HIGHLY CONSTRAINED UNIVERSITY CLASS SCHEDULING USING ANT COLONY OPTIMIZATION

Al-Mahmud

Department of Computer Science and Engineering (CSE), KUET, Khulna-9203, Bangladesh

ABSTRACT

Solving University Class Scheduling Problem (UCSP) is a complex real-world combinatorial optimization task that has been extensively studied over the last several decades. Many meta-heuristic based techniques, including prominent swarm intelligence (SI) methods have been investigated to solve it in different ways. In this study, Ant Colony Optimization (ACO) based two methods are investigated to solve UCSP: ACO based method and ACO with Selective Probability (ACOSP). ACO is the well-known SI method that differs from other SI based methods in the way of interaction among individuals (i.e., ants); and an ant interacts with others indirectly through pheromone to solve a given problem. ACO based method considers probabilistically all the unassigned time slots to select next solution point for a particular course assignment. In contrast, ACOSP probabilistically selects next solution point for a particular course assignment from the selective probabilities. Such selective probability employment with ACO improves performance but reduces computational cost. The performances of the proposed methods have been evaluated comparing with Genetic Algorithm (GA) in solving real-world simple UCSPs. In addition, proposed methods are compared with each other for solving highly constrained UCSPs. Both the proposed methods outperformed GA and ACOSP was the best to solve the given problems.

KEYWORDS

University Class Scheduling Problem (UCSP), Ant Colony Optimization (ACO), and Selective Probability (SP).

1. INTRODUCTION

University Class Scheduling Problem (UCSP) is a scheduling problem concerned with assignment of classes for instructors and students among suitable time slots satisfying a set of constraints. UCSP can be described as: given a set of classes, a set of (contiguous) time slots, a set of students, and a set of instructors, the task is to assign classes to time slots for instructors and students satisfying a set of hard and soft constraints. Mathematically, the UCSP is defined as triple $\langle E, T, C \rangle$, where $E = \{c_i, s_j, i_k\}$. contains three triples: set of classes c_i, set of students s_j, and set of instructors i_k. $T = \{t_1, ..., t_n\}$ is a set of time slots and $C = \{c_1, ..., c_n\}$ is the set of hard and soft constraints. The task is to assign E_i to the time slot T_i satisfying constraints C_i where $C_i \in C$.

There are usually two types of constrains involved in UCSP: hard constraints [1, 2, 7] and soft constraints [1, 2, 7]. Hard constraints must be satisfied completely so that the generated solutions become feasible solutions. Soft constraints which are related with objective function and they must be satisfied as much as possible, but it is not necessary that soft constraints are being satisfied as hard constraints. UCSP has to maintain various types of constraints such as

instructors' constraints, students' constraints, rooms' constraints, and administrative constraints etc. The main feature of UCSP is to satisfy soft constraints as much as possible where there must be no violation of hard constraint. The main purpose of the study is to solve course scheduling problem of Khulna University of Engineering & Technology (KUET).It is to be noted that both hard and soft constraints may vary from university to university.

The UCSP has been extensively studied over the last 25 years [5]. Different Swarm Intelligence (SI) based techniques are investigated to solve UCSP. The popular SI based techniques are adaptive Particle Swarm Optimization (PSO) [2], PSO based algorithm with local search [3], PSO with interchange heuristic [14] and honey-bee mating optimization algorithm [11] are well studied for solving UCSP. Hybrid Particle Swarm Optimization (HPSO), one of the most recent algorithms, includes features to consider instructor's preferences, and employs a repair process updating instructor timetable with re-generated feasible time slots to solve UCSP [4].There are also different meta-heuristic based techniques such as Tabu Search (TS) based techniques [9, 10] and Genetic Algorithm (GA) based techniques [6, 8] have been investigated to solve UCSP. Moreover, other well-known techniques are also found effective to solve UCSP in the literature [12, 13]. Ant Colony Optimization (ACO) is another well-known SI based method which has not been used to solve UCSP. Therefore, there is a scope to solve UCSPs using ACO based methods.

