International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:6, No:8, 2012

Transformation of Course Timetablinng Problem to RCPSP

M. Ahmad, M. Gourgand, C. Caux

***Abstract*—**The Resource-Constrained Project Scheduling Problem (RCPSP) is concerned with single-item or small batch production where limited resources have to be allocated to dependent activities over time. Over the past few decades, a lot of work has been made with the use of optimal solution procedures for this basic problem type and its extensions. Brucker and Knust[1] discuss, how timetabling problems can be modeled as a RCPSP. Authors discuss high school timetabling and university course timetabling problem as an example. We have formulated two mathematical formulations of course timetabling problem in a new way which are the prototype of single-mode RCPSP. Our focus is to show, how course timetabling problem can be transformed into RCPSP. We solve this transformation model with genetic algorithm.

***Keywords*—**Course Timetabling, Integer programming, Combinatorial optimizations

I. Introduction

TIMETABLING is a complex combinatorial problem. Timetabling can be generally defined as the activity of assigning, subject to constraints, a number of events to a limited number of time periods and locations such that desirable objectives are satisfied as nearly as possible [2]. Practical cases where such activity arises are, among others, educational timetabling, employee timetabling, sport timetabling, transport timetabling and communication timetabling. Usually, the solution of such problems is decomposed into two stages. In stage one, the assignment is performed while in the second stage scheduling decisions are taken. For railway and airline crew scheduling, in a first stage, scheduling problem is solved by fixing the timetable and the routing of trains or airplanes. In a second stage tasks are derived and the tasks are assigned to crew members. For university and school course teacher timetabling, it is decomposed in reverse order. In stage one, courses are assigned to teachers and in second phase schedule for courses is produced.

A. Educational Timetabling

Educational timetabling can be sub-divided into three main classes, which are school timetabling, course timetabling and exam timetabling [3]. In university course timetabling, a set of lectures must be scheduled into rooms and timeslots subject to constraints that are usually divided in two categories, which are hard and soft constraints.

M. Ahmad, Limos ; Université Blaise Pascal, Campus des Cézeaux-BP 10125, 63173 Aubière, France (phone: 33-473-407443; e-mail: maqsood@ isima.fr).

M. Gourgand, Limos ; Université Blaise Pascal, Campus des Cézeaux-BP 10125, 63173 Aubière, France (e-mail: michel.Gourgand@ isima.fr).

C. Caux, Limos ; Université Blaise Pascal, Campus des Cézeaux-BP 10125, 63173 Aubière, France (e-mail: christophe.caux@ ifma.fr).

Hard constraints must be strictly satisfied with no violation allowed, while in the case of soft constraints it is desirable, but not essential, to minimize violations. Typical of course timetabling, with respect to school and exam timetabling, are the availability of a limited number of timeslots and the requirements , allocating lectures only into suitable rooms, having no more than one lecture per room and scheduling lectures with common students in different timeslots. The timetabling problem, like many others in the area of combinatorial optimization, has been approached by several well-known techniques of the operational research and the computer science fields. Several surveys on course timetabling [4], [5], as well as others on more focused aspects of the problem, have managed to record this work in a systematic way, categorizing thus the different variations of the problem and solution approaches.

B. Basic Single-Mode RCPSP and course Timetabling

Our purpose is to transform educational timetabling problem (course timetabling) to resource constrained project scheduling problem. There are major six different classes of RCPSP: 1. Basic Single-Mode RCPSP 2. Basic Multi-Mode RCPSP 3. RCPSP problems with non regular objective functions 4. Stochastic RCPSP 5. Bin-packing-related RCPSP problems 6. Multi-resource-constrained project scheduling problems. But here we shall only discuss and use single mode RCPSP because in timetabling problem, courses are assigned to teachers in first stage and total duration for each course is also predefined. So by using single mode RCPSP, scheduling of these courses could be done. In our formulations we use set of lessons instead of set of courses to transform problem in RCPSP for that purpose we decompose firstly the courses durations into set of lessons. One benefit to transform timetabling problem in RCPSP is that durations of lessons can be set according to choice but normally solvers take lesson length uniform for the easiness and the other thing is that if there are precedence constraints (i.e lesson i of duration d_i must be taught before lesson j of duration d_j) between lessons then this kind of formulations will be more beneficial to use than others timetabling formulations. The other aspect of attention is a new dimension of thought and beauty of mathematical work which can open new rooms for researchers. We are working on the idea that how many other features of RCPSP and its generalizations can be attached to timetabling problems that these problems could be solved by using RCPSP solvers or techniques.

