ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Timetable Generation Using Ant Colony Optimization Algorithm

Priyanka Gore[1], Poonam Sonawane[1], Suneha Potdar[2]

B.E Student, Dept. of Computer, Dilkap Collage, Mumbai University, Neral, Maharashtra, India[1]

Guide, Professor, Dept. of Computer, Dilkap Collage, Mumbai University, Neral, Maharashtra, India[2]

ABSTRACT: Scheduling is one of the important tasks encountered in real life situations. Education timetable scheduling problem is known to be NP hard. Hence, evolutionary techniques have been used to solve the timetable scheduling problem. The ant colony optimization metaheuristic algorithm has already proved that it can be used in simplified artificial instances of College course timetabling problems. Till now limited work has been done applying it to practical timetabling scheduling problems. In this paper, we will be focusing on the application of the ant colony optimization to a highly constrained real–world instance of the College course timetabling problem. And we present the design of the memory–efficient of the construction graph and the sophisticated solution construction procedures. The implementation of mechanism here has been successfully used for timetabling at the consistent and organized pattern of behavior or activities.[2]

KEYWORDS*:* ACO, meta-heuristic algorithm, scheduling problem, timetable, Pheromone Update, Construction graph.

I. INTRODUCTION

In day to day life and also most probably people work related to organizations has occasionally faced some form of timetabling tasks. The college Course Timetabling Problem (CCTP) and the variations are parts of the larger class of timetabling and scheduling problems. A large number of college timetabling problems have been described in the literature, and they differ from each other based on the type of institution involved, the entities being scheduled and the constraints in the definition of the problem.[2]

Due to inherent complexations and variations of the problem, most real–world timetabling problems are NP–complete [1]. So we are making use of heuristic which can usually generate solutions that are good enough for practical use (although they do not guarantee an optimal solution). And if implemented then because of their manageability and good performance, metaheuristic techniques have shown to be particularly suitable for solving these kinds of problems.

In our project[1], we focus on the *college Timetabling Problem* (CTP), framed as an example of the university course timetabling problem. In our project we have used two different metaheuristics for timetable construction i.e. *genetic algorithm* [2] and *ant colony optimization metaheuristic* [6]. Constructing a system for solving instances of CTP is challenging both the ways i.e. technically and administratively.

There have been a number of approaches made in the past decades to the problem of constructing timetables for colleges and schools. Timetabling problems may be solved by di erent methods inherited from operations research such as graph coloring and mathematical programming, from local search procedures such as tabu search and simulated annealing, from genetic algorithms or from backtracking-based constraint satisfaction manipulation In our project, timetabling problem is formulated as a constraint satisfaction problem and we proposed a practical timetabling algorithm which is capable of taking care of both strong and weak constraints and finding variables instantiation, which is based on the forward search method.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

II. LITERATURE SURVEY

The purpose of this research is to use ant colony optimization (ACO) to develop a heuristic algorithm to solve the University timetabling problem. The traveling salesman problem (TSP) is one of the most important combinatorial problems of ACO. This algorithm takes into consideration the timetable scheduling satisfying constraints that avoid clash of faculty, class room slots, etc. The automatic course scheduling system proposed to produce course timetables that truly fulfill user's needs and increase teachers' satisfaction. It is effective method for producing high quality solutions to the college course timetabling problem [7].As mentioned above, many types of timetabling problems exist. But all these problems have several properties in common. One of these similarities is that certain entities have to be scheduled. e.g., the college timetabling problem has several entities such as students, lecturers, courses, practical and classes and labs. All these entities have properties e.g. classes are linked to the course and the students of this class are taught. Constraints Assignments usually cannot be done arbitrarily, but many constraints have to be considered. We distinguish two di□erent types, namely hard and soft constraints. A solution is feasible if no hard constraints are violated. A feasible solution is better than another if fewer soft constraints are violated. A timetabling algorithm can use di□erent strategies to get a solution without violations of hard constraints. Violations can either be avoided from the outset or penalized to lead the algorithm towards better solutions and introduce repair mechanisms.
A timetable construction is an NP-complete scheduling problem. It is not a standard job-shop problem because of the additional classroom allocation. It is large and highly constrained, but above all the problem di□ers greatly for di□erent colleges and educational institutions. It is di□cult to write a universal program, suitable for all imaginable timetabling problems. Although manual construction of timetables is time-consuming, it is still widespread, because of the lack of appropriate computer programs.