2. METHODOLOGY

There are two main reasons for solving UCSP is a challenging task. The first reason is the exponential growth problem of this problem is due to the faster growth students and secondly the numbers of constraints are varied from university to university. The proposed method follows ACO based strategy to solve UCSP. Based on this, two algorithms are proposed: ACO and ACO with Selective Probability (ACOSP). Originally, ACO was implemented for Travelling Salesman Problem (TSP), a well-known combinatorial optimization problem. To solve TSP using ACO, the number of cities is considered equal to the number of ants. But, in our proposed methods, each ant represents an UCSP solution. In the proposed method, the individual instructor's suitable solutions are found by calculating total preference value. The formula for calculating fitness of individual instructor, *iif* is given by-

$$iif = \sum_{i=1}^{n} PV_i \qquad (1)$$

PV = Preference values of assigned time slots *n* for each instructor.

For each triple (class, instructor, batch), $e \in E$, an ant chooses a time slot $t \in T$ probabilistically. The ants construct a partial assignment $A_i : E_i \to T$ for $(i = 0,1,.....|E|)$, where $E_i = \{e_1, e_2, ...e_i\}$. An ant starts with an empty assignment $A_i = \phi$. After construction of A_{i-1}, the assignment is built probabilistically as $A_i = A_{i-1} \cup \{(e_i, t)\}$. The time slot t is chosen probabilistically out of T based on probabilities $p(e_i, t)$, that depends on the pheromone matrix $\tau(A_{i-1})$ and the heuristic information $\eta(A_{i-1})$ is given by-

$$p(e_i,t)(\tau(A_{i-1}),\eta(A_{i-1})) = \frac{(\tau(e_i,t)(A_{i-1}))^{\alpha}.(\eta(e_i,t)(A_{i-1}))^{\beta}}{\sum_{\theta \in T}(\tau(e_i,\theta)(A_{i-1}))^{\alpha}.(\eta(e_i,\theta)(A_{i-1}))^{\beta}} \qquad (2)$$

The impact of the pheromone and the heuristic information can be weighted by parameters α and β respectively and the pheromone matrix is given by $\tau(A_i) = \tau_0$ where $(i = 0,1,....|E|)$. A simple method for computing the heuristic information is the following:

$$\eta((e_i,t)(A_{i-1})) = \frac{1.0}{1.0 + (v(e_i,t)(A_{i-1}))} \qquad (3)$$

Where $(v(e_i,t)(A_{i-1}))$ counts the additional number of soft constraints violations caused by adding (e_i,t) to the partial assignment A_{i-1} .

Let $Agbest$ be the assignment of the best solution found since the beginning. The following update rule is used:

$$\tau(e,t) = \begin{cases} (1-\rho).\tau(e,t)+2 & Agbest(e)=t \\ (1-\rho).\tau(e,t) & Otherwise \end{cases} \qquad (4)$$

Here ρ is pheromone evaporation coefficient.

The fitness of the UCSP solution is measured for k number of instructors is given by-

$$UCSPfitness = \sum_{j=1}^{k} iif_j \qquad (5)$$

Incorporation of selective probability (SP) technique with the ACO to form a novel algorithm named ACOSP in which probabilities of suitable unassigned time slots, instead of all the unassigned time slots is maintained to select a particular time slot. As a result, better result will be obtained. Another benefit of incorporating selective population is to provide better convergence.

2.1. ACO to Solve UCSP

In the proposed ACO based algorithm, each ant constructs a UCSP solution by choosing time slot of classes probabilistically. The probability of choosing time slots of classes is a function of pheromone value and heuristic information. The less conflicting (i.e., high heuristic value) route will be increasingly enhanced, and therefore become more attractive. On the other hand, more conflicting route will eventually disappear because pheromones are volatile. Initially, different ants construct different solutions. Overtime, the solutions will converge to a single optimal UCSP solution.