C. The Resource Constrained Project Scheduling Problem (RCPSP)

The classical resource constrained project-scheduling problem (RCPSP) may be stated as follows.

A project consists of a set of n activities numbered 1 to $\bar{j}$, where each activity has to be processed without interruption to complete the project. One consider additional activities $j = 1$ and $j = \bar{j}$ representing the single source and single sink activity of the network respectively. The duration of an activity j is denoted by d_j, where d_1=0 and $d_{\bar{j}} = 0$. There are R renewable resource types. The availability of each resource type r in each time period t is a_{rt} units, $r = 1,\ldots,R$. Each activity j requires u_{jr} units of resource r during each period of its duration where $u_{1r} = 0$, $u_{\bar{j}r} = 0$, $r = 1,\ldots,R$. All parameters are assumed to be non-negative integer valued.

There are precedence relations of the finish-start type with a zero parameter value (i.e., FS = 0) defined between the activities. In other words, activity i precedes activity j if j cannot start until i has been completed. The structure of a project can be represented by an activity-on-node network G = (V,A), where V is the set of activities and A is the set of precedence relationships. F_j (P_j) is the set of successors (predecessors) of activity j. It is assumed that $1 \in P_j$, $j = 2,\ldots, \bar{j}$ and $\bar{j} \in F_j$ $j = 1,\ldots, \bar{j}$-1. The objective of the RCPSP is to find a schedule S of the activities, i.e., a set of starting times $(s_1,\ldots,s_{\bar{j}})$, where $s_1 = 0$ and the precedence and resource constraints are satisfied, such that the schedule duration $T(S) = s_{\bar{j}}$ is minimised.

Our problem is based on non preemptive activities but in RCPSP activities could be preempted during processing at integer points in time, i.e., the fixed integer processing time d_j of activity j may be split into $j = 1, 2, \ldots, d_j$ process units. Time windows can be specified for every activity, [EFj, LFj], which denote the earliest and latest finishing time for activity j, and [ESj, LSj] which denote the earliest and latest starting time for activity j. One can use critical path analysis to determine these time windows.

A schedule S is called feasible if in each time period t the total resource demand is less than or equal to the availability of each resource type r, and the given precedence constraints are fulfilled. We call a problem of finding a feasible schedule with completion times Cj such that $Cj \le T$ for $j = 2,\ldots, \bar{j}$-1 a search problem or feasibility problem. A search problem with threshold value T has a solution if and only if a schedule S exists such that the makespan $C_{\max} = \max_{j=2}^{\bar{j}-1}(C_j) = \max_{j=2}^{\bar{j}-1}(s_j + d_j)$ is not greater than T. The RCPSP is usually formulated as the problem of finding a feasible schedule which minimizes the makespan. Other important objective functions besides are based on cost functions $f_j(t)$ for the activities. One has to find a feasible schedule which minimizes the total costs $\sum_{j=2}^{\bar{j}-1} f_j(C_j)$.

II. Literature Review

The timetabling problems were soIved by many authors, some used real data of any university, college or high school and some for assumed data. Breslaw [6] provided a solution for the faculty assignment problem, a problem closely related to the timetabling problem, using linear programming models. Schimmelpfeng and Helber [7] described an integer programming approach which has been implemented at the School of Economics and Management at Hannover University, Germany, to create the complete timetable of all courses for a term and formulation was solved with CPLEX solver.

