A Multi-objective optimization approach for solving university course timetabling problem - A case study of International University, VNU in HCMC, Vietnam

Nguyen Huu Tri †
Department of Industrial & Systems Engineering
International University-VNU, Ho Chi Minh City, Vietnam
Tel: (+84) 903381498. Email: nhtri@hcmiu.edu.vn

Ho Thanh Phong
Department of Industrial & Systems Engineering
International University-VNU, Ho Chi Minh City, Vietnam
Tel: (+84) 903718904. Email: htphong@hcmiu.edu.vn

Ho Doan Quoc
Department of Industrial & Systems Engineering
International University-VNU, Ho Chi Minh City, Vietnam
Tel: (+84) 908244933. Email: hdquoc@hcmiu.edu.vn

Abstract. University course timetabling problem is always a challenge for researcher due to the inherent complexity and variability of the problem. In this paper, two objectives were considered while optimizing the real course timetabling at the International University were considered, including i) the maximization the number of courses assigned and ii) the minimization timetable compactness. The Mixed Integer Programming (MIP) was employed for formulating and solving individual optimization problem. To deal with multiple objectives, Goal Programming (GP) approach was utilized to achieve the best compromised solutions. The solutions were obtained by using optimization software CPLEX. Computational results revealed that this approach yields better schedules in terms of timetable compactness as compared to the currently existing approach.
Key words: University course timetabling, Mixed integer programming, Multi-objective optimization, Goal programming, Timetable compactness.

1. INTRODUCTION

In recent years, the development of society has led to the blooming expansion of universities. As such, the number of student enrollment and courses has been increased dramatically. In order to maintain the quality of education, each university itself is enforced to deliver good timetables for students and teachers. To have an efficient timetable, each educational institution has to make the arrangement of limited resources of teachers and classrooms for scheduling a reasonable timetable. Timetabling problem has proved its attraction to a huge of researchers not only in the past, but also in the time being because this problem is in class of NP-hard problem. That is to say, finding an optimum solution gets more and more difficult as the model expands in terms of size as well as specialized requirements from each educational institution.

"Timetabling represents the most important planning exercise in the school calendar. It not only gives practical expression to the curricular philosophy of the school, it sets, maintains and regulates the teaching and learning pulse of the school and ensures the delivery of quality education for all students" Jardine (2006).

From what educators defined timetabling term, timetabling plays the extreme important role for educational institution.

Wren (2005) has defined timetabling as follows: *"Timetabling is the allocation, subject to constraints, of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives."*

According to Edmund Kieran Burke (2002), university timetabling problem is decomposed into two main categories: course timetabling and exam timetabling. These problems are

subject to some constraints which are usually categorized to two types: hard constraints and soft constraints Burke (1997). Hard constraints are constraints that must be satisfied under any circumstances while soft constraints are those can be violated but the violations may affect the quality of teaching Shao-wen (2014).

In order to solve the university timetabling problem, there are a lot of approaches used by research works. According to the researchers, the approaches are classified into some categories as follows:

1. Integer Programming
2. Constraints Satisfaction Programming
3. Genetic Algorithms
4. Tabu search

Integer Linear Programming: Linear programming is a mathematical programming technique to solve optimization problems. A linear programming problem is a mathematical formulation of an optimization problem defined in terms of an objective function and a set of constraints. The objective function is a linear function of the unknowns (variables) and the set of constraints consists of linear equalities and linear inequalities. Integer linear programming (ILP) or simply integer programming (IP) is a subset of linear programming, where some or all of the variables are restricted to take only integer or whole number (as opposed to fractional) values. If all the variables are restricted to take only integer variables, the problem is called pure integer linear programming problem. If the restrictions are such that, some but not all of the variables can take only integer values, then such a problem is said to be mixed integer linear programming problem. If the variables may take either 0 or 1, then such a problem is called binary integer linear programming problem Hillier (2010).