The pseudo code of proposed ACO based algorithm is given below:

Input: (Courses, Instructors, Batch), E for solving UCSP
Output: An optimal solution of UCSP
1: Assign preferences in each slot for every instructor
$2: \tau_0 = 1/\rho$
$3: \tau(e,t) = \tau_0 \forall (e,t) \in E \times T$
4: **while**(time limit not reached) do
5: **for** (a=1 to m) do // m is the number of ants

$A_0 = \phi$

6: **for** (i=1 to $|E|$) **do**

7: Compute probability using Eq. (3.2.2)

8: for each $\boldsymbol{e_i}$ choose time slot *t* probabilistically

$A_i = A_{i-1} \cup \{(e_i, t)\}$

9: Compute fitness value using Eq. (3.2.5) and save the best solution

10: **End for**

11: **End for**

12: Update global pheromone value

13: **End while**

2.2. ACO with Selective Probabilities (ACOSP) to Solve UCSP

The ACO based method use all unassigned time slots to select a particular time slot. The unassigned suitable time slot has chance to produce better optimal UCSP solution than the all unassigned time slots. To utilize this, modifications of the ACO is made and incorporate selective population to form another algorithm named ACOSP in which probabilities of suitable unassigned time slots, instead of all the unassigned time slots is maintained to select a particular time slot. As a result, better result will be obtained. Another benefit of incorporating selective probabilities is to provide better convergence. The bold text shows difference of these two algorithms.

The pseudo code of proposed ACOSP based algorithm is given below:

Input: (Courses, Instructors, Batch), *E* for solving UCSP
Output: An optimal solution of UCSP

1: Assign preferences in each slot for every instructor

$2: \tau_0 = 1/\rho$

$3: \tau(e,t) = \tau_0 \forall (e,t) \in E \times T$

4: **while**(time limit not reached) **do**

5:**for** (a=1 to m) **do** // m is the number of ants

$A_0 = \phi$

6: **for**(i=1 to $|E|$) **do**

7:Compute probability using Eq. (3.2.2) and maintain selective probabilities for time slots

8:for each $\boldsymbol{e_i}$ choose time slot $\boldsymbol{t}$ from selective probabilities

$\boldsymbol{A_i = A_{i-1} \cup \{(e_i, t)\}}$

9:Compute fitness value using Eq. (3.2.5) and save the best solution

10: **End for**

11: **End for**

12: Update global pheromone value

13: **End while**

3. EXPERIMENTAL STUDIES

In the experimental environment, both instructors 'flexibility, and students' flexibility is considered. Due to the complexity of developing such type of algorithm, configuring of the algorithms are strongly relied on an experimental methodology. The instructors' preferences are varied from 0 to 5. 0 means lowest preference and 5 means highest preference. Experiments with real-world input data coming from department of Computer Science and Engineering (CSE) of Khulna University of Engineering & Technology (KUET) have been conducted. In KUET, there are 5 days for teaching in a week and 9 teaching hours for each day. Two instructors conduct a theory course and there is a common class between them. Moreover, two instructors must conduct a laboratory course and the duration of each laboratory class is three consecutive time slots. The experiments have been done with real data which is even term data of CSE for course schedule. The simulation of the ACO and ACOSP on UCSP has been performed on a PC (Intel Core 2 Duo E7500 @ 2.93 GHz CPU, 2GB RAM, Win2007 OS). The algorithm is implemented in C++ programming language on Code Blocks 10.05.

3.1. Experiments on Simple Environment

In the simple environment, only theory courses are considered and they must be assigned among the first six periods for every instructor. Also, all the classes of a particular course will be conducted by only one instructor.

A comparison is made among GA, ACO based, and ACOSP method to solve UCSP for simple environment. The proposed ACO and ACOSP methods already been discussed. In the following, it will discuss about how the UCSP solution is encoded in the GA based method.