Daskalaki and Birbas [8] presented integer programming formulation for university timetabling problem. The timetable for the Electrical and Computer Engineering Department in the University of Patras was used as a case study. Bolanda and Hughesa [9] applied blocking strategy, in which the classes can be partitioned into sets of classes (or blocks) that will be timetabled in parallel. The problem of constituting the blocks and populating the classes, known as the course blocking and population problem. This formulation was made for high school timetabling and model was implemented in the modelling language AMPL, and solved using the ILOG package CPLEX 8.0. Birbas et al.[10] presented a 0–1 integer programming model for the timetabling problem of Greek high schools. In their model, a binary variable indicates whether or not a specific lesson to be taught by a given teacher is to be held at a specific time of the week. The model generates timetables that satisfy all the hard and soft constraining. Werra [3] presented some basic models for course timetabling problem and these were described with an emphasis on graph theoretical models.

Mingozzi et al. [11] presented a 0-1 linear programming formulation that requires an exponential number of variables corresponding to all feasible subsets of activities that can be simultaneously executed. Francisco Ballestin et al.[12] study the case when pre-emption is allowed for processing jobs. The generalized case of this problem is m_PRCPSP, which means that one job can be pre-empted at most m times but case studied in this paper is 1_PRCPSP, for reason, it is easy and also if one pre-empted in more time, objective function normally does not improve.

Sonke Hartmann and Dirk Briskorn [13] give an overview over extensions of the RCPSP such as multiple modes, minimal and maximal time lags. The extensions are classified according to the structure of the RCPSP. They summarize generalizations of the activity concept, of the precedence relations of the resource constraints and discuss the notations, models and classification schemes.

Sonke Hartmann and Rainer Kolisch [14] first present a literature survey. They discuss about X-pass method, in which they use SGS (Serial schedule generation scheme). Here they present the results and find out the performance of many state of the heuristics on some benchmark instances. They compare the results and point out the most performing procedure.

[15] Jat and Yang solved a timetabling problem proposed by Metaheuristic Network and spon sored by PATAT and they solved the problem with genetic algorithm. This problem was a simple problem which has a few constraints. [16] Jain and Chande discussed the problem of their university. They expressed the problem and proposed a genetic algorithm but did not solve any problem to test their algorithm. [17] Salwani Abdullah , Hamza Turabieh proposed memetic algorithm which hybridises a genetic algorithm with a Tabu Search Algorithm. This algorithm uses neighbourhood structures during the search process to get significant improvements in solution quality. They use International timetabling competition track curriculum based course timetabling (ITC 2007: CB-CTT) instances for numerical application.

III. General Features of the Models

In this section we shall define our sets and sub sets, which will be used in our formulations.

- A set of lessons $J=\{1,\ldots, \bar{j}\}$.
- A set of types of rooms $R=\{1,\ldots, \bar{r}\}$.
- A set of rooms $Y=\{1,\ldots, \bar{y}\}$.
- A set of time periods $T=\{1,\ldots, \bar{t}\}$. T is a set of time periods, which all have same length.
- A set of classes $C=\{1,\ldots, \bar{c}\}$. Class is a set of lessons which have common students.
- A set of teachers $P=\{1,\ldots, \bar{p}\}$. Each lesson will have a teacher previously assigned to it.

Some additional parameters and sets are defined on the basis of previous sets to make easy the presentation of model.

a_{rt} = Availability of rooms of type r in period t

u_{jr} = Use of room type r per period by job j, which is always one

J_p =Set of lessons taught by teacher p

J_r = Set of lessons requiring rooms of type r

Y_r = Set of rooms of type r

d_j = Duration of lesson j

ES_j = Earliest starting time of lesson j

LF_j = Latest finishing time of lesson j

P_j = Set of lessons which immediately precede lesson j

F_j = Set of lessons which immediately follow lesson j

Lesson 1 and lesson $\bar{j}$ are dummy lessons, which are called generally source and sink. To ease presentation, durations and resource usages for these lessons is considered zero. Earliest starting and latest finishing times can be obtained by a forward and backward pass respectively. Starting with $ES_1 = EF_1 = 0$, the forward pass calculates earliest starting and finishing times as follows.