Daskalaki et al. (2004) suggested a novel integer programming to model the timetabling problem. With the very large number of possible variables such as groups of students, teachers, courses, classrooms, they proposed 0-1 binary variables to bring great flexibility in the modeling process by reducing the large number of variables to manageable sizes. By minimizing the cost function, the model then is solved by commercial software using branch-and-bound approach.

Skocdopolova (2012) proposed integer goal programming to solve the real timetabling problem at department of econometrics at University of Economics, Prague. A sequential three-stage goal programming model is introduced. This approach involves three stages. At each stage, the optimal values are used as inputs for the next stage. At first teachers are assigned to courses, then courses are assigned to time slots, and finally time slots are assigned to classrooms.

Mixed integer programming (MIP): A mixed-integer programming (MIP) problem is one where some of the decision variables are constrained to be integer values (i.e. whole numbers such as -1, 0, 1, 2, etc.) at the optimal solution.

The "classic" method for solving these problems is called Branch and Bound. This method begins by finding the optimal solution to the "relaxation" of the problem without the integer constraints (via standard linear or nonlinear optimization methods). If in this solution, the decision variables with integer constraints have integer values, then no further work is required. If one or more integer variables have non-integral solutions, the Branch and Bound method chooses one such variable and "branches," creating two new sub problems where the value of that variable is more tightly constrained. These sub problems are solved and the process is repeated, until a solution that satisfies all of the integer constraints is found.

To be specific, Aizam (2012) presented Mixed Integer Linear Programming (MLIP) models which incorporate all hard constraints and the desirable soft constraints. Interestingly, the university timetabling problem has been decomposed into three sub models for different purposes. First, the first model solves hard constraints, then the second model solves the soft constraints. The last one is the combination of the first and the second models to be solved. The computational results proved that MLIP is capable of generating university course timetabling.

The tabu search: The tabu search is a metaheuristic originally introduced by Glover (1990). Starting from a candidate solution (also known as a potential solution), the tabu search is a local search algorithm that moves from one candidate solution to another candidate solution (referred to as a neighbor) until some problem dependent termination criteria has been met.

Moving from a candidate solution to a neighbor is accomplished using a move operator, where a single change is made to the candidate solution resulting in its neighbor. A neighborhood is defined as a set of neighbors that occur as a result of implementing a single move to a candidate solution. In order to prevent the search from cycling (returning to a previously encountered area of the search space), a tabu list is kept. This tabu list stores a set of k recent candidate solutions. Alternatively, previous moves that have been applied to candidate solutions can also be stored in the tabu list Schaerf (1996).

A move is rejected if it results in a candidate solution that is in the tabu list. Another move must then be made. An advantage of the tabu search is its use of the tabu list. This list resembles a form of memory, preventing the search algorithm from returning to previous candidate solutions. A disadvantage of the tabu search is that the focus is always on a single candidate solution. Thus, the possibility arises that a large area of the search space is not covered Sastrodjojo (1998).

Zhipeng L¨ua (2001) presents an Adaptive Tabu Search algorithm for solving the problem of curriculum-based course timetabling. The proposed algorithm follows a general framework composed of three phases: initialization, intensification and diversification. The initialization phase is primarily aimed to construct a feasible initial timetable using a fast greedy heuristic. When a feasible initial assignment is reached, an adaptively combined intensification (Tabu Search) and diversification (Perturbation Operator from Iterated Local Search) phase is used in order to reduce the number of soft constraint violations without breaking hard constraints any more.

Genetic Algorithms: The concept of GAs were essentially invented by one man—John Holland—in the 1960's. Genetic algorithms (GAs) are numerical optimization algorithms that are as a result of both natural selection and natural genetics. The method which is general in nature is capable of being applied to a wider range of problems unlike most procedural approaches. Genetic algorithms help to solve practical problems on a daily basis. The algorithms are simple to understand and the required computer code easy to write. Alberto Colorni (1990) compared two versions of the genetic algorithm (GA), with and without local search, both to a handmade timetable and to two other approaches based on simulated annealing and tabu search. The results show that GA with local search and tabu search with relaxation both outperform simulated annealing and handmade timetables.