Every solution is considered with one vector for UCSP, with the length of the vector equal to the number of theory classes. The first row represents with triple {classes (E), instructors (I), batches (B) }; the second row represents the assigned timeslot (T) to the corresponding triple $\{E, I, B\}$ as shown in Figure 1.

$$\begin{pmatrix} \{E1, I1, B1\} & \{E2, I2, B3\} & \{E3, I1, B2\} & \ldots & \{EN-1, IN-1, B1\} & \{EN-1, IN, B3\} & \{EN, IN-2, B4\} & \{EN, IN-1, B1\} \\ T10 & T2 & T20 & \ldots & T1 & T30 & T40 & T28 \end{pmatrix}$$

Figure 1. Representation of the solution matrix.

The fitness of a solution is computed so that particular solution can be preferred than all other solutions. The fitness of solution is calculated using Eq. (5).

3.1.1. Input Data Preparation

Table 1 consists of which course related to which batch and also credit hour of each course. Table 2 consists of number of courses of each instructor and which instructor will conduct which course.

Table 1. Batch information for simple environment

Batch Name	Course No.	Credit	No. of assignment/week
1st Year 2nd Term	CSE 1201	3.0	3
	CSE 1207	3.0	3
	CHEM 1207	3.0	3
	EEE 1217	3.0	3
	MATH 1207	3.0	3
2nd Year 2nd Term	CSE 2201	3.0	3
	CSE 2207	3.0	3
	CSE 2213	3.0	3
	EEE 2217	3.0	3
	MATH 2207	3.0	3

Table 2. Instructor's information for simple environment

Instructor	No. of Course Taken	Course No.
T1	2	CSE 1207,CSE 2213
T2	2	CSE 2201,CSE 1201
T3	1	CSE 2207
T4	1	EEE 1217
T5	1	EEE 2217
T6	1	MATH 1207
T7	1	MATH 2207
T8	1	CHEM 1207

3.1.2. Experimental Analysis

A small data set is considered to conduct the experiment where 8 instructors, 10 courses, and 2 batches. The experiment is conducted varying population size and number of iterations. The individual fitness value is calculated using eq. (1) and UCSP fitness value is calculated using eq. (5). Actually, the UCSP fitness values are plotted against number of iteration and population. The experimental results are given after 10 trials. In Figure 2, 100 iterations are set for GA, ACO methods and 10% selective probability for ACOSP method. In this case the population size is varied. In Figure 3, 100 population is set for all three methods. In this case the numbers of iterations are varied.

It is seen from Figure 2 and Figure 3; the proposed two algorithms have better fitness value than that of GA. The main reason is GA assigns courses to the time slot without considering probabilities of time slots. Whereas both ACO and ACOSP calculate probabilities of time slot before selecting a particular time slot. Therefore, there is a chance to choose suitable time slots. On the other hand, ACOSP method provides better result than ACO method. The main reason is ACOSP maintains

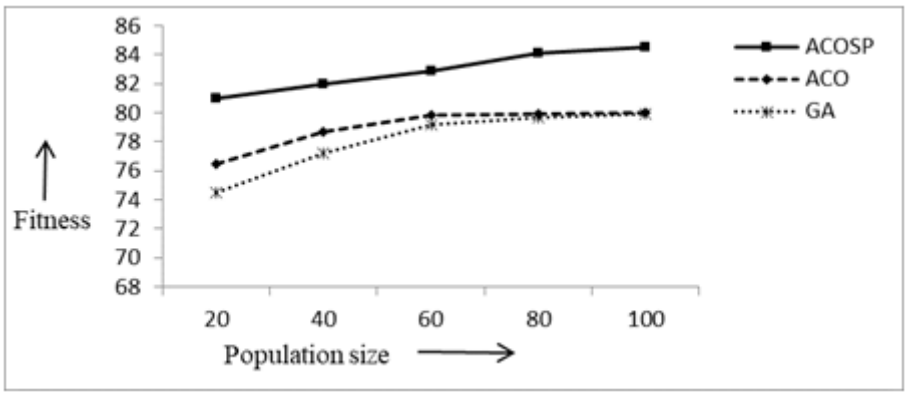

Figure 2. Comparison among GA, ACO, and ACOSP varying population size.