III. PROBLEM STATEMENT

The timetable construction problem is a combinatorial optimization problem that consists of four finite sets: (*i*) a set of meetings, (*ii*) a set of available resources (e.g. rooms, staff, and students), (*iii*) a set of available time–slots, and (*iv*) a set of constraints. The problem is to assign resources and time slots to each given meeting, while maintaining constraints satisfied to the highest possible extent. The University Course Timetabling Problem (UCTP) is a timetabling problem where the given data consists of a set of students and courses that each of the students needs to attend. A *course* is a set of events that need to take place in the timetable. The main characteristic that distinguishes the university course timetabling problem from other types of timetabling problems is the fact that students are generally allowed to choose the courses in which they wish to enrol [9].
The above description of the UCTP defines a broad range of problems, whose complexity significantly depends on the specific constraints defined. Particular timetabling applications are usually focused on a more strictly defined subset of the problems, as the constraints and dimensions of the problem vary among institutions. We use the same approach, giving a detailed formal description of the problems for which our application is designed.[2]

A. Definition of timetabling problem

The *Timetabling problem* is defined as a six–tuple:

$$TP = (T,L,R,E,S,C) ,$$

where*T* is a set of *time–quanta* of scheduling, *L* is a set of *limited assets* at the university, *R* is a set of *rooms*, *E* is a set of *events* that need to be scheduled, *S* is a set of attending *students*, and *C* is a set of *constraints*. Assuming durations of all the events can be quantified as multiples of a fixed value of time that we call a *time–quantum*. A *time–slot* is defined as one or more consecutive time–quanta in the timetable [8]. The duration of the quantum reflects a trade–off between the precision of scheduling and the size of the search space.The set of limited assets (resources) shared among the different exercises is denoted *L*. For each resource $l \in L$, a fixed number of workplaces can use the resource concurrently.
Each room is defined as a set of *workplaces*, atomic room resources varying from room to room, such as seats in ordinary classrooms, computers in computer classrooms, etc. For each room $r \in R$, the number of workplaces, denoted $size_r \in \mathbb{N}$ is defined. For each of the events, the desired number of students per workplace is defined. Since some rooms may not be available all the time, a set of time quanta $T_r \subseteq T$ in which the room is available is defined for each roomSolving TP Using Ant Colony Optimisation.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

b. Construction Graph

The main issue in applying ACO to a problem is to find an appropriate representation that can be used by the artificial ants to build solutions [3]. This representation is called the *construction graph*. To ensure that the problem representation is suitable for large instances of TP, memory–efficiency is the main design goal. The construction graph we devised can be seen in figure 1. Semantically, each of the nodes represents one of the following: a student, an event or a *dock node*.

An edge connecting a dock node and an event node means that the event can be scheduled in that time and place. This means that the dock represents the room and time that are suitable for that event. Dock nodes are connected to student nodes as well. Student nodes are only connected to docks under the following conditions: (*i*) the dock is connected to at least one of the events the student needs to attend and (*ii*) the student is free from pre–assignments in time–quanta represented by the dock and the dur_{es}consecutive time quanta. The event $e_s$is the shortest event enrolled by the student that can be held in the aforementioned dock, and its duration is denoted dur_{es}. To each of the graph's edges, a pheromone concentration value $\tau_{ij}$and a heuristic information value $\eta_{ij}$are assigned. The TP solution is a timetable with all of the events and students scheduled into the appropriate docks. A candidate solution (timetable) is represented as a subset of edges of the construction graph, connecting the timetable building blocks into a specific timetable.