Constraint satisfaction problem (CSP): deals with assignment of values from its domains to each variable such that no constraint is violated. CSP has three components: variables, values and constraints. In general, CSP consists of: a finite set of variable X = {x1,…,xn} with respective domains D = {D1,…, Dn} which list the possible values for each variable Di = {vi,…,vk} and a set of constraints C = {C1, …, Ct} .The constraints limit the possible values that a variable can have. A solution of a CSP is a consistent assignment of all variables to values in such a way that all the constraints are satisfied Zhang (2005). A sample case study problem is investigated and a constraint satisfaction programming approach is implemented using ILOG Scheduler and ILOG Solver.

Compactness is a type of constraint that is usually required in school timetabling for the students but not required for the teachers. There are several research works mentioned compactness and specifically modeled as a hard constraint (Andreas Drexl et al., 1997). On the other hand, some researchers presented the compactness as soft constraints or modeled as objective function. Lübbecke (2010) regarded compactness for curriculum as soft constraints because the purpose is to schedule the corresponding courses consecutively over a day for every curriculum. In contrast, Ramon Alvarez-Valdes (2002) presented the concept of forming compactness with the aim that every student's timetable must be as compact as possible. As such, they proposed the calculation of compactness based on the outputs from scheduling student timetables.

This paper deals with the compactness of every student class. The difference is that constraints are added to the model and objective function is built as goal of scheduling timetabling in VNUIU. Figure 1 below is the illustration of the way to calculate the compactness. The mathematical formulation is presented in the modeling section.

Morning	1	2	3	4	5	6
	■		■		■	
Afternoon	7	8	9	10	11	12
	■		■		■	

(a)

Morning	1	2	3	4	5	6
	■		■		■	
Afternoon	7	8	9	10	11	12
		■		■		

(b)

Figure 1: (a) Example of a compact timetable where shadow zones stand for assigned courses; (b) Example of a non-compact timetable

As for example (a), the total compactness is 3 because three days without classes will be received the value of one per each. Conversely, the total compactness for the example (b) are 11 because five days with only one single class will be received the value of two per each and one day without classes will be received the value of one

2. MODEL DEVELOPMENT:

2.1. Assumptions:

- Each course is taught by one lecturer;
- Each course is already assigned one lecturer except for general courses such as physical training, politics courses, etc.…
- Each course is assigned one session per week regardless of the length of each course. For those courses with two credits, they are required to be assigned consecutively over a session such as IE 1, IE2, etc.….
- Laboratory courses are treated as separate courses;
- Courses are scheduled on a weekly-basis and there is no change of the schedule until the end of semester.
- Each student class requires prerequisite courses. These courses are required to schedule in different timeslots. Furthermore, each semester, faculties allow students to register some elective courses as

well as courses which students have failed in earlier semesters. These courses can be schedule simultaneously with required courses belonging to each student class.

- Each session lasts for four periods with 180 minutes per session including break time.
- Each course is scheduled to one classroom.

2.2. Notation:

Variable	Definition
PARAMETERS	
C	Set of courses
C_{type}	Type of course
L	Set of lecturers
R	Set of classrooms
TY	Set of type of classrooms (theoretical, biotech laboratories…)
$Capa_{type}$	The number of availabilities of each type of classroom
T	Set of timeslots
G	Set of student classes
$M_{l,t}$	Lecture-timeslot matrix. Binary matrix in which a 1 in position (l, t) means that lecturer l is available for teaching at timeslot t and vice versa.
$K_{c,l}$	Course-Lecturer matrix. Binary matrix in which a 1 in position (c,l) means that lecturer l can teach course c and vice versa.
$H_{c,g}$	Course-Group matrix. Binary matrix in which a 1 in position (c,g) means that required course c is studied by group g and vice versa.
DECISION VARIABLES	
$x_{c,l,t}$	$\begin{cases} 1, if \text{ course c taught by lecturer l can be assigned to timeslot t} \\ 0, otherwise \end{cases}$
$assGroupTimes_{g,t}$	$\begin{cases} 1, & \text{if course c belonging to group g can be assigned to timeslot t} \\ 0, otherwise \end{cases}$
$compactGroup_{g,t}$	$\begin{cases} 0, \text{If } assGroupTimes(g,t) = 1 \text{ and } assGroupTimes(g,t+6) = 1 \\ 1, \quad \text{if } assGroupTimes(g,t) = 0 \text{ and } assGroupTimes(g,t+6) = 0 \\ 2, otherwise \end{cases}$