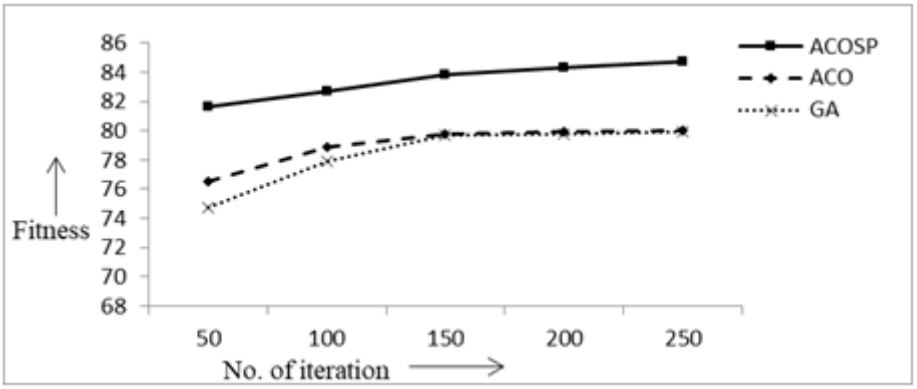

Figure 3. Comparison among GA, ACO, and ACOSP varying no. of iteration.

selective probabilities of unassigned suitable time slots to select a particular time slot. It is also seen that number of population (Figure 2) and iteration (Figure 3) increases the fitness value is increases. The main reason is that both exploration and exploitation is increases area of the search space which guides the solution towards better solution. The solution is converged at 100 population (Figure 2) and at 250 ant (Figure 3) and optimum solution is found.

3.2. Experiments on Highly Constrained Environment

In highly constrained environment, both theory and laboratory courses are considered. Two instructors conduct a class each and a common class for the theory course and the laboratory course must be conducted by two instructors using three consecutive time slots. In this environment, larger data set is considered than the simple environment. GA is excluded for highly constrained environment because it is quite difficult to encode a solution in such environment.

3.2.1. Input Data Preparation

For input data, 35 instructors, 37 courses, and 4 batches is considered. Table 3 consists of which course related to which batch and also credit hour of each course. Odd number of courses is treated as theory courses and even number of courses is treated as laboratory courses. Table 4 consists of number of courses of each instructor and which instructor will conduct which course.

Table 3. Batch information for highly constrained environment

Batch Name	Course No.	Credit	No. of assignment/week	Type of course
1st Year 2nd Term	CSE 1201	3.0	3	Theory
	CSE 1202	1.5	2	Laboratory
	CSE 1207	3.0	3	Theory
	CHEM 1207	3.0	3	Theory
	CHEM 1208	1.5	2	Laboratory
	EEE 1217	3.0	3	Theory
	EEE 1218	1.5	2	Laboratory
	HUM 1208	0.75	1	Laboratory
	MATH 1207	3.0	3	Theory
	ME 1270	0.75	1	Laboratory
2nd Year 2nd Term	CSE 2200	1.5	2	Laboratory
	CSE 2201	3.0	3	Theory
	CSE 2202	1.5	2	Laboratory
	CSE 2207	3.0	3	Theory
	CSE 2208	0.75	1	Laboratory
	CSE 2213	3.0	3	Theory
	EEE 2217	3.0	3	Theory
	EEE 2218	1.5	2	Laboratory
	MATH 2207	3.0	3	Theory
3rd Year 2nd Term	CSE 3201	3.0	3	Theory
	CSE 3202	1.5	2	Laboratory
	CSE 3203	3.0	3	Theory
	CSE 3204	0.75	1	Laboratory
	CSE 3207	3.0	3	Theory
	CSE 3211	3.0	3	Theory
	CSE 3212	0.75	1	Laboratory
	ECE 3215	3.0	3	Theory
4th Year 2nd Term	CSE 4105	3.0	3	Theory
	CSE 4106	0.75	1	Laboratory
	CSE 4107	3.0	3	Theory
	CSE 4108	0.75	1	Laboratory
	CSE 4109	3.0	3	Theory
	CSE 4110	0.75	1	Laboratory
	CSE 4113	3.0	3	Theory
	CSE 4120	0.75	1	Laboratory
	CSE 4123	3.0	3	Theory
	CSE 4124	0.75	1	Laboratory