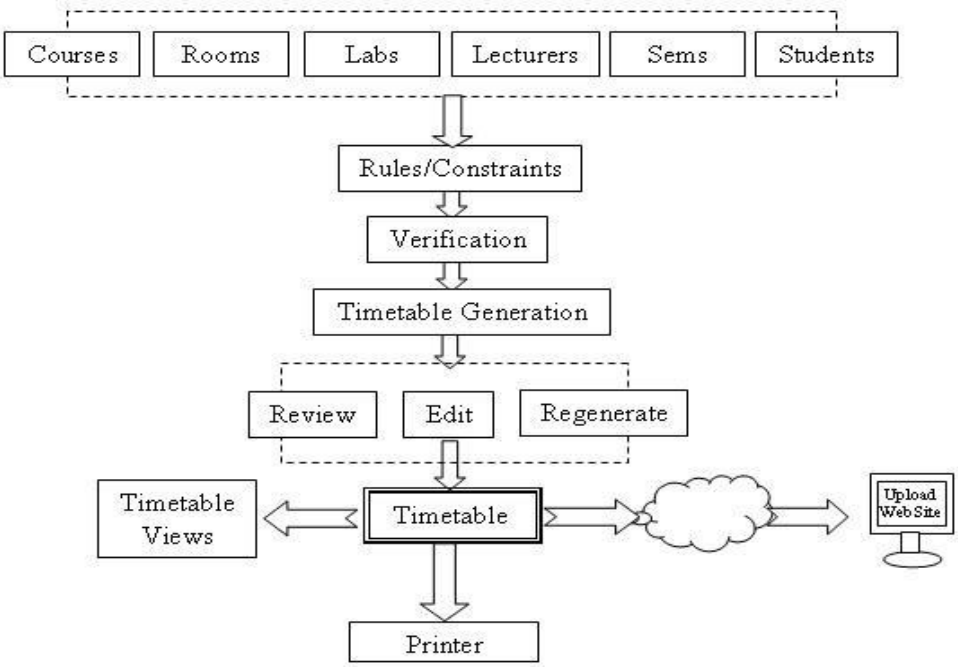


Fig. General View of TTGS

c. Algorithm Description

Our approach uses a MAX − MIN Ant System [3], since such systems have shown great promise on various different problems, including artificially generated timetabling problems [5]. A colony of *m* ants is used. At each algorithm iteration, each ant constructs a complete timetable (a candidate solution). In each of the generated solutions, all of the hard constraints are satisfied.

In choosing solution components, the probability p_{ij}^k that the ant *k*, currently at node *i*will choose the node *j* is calculated using the *random proportional rule* given by

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, \qquad j \in \mathcal{N}_i^k, \qquad (1)$$

where $\tau_{ij}$is the pheromone trail value on the edge connecting node *i*to node *j*, $\eta_{ij}$is the heuristic value of that edge, α and β are the parameters which determine the relative influence of pheromone trail and the heuristic information, and N_i^k is the feasible neighbourhood of ant *k* when it is at node *i*.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

The pheromone trail between dock node *i*and event/student node *j* marks the desirability of scheduling that event/student in that dock. In each algorithm iteration only one of the ants updates the pheromone trails based on the quality of the constructed solution candidate. The heuristic value $\eta_{ij}$is controlled by the constraint fence for each ant. It is important to note that the heuristic value is dynamically modified throughout the algorithm execution [4]. It is set to zero when it is determined that the constraints of the problem would be violated if the edge (*i*,*j*) were included in the the tour, and to one otherwise.

d. Pheromone Update

The pheromone trail value is updated at the end of each algorithm iteration. The pheromone is updated by the best–so–far ant or the iteration–best ant. The probability that the best–so–far ant is allowed to update the pheromone trail is 5%. Unlike usual ACO approaches, the pheromone gain value in our approach is not the same for all of the edges of a single tour. Usually, more than one event can be scheduled in a single dock. Nevertheless, after event *e* is scheduled in dock *d*, no other event can be scheduled into *d*. Therefore, the quality of a partial schedule of a single event cannot be measured without considering its influence on other events. For example, suppose that event e_1 can be scheduled in the set of docks $\{d_0,d_1,d_2\}$ and event e_2 can be scheduled only in dock d_1. Dock d_1 may seem to be appropriate for event e_1 since it would leave zero students that need to attend e_1 unscheduled. However, this is a very poor choice considering that d_1 is the only suitable dock for e_2, so assigning it to e_1 would leave all of the students who need to attend e_2 unscheduled. The level of influence between two events is modelled by the influence function $f : E \times E \rightarrow [0,1]$. When $f_{a,b}= 0$, event *a*has no influence on event *b*, while $f_{a,b}= 1$ means that event *a* is scheduled in the entire set of docks suitable for the event *b*. More formally, the influence of event *a* on *b*, denoted $f_{a,b}$, is defined as:

$$f_{a,b} = \frac{|chosenD_a \cap D_b|}{|D_b|} \ ,$$

where$chosenD_a \subseteq D_a$ is a subset of docks in which event *a* is scheduled in a given suggested solution, and $D_b \subseteq D$ is the set of docks suitable for the event *b*.We use the number of unscheduled *obligations* as the optimised variable. An obligation is defined as an assignment of a given student to one of the laboratory exercises he or she needs to attend. In our problem representation, this is done by assigning students to the dock nodes during the second phase of the construction procedure for a single event, as described in Section 3.2. The solution quality function $Q_e$for the event *e* and the solution suggestion *Sug*is defined as:

$$Q_e = \frac{assignedObligations(e, Sug)}{numberOfObligations(e)} \cdot spaceEfficiency(e, Sug)^3 \ ,$$

$$spaceEfficiency(e, Sug) = \left[\frac{assignedObligations(e, Sug)}{reservedSeats(e, Sug)}\right] \ .$$

The space efficiency factor ensures that better solutions use the dock space more efficiently. The pheromone gain for each event $\Delta\tau_e$is given by

$$\Delta\tau_e = \left(\frac{\sum_{e_i \in E} (f_{e,e_i} \cdot Q_{e_i})}{|E|}\right)^4 , \qquad e \in E \ .$$

The pheromone gain $\Delta\tau_e$is deposited on the edges of the tour connecting the event *e* to the dock *d* and on the edges connecting a student *s* to any of the docks in which the event is scheduled.

e. MAX−MIN Ant System Parameters

Several configurations of the metaheuristic were evaluated, and the best results were achieved using the settings in Table 1. The pheromone values are initially set to τ_{max}. The pheromone trails are updated after each algorithm iteration, as described in Section 3.3, and evaporation is used for each edge, according to the rule $\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij}$.

Our system uses either 10000 iterations or *penalty* = 0 as the stop criteria. To prevent stagnation, if the best solution found is not improved after 125 consecutive iterations, the pheromone trails are reset by setting pheromone value on each edge back to τ_{max}.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Table 1. Ant colony optimisation algorithm parameters

Parameter	Value	Parameter	Value
number of ants (m)	5	ρ	0.02
α	1.0	$\tau_{max} = 1/\rho$	50
β	1.0	τ_{min}	0.5

III. RESULTS

The system described here was successfully applied to the laboratory exercises timetabling problem at the authors' institution. The performance of the algorithm on several datasets is presented in Table 2. These problem instances have different durations and widely varying numbers of events and attending students. Two values, $S_{e,s}$and $Ts$are given as measures of the problem instance complexity. The *student event sum*, denoted $S_{e,s}$, is defined as the aggregate number of events that each of the students needs to attend. More formally,
$S_{s,e}= {}^{P}{}_{s\in S}|E_s|$, where $E_s$is the set of events that student s needs to attend. The *timespan* of the problem instance, denoted Ts, is defined as the number of days on which the events need to be scheduled. To illustrate the effectiveness of our approach, the results are compared with a GRASP technique (Table 2). The tested GRASP technique uses a construction search (different from the search procedure defined for our ants) to build solutions satisfying hard constraints, after which a local search that optimises the schedule of students is performed. We used the Mann–Whitney test to check the H0 hypothesis that the distribution functions of the algorithm performances were the same for the results of both the ACO and GRASP techniques. For each of the datasets, the p values were well below 0.05. On each problem instance, with very high statistical significance, we conclude that the ACO technique performs better than the GRASP technique.Note that although the Genetic algorithm has also been applied to the TP problem as a part of a sister project at our institution [2].
In some problem instances, a solution where all students are scheduled could not be found. This was usually caused by a constellation of conflicting events, or events with infeasible requirements posed by the course organisers. In such instances, the system is used as a tool for identifying the problematic events. In practice, the process of scheduling is usually an iterative process of querying the system for the best results, interpreting those results, and allowing the staff to make informed decisions.

Table 2. Algorithm performance on different *laboratory exercise timetabling problem* instances.

