tue	Wed	Thu	Fri
Classroom 010 (CAD Lab)				
IENG104 kocaman		IENG104 kocaman		
	IENG104 kocaman		IENG104 kocaman	
Classroom 011				
				CENG132 zincir
	IENG102 sarman			
CENG132 zincir		IENG102 sarman		IENG204 kozan
IENG102 sarman		IENG204 kozan		
Classroom 012 (CAM Lab.)				
				IENG204 kozan
			ENG406 sarman	
	IENG406 sarman	IENG204 kozan		
Classroom 013				
	IENG312 oner	IENG464 guldogan		IENG302 oner
		IENG304 kozan		IENG202 sarman
EENG242 ungan		IENG312 oner		
Classroom 014 (Eln Lab)				
			EENG242 ungan	
	EENG242 ungan			

Figure 9.(Continued) Sample Solution Developed by Proposed Algorithm

REFERENCES

[1] Shiau, D., " A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences." *Expert Systems with Applications*, 38, pp. 235-248, 2011.

[2] Pongcharoen, P., Promtet, W., Yenradee, P., and Hicks, C., "Stochastic optimisation timetabling tool for university course scheduling." *International Journal of Production Economics*, 112(2), 903–918, 2008.

[3] Gunawan, A., Ng, J. M., & Poh, K. L., "Solving the teacher assignment-course scheduling problem by a hybrid algorithm." *International Journal of Computer, Information, and System Science, and Engineering*, 1(2), 136–141, 2007.

[4] Chiarandini, M., Birattari, M., Socha, K., and Rossi-Doria, O., "An effective hybrid algorithm for university course timetabling." *Journal of Scheduling*, 9(5), 403–432, 2006.

[5] Karaboga, D., “An idea based on honey bee swarm for numerical optimization” Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005

[6] Azimi, Z. N., “Hybrid heuristics for examination timetabling problem.” *Applied Mathematics and Computation*, 163(2), 705–733, 2005.

[7] Duong, T. A., & Lam, K. H., “Combining constraint programming and simulated annealing on university exam timetabling.” *In Proceedings of international conference RIVF’04, Hanoi* (pp. 205–210), 2004.

[8] Wang, Y. Z., “Using genetic algorithm methods to solve course scheduling problems.” *Expert Systems with Applications*, 25(1), 39–50, 2003.

[9] Chiarandini, M., & Stützle, T., “*A landscape analysis for a hybrid approximate algorithm on a timetabling problem.*” TU Darmstadt Technical Report AIDA-03-05, 2003.

[10] Alvarez-Valdes, R., Crespo, E., & Tamarit, J. M., “Design and implementation of a course scheduling system using tabu search.” *European Journal of Operational Research*, 137(3), 512–523, 2002.

[11] Merlot, L. T. G., Boland, N., Hughes, B. D., and Stuckey, P. J,. “A hybrid algorithm for the examination timetabling problem.” In E. Burke & M. Carter (Eds.), *Practice and theory of automated timetabling IV. Lecture notes in computer science* (Vol.2740, pp. 207–231). New York: Springer-Verlag, 2002.

[12] Wang, Y. Z., “An application of genetic algorithm methods for teacher assignment problems.” *Expert Systems with Applications,* 22(4), 295–302, 2002.

[13] Badri, M.A., Davis, D. L., Davis, D. F. and Hollingsworth, J., “A multi-objective course scheduling Model: combining faculty preferences for courses and times.” *Computers and Operations Research* 25 (4), 303–316, 1998.

[14] Thompson, J. M., and Dowsland, K. A., “A robust simulated annealing based examination timetabling system.” *Computers and Operations Research*, 25(7-8), 637–648, 1998.

[15] Drexl, A., & Salewski, F., “Distribution requirements and compactness constraints in school timetabling.” *European Journal of Operational Research*, 102(1), 193–214, 1997.

[16] Deris, S. B., Ormatu, S., Ohta, H. and Samat, P.A., “B.D. *University□ timetabling□ by□ constraint-based□ reasoning:□ a□ case□ study.*” *Journal of the Operational Research Society*, **48**,□1178□1190,□1997.□

[17] Hultberg, T.H. and Cardoso, D.M., “*The□ teacher□ assignment□ problem:□ a□ special□ case□ of□ fixed□ charge□ transportation□ problem.*” *European Journal of Operational Research*, **101**, 463–473,□ 1997.□

[18] Alvarez-Valdes,R., Martin, G. and Tamarit, J.M., “Construction good solutions for the Spanish school timetabling problem.” *Journal of Operational Research Society,* Vol. 47, pp. 1203-1215, 1996.

[19] Carter, M.W., Laporte, G. and Lee, S.Y., “Examination timetabling: Algorithmic Strategies and Applications.” *Journal of Operational Research Society,* 47, 373-383, 1996.

[20] Wright, M., “School timetabling using heuristic search.” *Journal of Operational Research Society*, 47, 347-357, 1996.

[21] Weare, R., Burke, E., & Elliman, D., *“A hybrid genetic algorithm for examination timetabling problem.”* University of Nottingham, Computer Science Technical Report No. NOTTCS-TR-1995-8, 1995.

[22] Carter, M.W., Laporte, G. and Chinneck, J.W., “A general examination scheduling system.” *Interfaces,* 24, 109-120, 1994.

[23] Hertz, A., “Tabu search for large scale timetabling problems.” *European Journal of Operataions Research*, 54, 39-47, 1991.

[24] Carter, M.W., “A survey of practical applications examination timetabling algorithms.” *Operations Research*, 34, 193-202, 1986.

[25] Laporte, G., and Desroches, S., “Examination timetabling by computer.” *Computers and Operations Research*, 11, 351-360, 1984.

[26] McClure, R. and Wells, C., “A mathematical model for faculty course assignment.” *Decision Science*, 15, 409-420, 1984.

[27] Mehta, N.K., “Application of a graph coloring method to an examination scheduling problem.” *Interfaces*, 11, 57-64, 1981.

[28] Tripathy, A., “A Lagrangean relaxation approach to course timetabling.” *Journal of Operational Research Society*, 31, 599-603, 1980.

[29] J. Brelaz, D., “New methods to color the vertices of a graph.” *Comm. A.C.M.* 22, pp. 251-256, 1979.

[30] Shih, W., and Sullivan, J., “Dynamic course scheduling for college faculty via zero-one programming.” *Decision Science*, 8, 711-721, 1977.

[31] Lawrie, N. L., “An integer linear programming model of a school timetabling problem.” The Computer Journal, 12, 307–316, 1969

[32] Akkoyunlu,□E.A.,□□A linear algorithm for computing the optimum university timetable.□□ *The□Computer□Journal,*□**16**□(4),□347□350,□1973.□