Table 4. Instructor's information for highly constrained environment

Instructor	No. of Course Taken	Course No.
T1	2	CSE 1207,CSE 2213
T2	3	CSE 3211,CSE 3212,CSE 4120
T3	2	CSE 3201,CSE 3202
T4	3	CSE 3203,CSE 3204,CSE 4113
T5	5	CSE 2200,CSE 4105,CSE 4106,CSE 4123,CSE 4124
T6	2	CSE 2207,CSE 2208
T7	5	CSE 3207,CSE 4107,CSE 4108,CSE 4109,CSE 4110
T8	2	CSE 2207,CSE 2208
T9		CSE 2200,CSE 2201,CSE 2202
T10	4	CSE 3201,CSE 3202,CSE 4123,CSE 4124
T11	7	CSE 1201,CSE 1202,CSE 4107,CSE 4108,CSE 4109,CSE 4110, CSE 4120
T12	5	CSE 1207,CSE 2201,CSE 2202,CSE 3207,CSE 4113
T13	4	CSE 3203,CSE 3204,CSE4105,CSE 4106
T14	4	CSE 1201,CSE 1202,CSE 3211,CSE 3212
T15	1	CSE 2213
T16	1	EEE 1217
T17	1	EEE 1217
T18	2	EEE 2217,EEE 2218
T19	2	EEE 2217,EEE 2218
T20	1	EEE 1218
T21	1	EEE 1218
T22	1	MATH 1207
T23	1	MATH 1207
T24	1	MATH 2207
T25	1	MATH 2207
T26	1	ECE 3215
T27	1	ECE 3215
T28	1	ME 1270
T29	1	ME 1270
T30	1	CHEM 1207
T31	1	CHEM 1207
T32	1	CHEM 1208
T33	1	CHEM 1208
T34	1	HUM 1208
T35	1	HUM 1208

3.2.2. Experimental Analysis

For highly constrained environment, r data consisting of 35 instructors, 37 courses, and 4 batches is considered to conduct the experiment. Experiment is conducted by varying number of iterations and number of ants. The individual fitness value is calculated using eq. (1) and UCSP fitness value is calculated using eq. (5). Actually, the UCSP fitness values are plotted against number of iteration and population or ant. The experimental results are given after 10 trials. In Figure 4, 300 ants are set for both the methods. In this case the numbers of iterations are varied. In Figure 5, 200 iterations are set for ACO method and 200 iterations with 50% selective population for ACOSP method. In this case the numbers of ants are varied.

Figure 4 and Figure 5 represents the comparison of proposed ACO and ACOSP based methods for the same size of data varying number of iterations and number of ants respectively. From the above experimental data, it is seen that with the same volume of data and varying number of iterations and number of ants, the average fitness of solution is significantly improved in the ACOSP method rather than ACO method (Figure 4 and Figure 5 where bold line shows fitness of ACOSP method and dotted line shows the fitness of ACO method). Because, incorporating selective probabilities for unassigned suitable time slots with ACO has more pheromone value and more heuristic information for a particular time slots.