			ACO penalty			median comparison	
instance	$S_{s,e}$	Ts	min	max	st.dev. (σ)	ACO	GRASP
C1	2104	5	66	150	23.31	102	330
C2	7081	9	1	14	3.46	8	71
C4	4868	5	5	73	12.42	13	125
C8	5430	4	34	146	31.00	58	190
C12	5934	5	32	78	9.31	41	333

IV. CONCLUSION

This paper presents a case study of applying ACO metaheuristic for solving a complex large–scale timetabling problem at our institution. Our solution uses a relatively general problem representation, suitable for different types of institutions. We present an innovative, memory–efficient problem representation that is appropriate for large problem instances. Moreover, the modular design of the constraint library facilitates the addition of new constraints. It makes the system manageable, which is extremely important for practical timetabling applications. The exact problem we are solving is formulated as the laboratory exercise timetabling problem, a subset of the university course timetabling problem.[2]
This work arose out of the specific timetabling needs of one institution. However, the approach described here is not limited to TP, since it shares many commonalities with other UCTP instances. Thus, it is likely that the challenges we

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

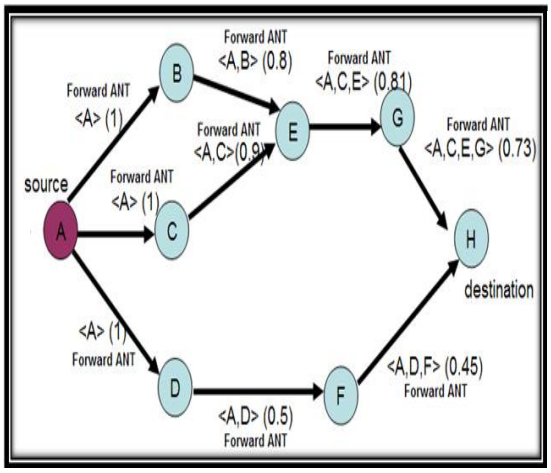
(An ISO 3297: 2007 Certified Organization)

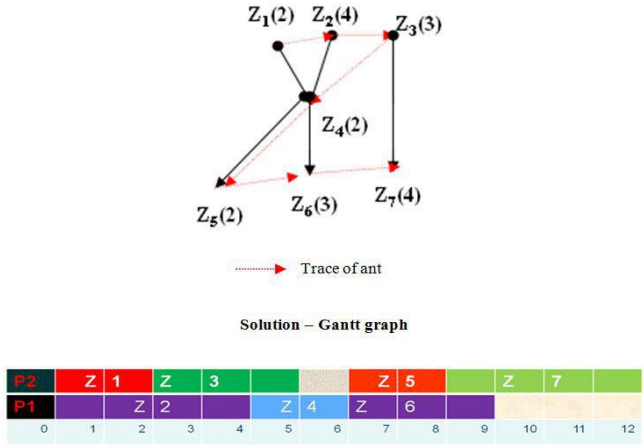
Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

faced, such as reducing the memory footprint of the construction graph or ensuring the ease of adaptation to problem modifications, will also be faced by other researchers. Other authors may use our approach without modifying the problem representation. The only necessary adaptation may be the implementation of additional constraints that are not supported by our current library.

Furthermore, since prior work on ant colony optimisation mainly considered artificially generated UCTP instances, our work proves that ACO can be highly successful in solving real–world timetabling problems. It is an effort to help bridge the gap between theoretical research and practical adaptation of metaheuristic techniques that is currently so prevalent in the area of automated timetabling. Our work can also be viewed as an additional confirmation that ACO is not only an interesting academic research topic, but also a manageable and efficient approach able to solve highly complex real–world problems.





V. ACKNOWLEDGEMENT

This project has been a great learning experience for us. However, it would not have been possible without the kind support and help of many individuals and faculty members. I would like to extend my sincere thanks to all of them.

We are highly indebted to **Prof.YatinBagul** for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