$$ES_j = \max\{EF_i / i \in P_j\};$$

$$EF_j = ES_j + d_j \text{ for } j = 2,\ldots, \bar{j}.$$

The backward pass is performed beginning with $LF_{\bar{j}} = LS_{\bar{j}} = \bar{t}$. This gives latest finishing and starting times LF_j and LS_j as follows.

$$LF_j = \min\{LS_h / h \in F_j\};\ LS_j = LF_j - d_j \text{ for } j = \bar{j} - 1,\ldots,1$$

E(t) = $\{j / J$ and $ES_j + 1 \le t \le LF_j\}$. This is a set of lessons which are eligible to schedule for a period t.

The latest finishing and earliest starting times correspond to time points delimiting periods. So it is important to clear difference of time period and time point for better understanding of the formulation. Two time points t and t+1 define the start and the end of period t+1 respectively. If earliest starting time of any lesson is ES_j then the earliest time period for its execution could be $ES_j + 1$.

A. The Mathematical Formulations

We have formulated timetabling problem in a two different ways on the prototype of Resource constrained project scheduling problem. The purpose of writing two models was to express two different thoughts about same problem. These are two logics to demonstrate the same timetabling problem. We have used the objective function, which was used in classical article of D. de WERRA [3].

B. Mathematical Model 1

With these variables, this is our first formulation.

$\{x_{jt} = 1$ if lesson j is scheduled in period t, $x_{jt} = 0$ otherwise;$\}$ for $j \in J$ and $t \in$ T

Max $\sum_{t\in T}\sum_{j\in J} x_{jt} d_{jt}$ where d_{jt} is the desirability to happen of lesson j in period t.

$$\sum_{t\in T} x_{jt} = d_j \quad j \in J \tag{1}$$

The constraint (1) ensure that each lesson is scheduled for d_j periods.

$$d_j.(x_{jt} - x_{j,t+1}) - \sum_{q=ES_j+1}^{t} x_{jt} \le 0 \qquad j \in J \text{ and } t \in [ES_j + 1, \ldots, LF_j - 1] \quad (2)$$

The constraint (2) is a non preemption constraint which ensures that processing of each lesson is not interrupted.

$$d_i.x_{jt} - \sum_{q=ES_i+1}^{t-1} x_{it} \le 0 \qquad j \in J,\ i \in P_j,\ t \in [ES_j + 1, \ldots, LF_i] \quad (3)$$

The constraint (3) shows that a lesson *j* must not be started before all its predecessors have been processed completely.

$$\sum_{j \in E(t)} u_{jr}.x_{jt} \le a_{rt} \quad r \in R \text{ and } t \in T \quad (4)$$

The constraint (4) proves that the number of lessons scheduled in period *t* requiring rooms of type *r* will be less than or equal to the number of rooms of type *r* available at period *t.*

$$\sum_{j \in J_p} x_{jt} \le 1 \quad t \in T \text{ and } p \in P \quad (5)$$

The constraint (5) demonstrates that teacher *p* cannot teach more than one lesson at period *t*.

$$\sum_{j \in c} x_{jt} \le 1 \qquad t \in T \text{ and } c \in C \quad (6)$$

The constraint (6) ensures that class *c* cannot attend more than one lesson at period *t.*

$$x_{jt} \in \{0,1\} \qquad j \in J \text{ and } t \in T$$

Where d_{jt} is desirability to schedule lesson *j* in period *t*. Basically this is preference to teach lecture for teachers in time periods because sometimes they are performing some other administration duties and some teaching periods are more suitable for them than others.

Objective function can be used according to demand, if one wants to schedule these lessons as early as possible, one can use Min $\sum_{t=ES_n+1}^{LF_j} t.x_{jt}$, which is same as one uses in RCPSP (minimize the project completion time).