It can be seen that this problem is complicated and requires many combinations of integer values, which can rise exponentially with the size of the problem. In order to reduce the number of variables and optimize the performance of CPLEX solver, some binary matrices are employed to the model. Thanks to CPLEX Optimization, by using these matrices, the numbers of generated variables have been reduced.

Below is the formulation of university timetabling model. This model addresses solving two critical requirements at VNUIU. Firstly, it aims to resolve the number of courses can be scheduled. This is a must for not only in VNUIU, but also in other universities. Each semester, VNUIU has to schedule a fixed number of courses for student registration. The problem is that VNUIU sometimes cannot schedule all courses right away due to special requirements. As such, it expects to schedule as many courses as possible. This requirement is put to the first objective which maximizes the number of courses assigned.

Because of a relative far distance from downtown, the students love to study a full day without single session per day. It is convenient for them if they have more spare time to do personal works and out-of-class study. As such, the second objective is formulated to address this issue. It is seen as "soft" objective due to the fact that it can be violated in some cases because of limited resources such as classroom, availability of lecturer, etc.

Then, the group of constraints is built to guarantee that that there is no violation. What makes difference when modeling this problem is that there are fewer constraints than some previous university timetabling problems while solving multi-objective problem, which contributes to the reduction of variables and iterations.

2.3. Mathematical Model:

Objective 1: Maximize the number of courses can be assigned

Maximize

$$\sum_{c\in C} \sum_{l\in L} \sum_{t\in T} x_{c,l,t} \quad \forall c \in C, t \in T, l \in L \qquad (1)$$

Objective 2: Minimize the compactness of student class

Minimize

$$\sum_{g\in G} \sum_{t\in T} compactGroup_{g,t} \quad \forall g \in G, t \in T \; where \; t = 1..6 \qquad (2)$$

Problem constraints:

A lecturer l teaches at most one course at a timeslot t:

$$\sum_{c\in C} x_{c,l,t} \leq 1 \quad \forall \; t \in T, l \in L \qquad (3)$$

The 1st constraint assures that a lecturer teaches at most one course at a timeslot at any circumstances.

A course is scheduled at most one timeslot:

$\sum_{l\in L} \sum_{t\in T} x_{c,l,t} \leq 1 \forall\ c \in C$ (4)

The 2nd constraint guarantees that one course is scheduled at most one timeslot. In VNUIU, most of courses are studied one session per week.

Number of courses scheduled to timeslot t is equal to the number of rooms available for the course of type ty:

$\sum_{c\in C} \sum_{l\in L} x_{c,l,t} \leq \text{Capa}_{\text{type}}\ \forall\ t \in T, ty \in TY$ (5)

As courses assigned for student registration are divided into different type of courses, which requires different type of classrooms. As such, the 3rd constraint ensures that with each type of course has to be schedule to the appropriate type of classroom and at most the number of rooms available for such type of classroom.

Required courses for student class must not be overlapped:

$\sum_{c\in C} \sum_{l\in L} x_{c,l,t} * H_{c,g} = \text{assGroupTimes}_{g,t}\ \forall\ g \in G, t \in T$ (6)

Each semester, students belong to different student classes are required to study required courses at any circumstances. These courses are subject to their curriculum upon their entry to VNUIU. The 4th constraint assures that there is no violation of time overlapping among those courses.