REFERENCES

1. Cooper, T.B., Kingston, J.H.: The complexity of timetable construction problems. In: Proceedings of the First International Conference on the Practice and Theory of Automated Timetabling (ICPTAT '95). (1995) 511–522
2. Bratkovi´c, Z., Herman, T., Omrˇcen, V., Cupi´c,ˇ M., Jakobovi´c, D.: University course timetabling with genetic algorithm: A laboratory excercises case study. In Cotta, C., Cowling, P., eds.: Proceedings of EvoCOP 2009 - 9th European Conference Evolutionary Computation in Combinatorial Optimization. Volume 5482 of Lecture Notes in Computer Science., Springer, Heidelberg (2009) 240–251
3. Dorigo, M., Stutzle, T.: Ant Colony Optimization. Bradford Books. The MIT Press (july 2004)
4. Socha, K., Sampels, M., Manfrin, M.: Ant Algorithms for the University Course Timetabling Problem with Regard to the State-of-the-Art. In: Proceedings of EvoCOP 2003 – 3rd European Workshop on Evolutionary Computation in Combinatorial Optimization. Volume 2611 of Lecture Notes in Computer Science., Springer
5. Verlag, Berlin, Germany (2003) 334–345
6. Socha, K., Knowles, J., Sampels, M.: A MAX-MIN Ant System for the University Timetabling Problem. In Dorigo, M., G. Di Caro, Sampels, M., eds.: Proceedings of ANTS 2002 – From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms. Volume 2463 of Lecture Notes in Computer Science., Springer Verlag, Berlin, Germany (September 2002) 1 – 13
7. Rossi-Doria, O., Sample, M., Birattari, M., Chiarandini, M., Dorigo, M., Gambardella, L., Knowles, J., Manfrin, M., Mastrolilli, M., Paechter, B., Paquete, L., Stu¨tzle, T.: A Comparison of the Performance of DifferentMetaheuristics on the Timetabling Problem. In: The Practice and Theory of Automated Timetabling IV: Revised Selected Papers from the 4th International conference, Gent 2002. Volume 2740 of Springer Lecture Notes in Computer Science., Springer, Berlin, Germany (2003) 329–351
8. Azimi, Z.: Comparison of Methheuristic Algorithms for Examination Timetabling Problem. Applied Mathematics and Computation **16**(1) (2004) 337–354
9. McCollum, B.: University timetabling: Bridging the gap between research and practice. In E Burke, H.R., ed.: PATAT 2006—Proceedings of the 6th international conference on the Practice And Theory of Automated Timetabling, Masaryk University (2006) 15 – 35
10. Gross, J.L., Yellen, J.: A Handbook of Graph Theory. Discrete Mathematics and Its Applications. CRC books (December 2003)

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

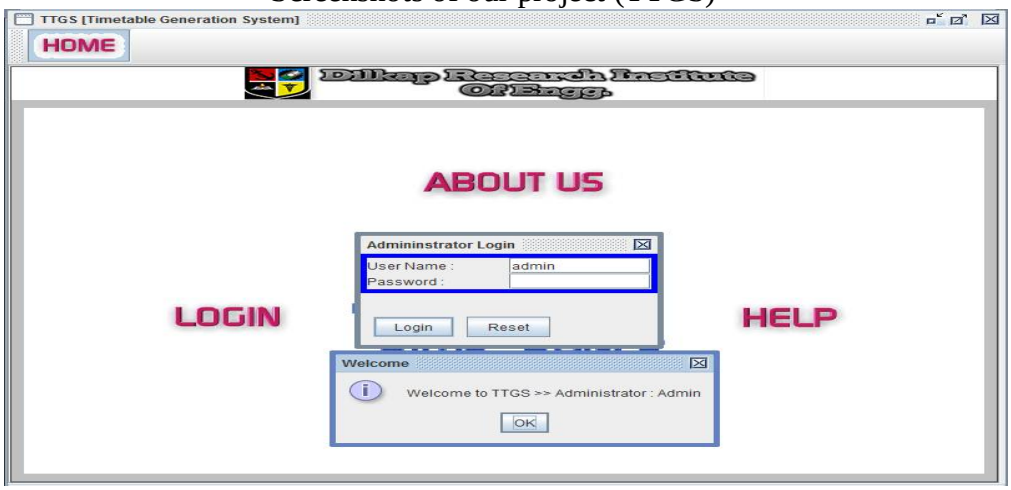
International Journal of Innovative Research in Computer and Communication Engineering