This pheromone value and heuristic information value is directly involved to calculate the probability of time slots. Thus, ACOSP considers some better of probabilities of time slots

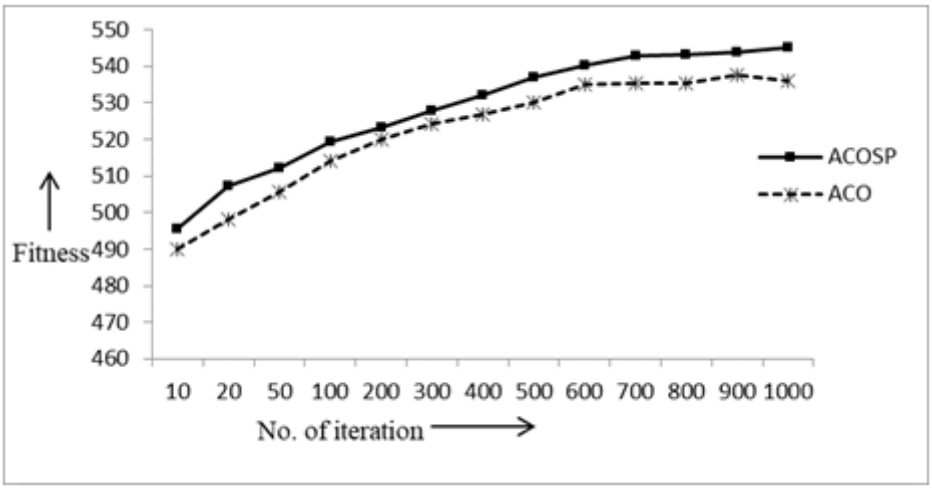

Figure 4. Comparison of ACO with ACOSP varying no. of iteration.

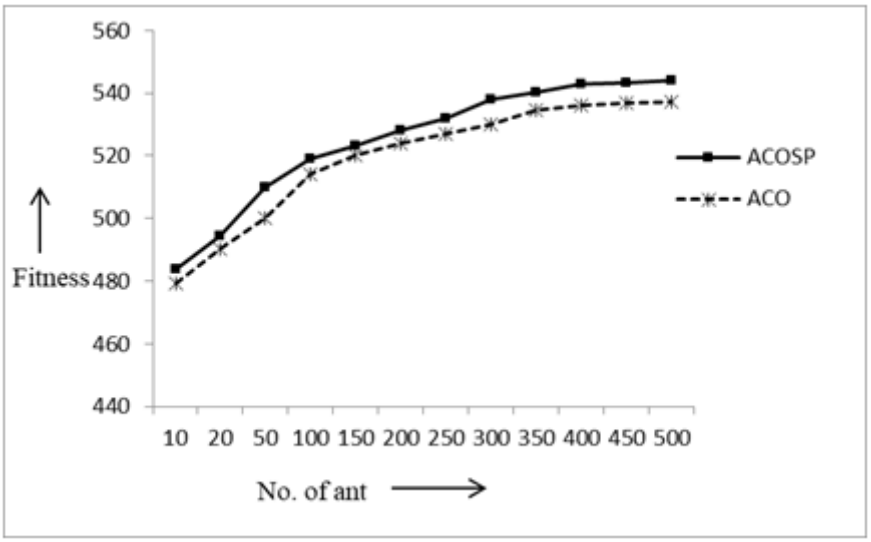

Figure 5. Comparison of ACO with ACOSP varying no. of ants.

rather than all the time slots. It indicates that better probabilities of time slots have more chance to be chosen. On the other hand, ACO algorithm selects time slots probabilistically for assigning the courses among all the unassigned time slots. Moreover, it is seen from the figures (Figure 4

and Figure 5), if the number of iterations and number of ants are increased the fitness of both algorithms are also increased. The main reason is that more number of iterations and more number of ants are able to find better solution.

From the experimental result it is seen that the ACOSP algorithm performs better than ACO algorithm i.e., better quality of UCSP solution as it operates with suitable time slots to select a particular time slot. On the other hand, ACO operates with all the unassigned time slots. Therefore, less chance the pheromone will be deposited to suitable time slots resulting better path of pheromone trail will be disappeared. Consequently, decrease the quality of the UCSP solution. It is also seen that number of iteration (Figure 4) and population or ant (Figure 5) increases the fitness value is increases. The main reason is that both exploration and exploitation is increases area of the search space which guides the solution towards better solution. Solution is converged at 1000 iteration (Figure 4) and at 500 ant (Figure 5) and optimum solution is found.