C. Mathematical model 2

The second formulation is proposed using these variables. This is equivalent to the first formulation.

{ x_{jyt} = 1 if lesson *j* is scheduled in period *t* at room *y*, x_{jt} = 0 otherwise;} for $j \in J$, $y \in Y$ and $t \in$ T

$$\text{Max} \sum_{y \in Y} \sum_{t \in T} \sum_{j \in J} x_{jyt} d_{jt}$$

$$\sum_{y \in Y} \sum_{t \in T} x_{jyt} = d_j \qquad j \in J \quad (7)$$

$$\sum_{y \in Y} d_j.(x_{jyt} - x_{j,t+1}) - \sum_{y \in Y} \sum_{q=ES_j+1}^{t} x_{jyt} \le 0 \qquad j \in J \text{ and } t \in [ES_j + 1, \ldots, LF_j - 1] \quad (8)$$

$$\sum_{y \in Y} d_i.x_{jyt} - \sum_{y \in Y} \sum_{q=ES_i+1}^{t-1} x_{iyt} \le 0 \qquad j \in J,\ i \in P_j,\ t \in [ES_j + 1, \ldots, LF_i] \quad (9)$$

$$\sum_{j \in J_r} \sum_{y \in Y_r} x_{jyt} \le a_{rt} \qquad r \in R \text{ and } t \in T \quad (10)$$

$$\sum_{y \in Y} \sum_{j \in J_p} x_{jyt} \le 1 \qquad t \in T \text{ and } p \in P \quad (11)$$

$$\sum_{y \in Y} \sum_{j \in c} x_{jyt} \le 1 \quad t \in T \text{ and } c \in C \quad (12)$$

$$x_{jyt} \in \{0,1\} \qquad y \in Y,\ j \in J \text{ and } t \in T$$

The constraints 1 and 7, 2 and 8, 3 and 9, 4 and 10, 5 and11, 6 and12 are representing the same constraints.

IV. The Genetic Algorithm

Genetic algorithms are metaheuristics which mimics the process of natural evolution. John Holland's book "Adaptation in natural and artificial systems" as well as De Jong's "Adaptation of the behavior of a class of genetic adaptive systems," both published in 1975, are seen as the foundation of Genetic Algorithms (GAs) [18].

GAs have been used for timetabling since 1990 [19]. Since then, there are a number of papers investigating and applying GA methods for the Course timetabling problem [20].

Algorithm: Pseudo code for Genetic Algorithm
input : A problem instance I
set the generation counter g := 0
{initialize a random population}
while (solution_colony. population_size < **n) do**
create an empty timetable
assign the lessons randomly
timetable after applying Two step Verification
calculate the cost of timetable
enter this timetable to the population colony

end while
while the termination condition is not reached **do**
kill costly timetables of the colony
while (solution_colony. population_size < **n) do**
choose two parents via Roulette Wheel Selection
randomly choose crossover points
swap genetic material between two chromosomes.
child solution generated by applying the crossover operator with a cross over rate
child solution after mutation with a mutation rate
Calculate the cost of child solution
enter this child timetable to the population colony
end while
g := g + 1
end while
output : The best achieved solution for the problem instance **I**

A. Chromosome Representation

The timetable is a collection of each room timetable, room timetable is a two dimensional array. If no lesson is booked in any time period, it is called null booking which has value zero. Every timetable stores information that which lesson is placed in which room at what time on which day of the week, each booking is one gene. A time table has many fields to store information about its genetics, costs, number of violations of constraints. A population is a collection of timetables, which also have many fields to store information about timetables like less costly timetable, most costly timetable, average cost, average number of violations of constraints and the total number of timetables in population. Timetables are ordered from least costly to most costly. A two stage verification strategy is used which ensures that each lesson of a course is scheduled exactly once. It is done in two steps, in first step checking each lesson which appear more than once altered in such a way that it appear exactly once and in second step any lessons which did not appear are booked to spare spaces randomly.