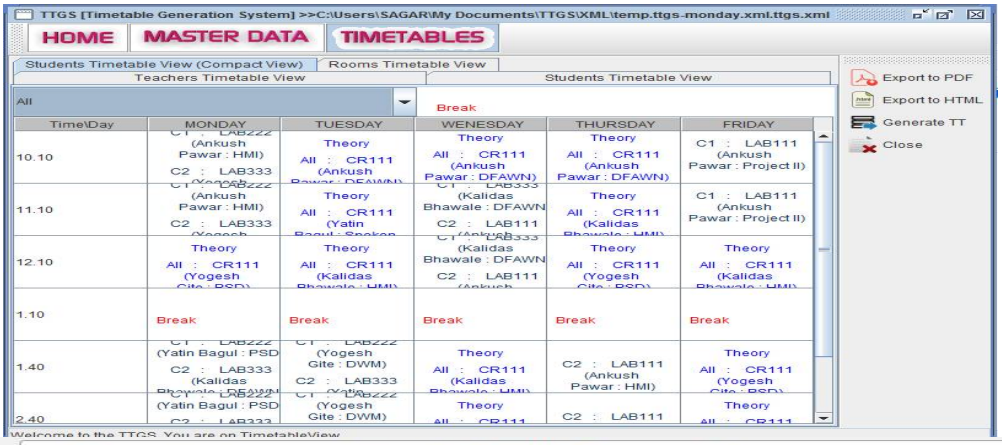
(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Screenshots of our project (TTGS)





AllCompact Timetable

Dilkap College OF Engineering
Computer Engineering Department

Time/Day	MONDAY	TUESDAY	WENESDAY	THURSDAY	FRIDAY
10.10	Practical C1 : LAB222 (Ankush Pawar : HMI) C2 : LAB333 (Yogesh Gite : DWM) C3 : LAB111 (Kalidas Bhawale : DFAWN)	Theory All : CR111 (Ankush Pawar : DFAWN)	Theory All : CR111 (Ankush Pawar : DFAWN)	Theory All : CR111 (Ankush Pawar : DFAWN)	Practical C1 : LAB111 (Ankush Pawar : Project II)
11.10	Practical C1 : LAB222 (Ankush Pawar : HMI) C2 : LAB333 (Yogesh Gite : DWM) C3 : LAB111 (Kalidas Bhawale : DFAWN)	Theory All : CR111 (Yatin Bagul : Spoken English)	Practical C1 : LAB333 (Kalidas Bhawale : DFAWN) C2 : LAB111 (Ankush Pawar : Project II) C3 : LAB222 (Yatin Bagul : PSD)	Theory All : CR111 (Kalidas Bhawale : HMI)	Practical C1 : LAB111 (Ankush Pawar : Project II)
12.10	Theory All : CR111 (Yogesh Gite : PSD)	Theory All : CR111 (Kalidas Bhawale : HMI)	Practical C1 : LAB333 (Kalidas Bhawale : DFAWN) C2 : LAB111 (Ankush Pawar : Project II) C3 : LAB222 (Yatin Bagul : PSD)	Theory All : CR111 (Yogesh Gite : PSD)	Theory All : CR111 (Kalidas Bhawale : HMI)
1.10	Break	Break	Break	Break	Break
1.40	Practical C1 : LAB222 (Yatin Bagul : PSD) C2 : LAB333 (Kalidas Bhawale : DFAWN) C3 : LAB111 (Ankush Pawar : Project II)	Practical C1 : LAB222 (Yogesh Gite : DWM) C2 : LAB333 (Yatin Bagul : PSD) C3 : LAB111 (Ankush Pawar : HMI)	Theory All : CR111 (Kalidas Bhawale : HMI)	Practical C2 : LAB111 (Ankush Pawar : HMI) C3 : LAB222 (Yogesh Gite : DWM)	Theory All : CR111 (Yogesh Gite : PSD)
2.40	Practical C1 : LAB222 (Yatin Bagul : PSD) C2 : LAB333 (Kalidas Bhawale : DFAWN) C3 : LAB111 (Ankush Pawar : Project II)	Practical C1 : LAB222 (Yogesh Gite : DWM) C2 : LAB333 (Yatin Bagul : PSD) C3 : LAB111 (Ankush Pawar : HMI)	Theory All : CR111 (Yogesh Gite : PSD)	Practical C2 : LAB111 (Ankush Pawar : HMI) C3 : LAB222 (Yogesh Gite : DWM)	Theory All : CR111 (Yatin Bagul : DWM)
3.40	Theory All : CR111 (Yatin Bagul : DWM)	Theory All : CR111 (Yatin Bagul : DWM)	Theory All : CR111 (Yatin Bagul : Spoken English)	Theory All : CR111 (Yatin Bagul : DWM)	Theory All : CR111 (Ankush Pawar : DFAWN)