4. CONCLUSION

Solving Constrained Satisfaction Problem (CSP) has been an active research area for several decades. University Class Scheduling Problem (UCSP) is one of the most popular CSP and interest grows in last few years to solve it novel ways. Recently, Swarm Intelligence (SI) based methods had drawn great attraction to solve UCSP. In this paper an Ant Colony Optimization (ACO) and ACO with selective probability (ACOSP) methods are used for solving UCSPs. The experiment has been conducted for two different environments: simple and highly constrained. Varying number of ants/population size and number of iterations, at first a comparison is made for proposed ACO and ACOSP based methods with GA for simple environment. The experimental result shown that ACOSP based method performs better than GA and ACO based method. Then ACO and ACOSP based methods are applied for the highly constrained environment. From the experimental outcome it is shown that ACOSP based method is better than ACO based method. The main contribution is ACO algorithm is formulated and modified to solve course scheduling problem of KUET. The proposed algorithms are limited to applicable for particular domain of problem. In the future, the performance could be improved using more robust SI based algorithms. It is possible to solve similar domain or topic of problem with such algorithms.

REFERENCES

[1] Tomáš Müller, “Constraint-based Timetabling”, (2005), Ph.D. Thesis, Charles University, Prague.

[2] Tassopoulos Ioannis X., Beligiannis Grigorios N., “Solving effectively the school timetabling problem using particle swarm optimization”, (2012), Expert Systems with Applications, 39 (5), p.6029-6040.

[3] TassopoulosIoannis X., Beligiannis Grigorios N.,“A hybrid particle swarm optimization based algorithm for high school timetabling problems”, (2012), Applied Soft Computing, vol. 12 no. 11, pp. 3472-3489.

[4] D. F. Shiau, “A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences”, (2011), Expert Systems with Applications, vol. 38, pp. 235-248.

[5] R. A. Valdes, E. Crespo, J. M. Tamarit, “Design and implementation of a course scheduling system using tabu search”, (2002), European Journal of Operational Research,vol.137, no.3, pp.512–523.

[6] Ernesto Ferrer, Queiros Nunez, “A Hybrid Genetic Algorithm for the Student-Aware University Course Timetabling Problem”, (2010), Bachelor Thesis, Macalester College.

[7] Mohammad-Reza Feizi-Derakhshi, Hamed Babaei, Javad Heidarzadeh, "A Survey of Approaches for University Course Timetabling Problem", (2012), International Symposium on Intelligent and Manufacturing Systems (IMS), 307-321.

[8] Chohan, Ossam, "University scheduling using Genetic Algorithm", (2009), Master's Thesis, Dlarna University.

[9] Cagdas Hakan Aladag, Gulsum Hocaoglu, Murat Alper Basaran, "The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem", (2009), Expert Systems with Applications, 36 (10), p.12349-12356.

[10] ZhipengLü, Jin-Kao Hao, "Adaptive Tabu Search for course timetabling", (2010), European Journal of Operational Research, 200 (1), p.235-244.

[11] Nasser R. Sabar, MasriAyob, Graham Kendall, Rong Qu, "A honey-bee mating optimization algorithm for educational timetabling problems", (2012), European Journal of Operational Research, 216 (3), p.533-543.

[12] S Daskalaki, T Birbas, E Housos, "An integer programming formulation for a case study in university timetabling", (2004), European Journal of Operational Research, 153 (1), p.117-135.

[13] Jin-Kao Hao, Una Benlic, "Lower bounds for the ITC-2007 curriculum-based course timetabling problem", (2011), European Journal of Operational Research, 212 (3), p.464-472.

[14] Ruey-Maw Chen, Hsiao-Fang Shih, "Solving University Course Timetabling Problems Using Constriction Particle Swarm Optimization with Local Search", (2013), Algorithms 2013, 6, 227-244.