Any two required courses belonging to group g at timeslot t and timeslot t+6 preferred to be assigned over a day:

$|\text{assGroupTimes}_{g,t} - \text{assGroupTimes}_{g,t+6}| * 2 \leq \text{compactGroup}_{g,t}\ \forall t \in T\ where\ t < 7, g \in G$ (7)

$|\text{AssGroupTimes}_{g,t\ -1}| \leq \text{compactGroup}_{g,t-6}\ \forall t \in T\ where\ t > 6, g \in G$ (8)

The set of two constraints (7), (8) assign the penalty to each of course cannot be paired over a day. This constraint addresses the second objective which is to minimize the compactness of student class timetables.

$X_{c,l,t} \in \{0,1\};\ \forall C, T, L$ (9)

$\text{assGroupTimes}_{g,t} \in \{0,1\};\ \forall G, T$ (10)

$\text{compactGroup}_{g,t} \in \{0,1,2\};\ \forall G, T$ (11)

Timetabling problem in this paper is a multi-objectives model, so it is better to be solved by goal programming approach. As such, after solving each objective, the optimal value obtained is kept to input the goal constraints.

There are two specialized constraints which will be added to the group of constraints to serve the goal programming process including:

$Goal\ 1: \sum_{c\in C} \sum_{l\in L} \sum_{t\in T} X_{c,l,t} + (d_1\text{-}) - (d_1+) = Z^*\ \forall\ c \in C, t \in T, l \in L$ (12)

At this constraint, in order to obtain the maximum value from goal 1, (d1-) should be minimized

$Goal\ 2: \sum_{g\in G} \sum_{t\in T} \text{compactGroup}_{g,t} + (d_2\text{-}) - (d_2+) = Y^*\ \forall\ g \in G, t \in T\ where\ t = 1..6$ (13)

Z*: the value is greater than or equal to optimal value obtained from objective 1

Y*: the value is greater than or equal to optimal value obtained from objective 2

d_1-: the amount by which the number of courses can be assigned is less than the target value

d_1+: the amount by which the number of courses can be assigned exceeds the target value

d_2-: the amount by which the total of compactness is less than the target value.

d_2+: the amount by which the total of compactness exceeds the target value

Where d_1-, d_1+, d_2-, d_2+ ≥ 0

It is the fact that the timetabling at International University should meet both the number of courses can be assigned and minimize the idle time of student class timetable. Therefore, goal objective function: Min $\{p_1*(d_1\text{-}) + p_2*(d_2+)\}$

There are two alternatives:

Alternative 1: objective 1 is considered as first priority, so p_1 is assigned a very big value in comparison with p_2

Alternative 2: objective 2 is considered as first priority, so p_2 is assigned a very big value in comparison with p_1.

3. NUMERICAL ILLUSTRATION:

A real-world data from International University is used for the input to solve the model. The inputs are as follows:

Lecturer	Course	Room	Type of classrooms	Time slot	Group
353	715	150	14	12	251

With the real-world input data at VNUIU, the problem seems to be complicated to solve in the earlier stage as the booming of generation of variables and integer values, which restricted the model to run out of memory when inputting the model to CPLEX environment. The numbers of variables generated in the first time are more than 24 million. Thanks to CPLEX, the model then is modified by using subsets which allow binary matrices to be fully used to reduce intensely the number of variables.

Consequently, the formulated problem consisted of 870.139 non-zero coefficients, 14.155 constraints and 217.349 variables was solved by optimization solver called CPLEX OPL Studio 12.0 using mixed integer programming. To solve a large combinatorial problem, the solver has been installed in workstation, Intel Xeon, 2.66 GHz, 64 GB memory.

The model is thereafter solved by following priority: solving objective 1 and then objective 2, respectively. At the first attempt, the results return very quickly with only two minutes. The results prove that the first objective is fully achieved as the aim is to maximize the assignment of

715 courses. As stated in the modeling section, this objective is the critical requirements at VNUIU.