The benefit of this representation is that room must not be double booked and every lesson must be scheduled at once.

A university timetable stores information about what classes are booked in each room, at any hour of the day, on any day of the week. Each of these bookings (or NULL bookings) is one gene. A timetable also has fields which describe (decode) some aspect of this genetic information. A timetable has a field which stores its cost. It also has fields which store the number of breaches of each type of hard constraint.

B. Evaluation

Each timetable is evaluated when considering all constraints.

If M={1,…, $\bar{m}$ }is the set of total constraints for problem and α is the total number of violations of constraint m∈ M.

If ϑ_i is the penality of ith violation of constraint m∈ M then cost of a chromosome could be determine with this formula.

Fitness value of a timetable = $\sum_{m \in M} P_{(m)}$, where

$$P_{(m)} = \sum_{i=1}^{\alpha} \vartheta_i$$

C. Genetic Operators

1) Roulette wheel Selection

This genetic operator is used to select potentially useful solutions for recombination. We select Time tables with Roulette wheel selection from the population for breeding.

2) Crossover

We select timetables for breeding with roulette wheel selection from population. Unity order based cross over is applied on parents to bred a child. In this way each parent has an equal chance of providing each gene for child.

3) Mutation

Mutation is used to avoid from getting trapped on local optima. Probability that a gene will undergo mutation is 2 * Mutation rate /1000.

We have used an elitist natural selection operator for timetables eradication because timetables are in ordered link list, so its easy to use. We eradicate 50 % of timetables in each generation.

V. Data Generation

We generate data for our algorithm randomly. Total numbers of lessons, classes, teachers, rooms, room types, total number of periods are fixed.

1. For each class, number of lessons is randomly generated from 4 to 20; a lesson can be part of more than one class.
2. Each lesson is assigned randomly to a teacher in such a way that each teacher must have a minimum two lessons to teach.
3. Every lesson is randomly assigned a room type from a given set of room types (laboratory, with computers, projector or any other equipment).
4. Each lesson has 1 to 2 predecessors chosen randomly in such a way that they must satisfy the feasibility. Infeasibility means that lesson *a* is preceded by *b* and *b* is preceded by *a*.
5. Each lesson's duration is randomly chosen between 1 and 3.
6. We generate randomly a desirability chart for each size problem in such a way that 80 % values of the matrix (integer value only) are 1 and remaining others are 0.

VI. Experimental Results

To check the performance and efficiency of the algorithm, we apply the algorithm on randomly generated instances. We have generated 6 small, 6 medium and 6 large size problem instances for this purpose.

The program is coded in C and run on a Intel 2.5 GHz RAM 3.5 Go pc. We run algorithm 10 times for each problem instance to see its performance in detail.

For small size problem instances, we take, J=50, P=7, C=4, Y=3, R=1, T=40, TD=90, where J is number of lessons, P is number of teachers, C is number of classes, Y number of rooms, R number of room types, T is number of periods, TD is the total duration used by all lessons. After running each problem instance 10 times, we have got maximum cost 87 and minimum cost 79.

For medium size problem instances, we take, J=120, P=24, C=14, Y=8, R=2, T=40, TD=240. After running each problem instance 10 times, we have got maximum cost 235 and minimum cost 223.

For large size problem instances, we take, J=220, P=40, C=25, Y=13, R=3, T=40, TD=450. After running each problem instance 10 times, we have got maximum cost 438 and minimum cost 421.

Our stopping criteria for algorithm is maximum cost 90, 240, 450 for small, medium, large instances respectively or 2000 iterations, which comes first. Parameters for genetic algorithm are as follows: mutation rate = 6, crossover rate =2, population size for small and medium problem = 100, population size for large problem 250. Reason behind taking big population size for large problem is that it enhances diversity and reduces the number of generations.

VII. Conclusion and Future Work

An introduction to timetabling and RCPSP is given. We have proposed two equivalent mathematical formulations which are based on RCPSP.