Then, the second attempt has been done. As for this time, because of complicated combination of variables and the nature of second objective which is to minimize the compactness of student class timetables, the solving has taken more time than the first attempt to return solutions. This is the integer optimal solution with 10% gap as it did not return the exact optimal solution after solving several hours. The total compactness is 982. It can be seen that from the results that there are seven groups eliminated. The results for this objective are very favorable.

In addition, to measure the performance and results obtained when solving the model, three algorithms were employed including mixed integer programming (MIP), constraint programming (CP).

Table 1: Summary of result benchmarks

Algorithm	Scenario	Objective Running	Courses can be assigned	Total timetables compactness	Running time
Mixed Integer Programming	1	Objective 1	715	1969	1.5 minutes
Constraint programming	1	Objective 1	707	1944	45 minutes
Mixed Integer Programming	2	Objective 2	448[1]	982[2]	180 minutes
Constraint programming	2	Objective 2	#N/A[3]	#N/A[4]	More than 10 hours

At scenario 1, after running objective 1 (maximize the number of courses can be assigned), MIP yields the better solution than CP in terms of processing time and results. As such, 715 courses can be assigned when solving the model by MIP, whereas 707 courses can be assigned with CP even CP produces the better solution regarding total timetable compactness than MIP.

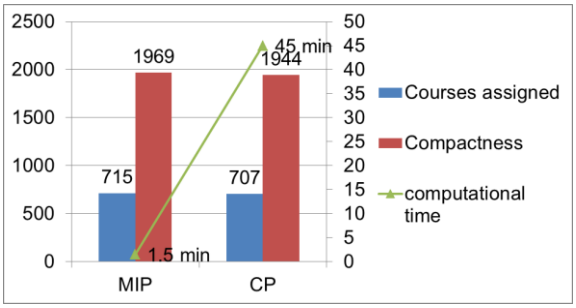

Figure 2: Measurement of performance between MIP and CP when solving objective 1

At scenario 2, the objective 2 has been solved by both MIP and CP. Interestingly, due to the complex combination of constraints and variables, this process required hours to solve. As a
result, only solving MIP return the near-optimal value with 10% GAP, while CP cannot produce the solution after more than 10 hour-running time.

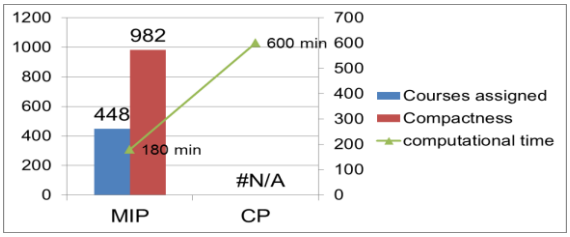

Figure 3: Measurement of performance between MIP and CP when solving objective 2

Because we cannot obtain fully the optimal solutions through the benchmarking in table 1, the goal programming approach is used to test whether the model is qualified or not to achieve the optimal solutions which can satisfy the two objectives.

In order to do that, we created two additional "soft" constraints: the first constraint is that the number of courses assigned is at least 715, so we have to minimize the amount by which the number of courses assigned fewer than 715. The second constraint is that the total compactness is at most 982, so we have to minimize the amount by which the total compactness greater than 982. In order to seek the optimal solution at this stage, there are 11 attempts running with different target values based on the priority of the goals. 1st running to 8th running addresses the priority of goal 1, while 9th running to 11th running prioritize goal 2. Then the weight is added as follows: those with first priority receive value of 1000, the remaining receive value of 1.

Table 2: Table of Priorities to solve goal programming

	Goal 1	Goal 2
Running 1	Priority	-
Running 2	Priority	-
Running 3	Priority	-
Running 4	Priority	-
Running 5	Priority	-
Running 6	Priority	-
Running 7	Priority	-
Running 8	Priority	-
Running 9		Priority
Running 10	-	Priority
Running 11	-	Priority

[1] No exact optimal solution (GAP:10%)
[2] No exact optimal solution (GAP:10%)
[3] No solution returns
[4] No solution returns

During the solving process at this stage, it takes for a while to accomplish the optimal solution because only a half of attempts running return the optimal solution. It can be seen that we cannot obtain the solutions if we prioritize the second goal as it did not return the solutions even for several hours with the powerful workstation. Table 3 clearly presents the comparison for each run.

Table 3: Result Benchmark for Goal Programming

Running	Target value for courses assigned	Target value for Compactness	Result	Remark
1	715	1500	Achieved	
2	715	1400	Achieved	
3	715	1300	Achieved	
4	715	1200	Achieved	
5	715	1050	Achieved	
6	715	1020	Achieved	
8	715	1000	No solution	After running several hours
9	500	982	No solution	After running several hours
10	650	982	No solution	After running several hours
11	700	982	No solution	After running several hours

Table 4: Result Benchmark for Goal Programming

Running	Target value for courses assigned	Target value for compactness	Number of courses assigned to group	Number of compacted groups	Computational time
1	715	1500	1327	142	6 min
2	715	1400	1327	158	6 min
3	715	1300	1327	191	6 min
4	715	1200	1327	191	6 min
5	715	1050	1327	233	270 min
6	715	1020	1327	245	600 min
7	715	1000	#N/A	#N/A	No solution after running several hours
8	715	950	#N/A	#N/A	No solution after running several hours
9	500	982	#N/A	#N/A	No solution after running several hours
10	650	982	#N/A	#N/A	No solution after running several hours
11	700	982	#N/A	#N/A	No solution after running several hours

case, all courses can be scheduled) while it yields the better compactness of timetables. The remarkable point is that using goal programming at this stage returns the optimal solutions rather than the former approaches used.

Table 5: Result comparison

	Solving objective 1	Solving objective 2
Number of groups can be assigned	251	244
Number of courses belonging to groups that can be assigned	1329	1161
Total timetable compactness	1969	982
Number of courses can be assigned	715	448

From table 5, the resulting from solving objective 1 shows that there are 715 courses assigned without violating constraints. The number of groups can be assigned and number of courses belonging to groups that can be assigned are 251, 1329, respectively. The total compactness of timetables is 1969. Things are different when solving objective 2 as the goal of objective 2 is to minimize the two required courses that can be assigned over a day; the total compactness of timetables is smaller than the earlier result. However, the number of courses assigned reduces dramatically from 715 to 448. There are seven groups dropped out due to their courses cannot assigned.

Based on the resulting table, we can observe the timetable for group 250 (MAMA15IU21) for three cases as follows:

Case 1 – Solving only first objective

Morning	1	2	3	4	5	6
		x	x			X
Afternoon	7	8	9	10	11	12
				x		x

Case 2 – Solving only second objective

Morning	1	2	3	4	5	6
					X	X
Afternoon	7	8	9	10	11	12
					x	x

Case 3 – Solving both objectives at the same time

Morning	1	2	3	4	5	6
	X			X		
Afternoon	7	8	9	10	11	12
	x	x		X		

It can be seen that in the first case, the timetable for group of student MAMA15IU21 is non-compact, whereas the second case is the compact one. However, the second case dropped out one course. The last case produces the balance solution which satisfies both objectives.

Table 6: Result comparison 2

	Current timetable system at VNU-IU	Paper
Methodology	Heuristics	MIP
Solutions	Optimal solutions are not achievable	Optimal solutions are fully achieved
Courses assigned	715	715
Compactness	2058	1020
Computational time	Approximately 4 hours	Approximately 10 hours

The current approach from VNUIU tends to use heuristics to find the solutions rather than the optimal solutions. In this aspect, the paper deals with MIP and then use goal programming to find out the optimal solutions. Practically, optimal solutions are fully achieved by the approach used in the paper.