The course timetabling problem can be formulated to RCPSP by using these models. We use genetic algorithm to solve problem. We get promising results but comparison with other techniques is difficult. Because these formulations transform timetabling problem into RCPSP but in literature most treated timetabling problems have different constraints than this problem. Literature timetabling problems focus on any real data problem or any generalized problem which have a lot of constraints.

Our purpose for this article is not to deal many constraints but is to provide a theoretical foundation for transformation of these two famous problems on analogous bases.

We are working on the university course timetabling problem and we would like to add some more features of RCPSP and correlate them to timetabling problem.

References

[1] P. Brucker and S. Knust. Resource-constrained project scheduling and timetabling. Burke and Erben (Eds): PATAT 2000, LNCS 2079, *Springer-Verlag Berlin Heidelberg*: 277-293, 2001.

[2] A.Wren. Scheduling, timetabling and rostering-A special relationship. Burke and Ross (eds), *Springer-Verlag Berlin Heidelberg*, 46-75, 1996.

[3] D.Werra. An introduction to timetabling. *European journal of Operational research*, 19: 151-162,1985.

[4] E. K. Burke and S.Petrovic. Recent research directions in automated timetabling. *European Journal of Operational Research,* 140: 266-280, 2002.

[5] R.Lewis. A survey of metaheuristic-based techniques for University Timetabling problems. *OR Spectrum*, 30:167–190, 2008.

[6] J. A.Breslaw. A linear programming solution to the faculty assignment problem. *Socio-Economic Planning Science*, 10: 227-230, 1976.

[7] K. Schimmelpfeng and S.Helber. Application of a real-world university-course timetabling model solved by integer programming. *OR Spectrum*, 29:783–803,2007.

[8] S.Daskalaki,T.Birbas and E.Housos. An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research*, 153: 117-135, 2004.

[9] N. Boland, B. D. Hughes, L.T.G. Merlot and P. J. Stuckey. New integer linear programming approaches for course. *Computers & Operations Research*, 35: 2209-2233, 2008.

[10] T. Birbas, S. Daskalaki and E. Housos. Timetabling for Greek high schools. *Journal of the Operational Research Society*, 48: 1191–1200, 1997.

[11] A. Mingozzi, V. Maniezzo, S. Ricciardelli and L.Bianco. An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation. *Management Sci.* 44:714–729,1998.

[12] F.Ballestın,V.Valls and S. Quintanilla. Pre-emption in resource-constrained project scheduling. *European Journal of Operational Research*, 189:1136-1152,2008.

[13] S.Hartmann and D. Briskorn. A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, 207: 1–14,2010.

[14] S. Hartmann and R. Kolisch. Experimental evaluation of the state of the art heuristics for the resource constrained project scheduling problem. *European Journal of Operational Research*, 127:394-407, 2000.

[15] S.N.Jat and S.Yang. A Guided Search Genetic Algorithm for the University Course Timetabling Problem. *Multidisciplinary International Conference on Scheduling* : Theory and Applications (MISTA 2009), Dublin, Ireland, 10-12 August 2009.

[16] A.Jain, S.Jain and P.K. Chande. Formulation of Genetic Algorithm to Generate Good Quality Course Timetable. *International Journal of Innovation, Management and Technology*, Vol. 1, No. 3, ISSN: 2010-0248, August 2010.

[17] S. Abdullah and H. Turabieh . On the use of multi neighbourhood structures within a Tabu-based memetic approach to university timetabling problems. Information Sciences, 191: 146-168, 2012.

[18] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[19] A. Colorni, M. Dorigo, and V. Maniezzo. Genetic algorithms - A new approach to the timetable problem. In Akgul et al. (eds.), *NATO ASI Series, Combinatorial Optimization*, Lecture Notes in Computer Science, F(82), pp. 235-239, 1990.

[20] M. W. Carter and G. Laporte. Recent developments in practical course timetabling. *Proc. of the 2nd Int. Conf. on Practice and Theory of Automated Timetabling*, LNCS 1408, pp. 3–19, 1998.