To measure the quality of solutions to identify whether it is valid or not in comparison with the results of scheduling timetabling obtained by current system at VNUIU, table 16 presents the result comparison between two approaches. It is obvious that solutions from this paper yield the impressive results in terms of timetable compactness. Once again, it is proved that goal programming approach is capable of generating efficient timetables for the large combinatorial university timetabling problem. Needless to say, the decision maker from VNUIU could easily choose these results as an alternative in comparison with the current solutions generated by current timetable system.

4. CONCLUDING REMARKS:

Results from this paper suggest that Mixed Integer Programming is capable of generating university timetabling. It is clear that the resulting reaches VNUIU's goals for scheduling timetabling.

With the strong support to solve scheduling problem by MIP from CPLEX Optimization Studio 12, the model is refined as compact as possible as long as it reduces the number of variables to optimize the solving process. Especially, the inputs are improved very much and fully supported by CPLEX for the solving stages.

Besides, this is a multi-objective problem, by using goal programming approach, the best compromised solutions are obtained.

In order to produce efficient timetables for student registration in the future, it is suggested that there should a utilization of the outputs from CPLEX to input the current timetable system at VNUIU to save time and achieve optimal solutions.

5. PREFERENCES:

Aizam, L. C. a. N. A. H. (2012). Mixed Integer Linear Programming Models for University Timetabling. *East-West J. of Mathematics,*, 90-99.

Alberto Colorni, M. D., Vittorio Maniezzo. (1990). Genetic Algorithms and highly constrained problems: The time-tabel case. *Parallel Problem Solving from Nature*, 55-59.

Andreas Drexl , F. S. (1997). Distribution requirements and compactness constraints in school timetabling *European Journal of Operational Research, 102*, 193-214

Birbas, T., Daskalaki, S., Housos, E. (1997). Course and Teacher Scheduling in Hellenic High Schools. *Proc. of the 4th Balkan Conference on Operational Research*.

Birbas, T., Daskalaki, S., Housos, E. (1999). *Rescheduling Process of a School Timetable: The Case of the Hellenic High Schools & Lyceums.* Paper presented at the Proc. of the 5th International Conference of the Decision Sciences Institute, Athens, Greece.

Burke, E., Kingston, J., Jackson, K., Weare, R., . (1997). Automated University Timetabling: The State of the Art. *The Computer Journal, 40*(9), 565-571.

Christos Valouxis, E. H. (2003). Constraint programming approach for school timetabling. *Computers & Operations Research, 30*, 1555 – 1572.

Daskalaki, S., Birbas, T., & Housos, E. (2004). An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research, 153*(1), 117-135. doi:10.1016/s0377-2217(03)00103-6

Edmund Kieran Burke, S. P. (2002). Recent research directions in automated timetabling. *European Journal of Operational Research, 140*, 266–280.

Glover, F. (1990). Tabu Search: A Tutorial. *The Institute of Management Sciences*, 74-94.

Hillier, F. S. (2010). *Introduction to Operations Research*: McGraw-Hill Higher Education; 9th edition.

Jardine, A. (2006). *Learning and Teaching Scotland*.

Sastrodjojo, L. K. (1998). *Graph Partitioning Problems Using Tabu Search*.

Schaerf., A. (1996). Tabu Search Techniques for large high-school timetabling problems. *Tech.Rep. CS-R9611, CWI, C.S. Dept., Amsterdam, NL*.

Shao-wen, Z. H.-n. a. Z. (2014). Solving UTP Containing Combining Classes using GA. *International Journal of u-and e-Service, Science and Technology, 7*(4), 277-286. doi:10.14257/ijunnesst.2014.7.4.25

Skocdopolova, V. (2012). Construction of time schedules using integer goal programming. *Proceedings of 30th International Conference Mathematical Methods in Economics*.

Wren, A. (2005). Scheduling, timetabling and rostering — A special relationship? *Lecture Notes in Computer Science, 1153*, 46-75.

Zhipeng L¨ua, Jin-Kao Hao. (2001). Adaptive Tabu Search for CourseTimetabling. *European Journal of Operational Research*.