NEW ALGORITHM TO SOLVE EXAMINATION TIMETABLE PROBLEM

Seifedine Kadry[1], Bilal Ghazal[2]

[1,2]American University of the Middle East, Egaila, Kuwait
Lebanese University, Zahle, Lebanon

ABSTRACT

In examination timetabling we have to assign the exams of a given set of courses to time slots, such that conflict exams are assigned to different time slots. Many authors have presented different approaches for solving a myriad of variants of the examination timetabling problem. In this paper, we describe a new proposed model for solving examination timetabling problem, we start by giving some definition about the graph theory, maximal independent set and there terms that using in the our model, then we describe our new algorithm.

Keywords: Scheduling, Timetabling, Examination Timetabling, Graph Coloring Heuristics, Maximal Independent Set.

I. INTRODUCTION

In this article, we develop a new algorithm for solving examination timetabling problem. We want to avoid as much as possible that students have two exams in adjacent period, in addition to minimize the whole period time. It follows from the outcome of the problem of minimizing the number of time slots that there is no solution possible in which no student has two exams per day. And minimize the number of day to achieve all exams. Hence, we treat it as a soft constraint, and we include the number of students with two exams in adjacent period in the objective function. In the below sections we provide some definition used in our model from graph theory, Maximal Independent Set, and Multicriteria Approach.

II. GRAPH THEORY

Graph theory may be said to have its beginning in 1736 when EULER considered the (general case of the) Königsberg bridge problem: Does there exist a walk crossing each of the seven bridges of Konigsberg exactly once? It took 200 years before the first book on graph theory was written. This was “Theorie der endlichen und unendlichen Graphen” by KÖNIG in 1936. Since then graph theory has developed into an extensive and popular branch of mathematics, which has been applied to many problems in mathematics, computer science, and other scientific and not-so-scientific areas. Graph theory has abundant examples of NP-complete problems. Intuitively, a problem is in P (Solvable – by an algorithm – in polynomially many steps on the size of the problem instances) if there is an efficient (practical) algorithm to find a solution to it [1]. On the other hand, a problem is in NP (Solvable nondeterministically in polynomially many steps on the size of the problem instances), if it is first efficient to guess a solution and then efficient to check that this solution is correct. It is conjectured (and not known) that P = NP. This is one of the great problems in modern mathematics and theoretical computer science. If the guessing in NP-problems can be replaced by an efficient systematic search for a solution, then P=NP. For any one NP-complete problem, if it is in P, then necessarily P=NP. A graph is a way to mathematically model a network, like a system of subway tunnels, a series of tubes, or computers on the Internet. It consists of a collection of vertices or nodes, representing locations, computers, or objects, and edges showing the connections between them [2]. Formally, a graph G consists of a finite set V (G) of vertices, a finite set E(G) of edges, and a function f assigning to every edge e a set {u, v} of vertices, which we think of as the two endpoints of the edge. We say that the edge e is incident to u and v. We usually require that u and v be distinct, but this is not always necessary.

III. MAXIMAL INDEPENDENT SET

The problem of finding a Maximal Independent Set of a graph is important for applications in Computer Science, Operation Research, and Engineering. There are many applications of the MIS such as graph coloring, scheduling final exam in a university, assigning channels to radio station, register allocation in a compiler, and the reader collision problem [3].

An independent set in a graph G = (V, E) is a set $I \subseteq V$ such that there is no pair of nodes in I linked by an edge in E. The maximal independent set problem is to find an independent set that is not properly contained in any other independent set. This problem is NP-Hard and it is natural to ask for approximation algorithms. We mention some special cases of MIS that have been considered in the literature, this is by no means an exhaustive list.

- Interval graphs; these are intersection graphs of intervals on a line. An exact algorithm can be obtained via dynamic programming and one can solve more general versions via linear programming methods.
- Note that a maximum (weight) matching in a graph G can be viewed as a maximum (weight) independent set in the line-graph of G and can be solved exactly in polynomial time. This has been extended to what are known as claw-free graphs.
- Planar graphs and generalizations to bounded-genus graphs, and graphs that exclude a fixed minor. For such graphs one can obtain PTAS due to ideas originally from Brenda Baker.
- Geometric intersection graphs. For example, given n disks on the plane find a maximum number of disks that do not overlap. One could consider other (convex) shapes such as axis parallel rectangles, line segments, pseudo-disks etc.

IV. MULTICRITERIA APPROACH

Multiple-criteria are a sub-discipline of operations research that explicitly considers multiple criteria in decision-making environments. Multicriteria analysis problems several criteria are simultaneously optimized in a feasible set of a finite number of explicitly given alternatives.

One of the most important imperfections of such a definition is its' comparatively low selectivity: power of set of optimal solutions is often commensurable with a set of all feasible plans transforming it into a sampling problem [4]. In the general case it does not exist one alternative, which optimizes all the criteria. There is a subset of alternatives however, characterized by the following: each improvement in the value of one criterion leads to deterioration in the value of at least one other criterion.

This subset of alternatives is called a set of the non-dominated alternatives. Each alternative in this set could be a solution of the multicriteria problem. In order to select one alternative, additional information is necessary, supplied by the so-called decision maker (DM). The information that the DM gives, reflects his/her preferences with respect to the quality of the alternative sought [5].

Such are the problems of evaluation and choice of resources, strategies, projects, offers, policies, credits, products, innovations, designs, costs, profits, portfolios, etc.

A number of criteria are defined to evaluate the quality of the timetables from different points of view with respect to different constraints imposed on the problem.

In [5] that provide new multicriteria approach to solving such problems is presented. A number of criteria will be defined with respect to a number of exam timetabling constraints the criteria considered in these research concern room capacities, the proximity of the exams for the students, the order and locations of events, etc. Of course, the criteria have different levels of importance in different situations and for different institutions.

V. PROPOSED MODEL

Examination timetabling is an important problem in any educational institution. The quality of a solution is of great importance to a number of parties including lecturers, students, and administrators. Many variants of the problem have been studied since the seventies.

We are given a set of exams E = {e1, e2... en}, and we know for each exam the students who are enrolled. Hence, we can compute for each pair of exams whether they conflict, and if so, the size of the overlap, which we define as the number of students who are enrolled on both exams. There are two major variants in exam timetabling, depending on whether the available time slots are given. In our case, the sets of hard and soft constraints differ significantly from university to university. We introduce one hard constraint which is present in all exam timetabling problems:

- The exams which are in conflict (i.e. have common students) must not be scheduled in the same time period.
- Number of students in one period should not exceed the size of all classrooms. Criteria are defined with respect to the other constraints that are imposed on specific problems. Each criterion expresses a measure of the violation of the corresponding constraint. We introduce soft constraint as following:
- All exams have to be scheduled.
- Each student has at most two exams in same day.
- Minimize the number of student that has two exams in adjacent period.
- Minimize the number of day to achieve all exams.
- Each classroom has at least two exams in the same time.

According to preliminary computational experiments, these could be met without a large additional cost, and hence we consider these as hard constraints and minimize the soft constraint. Finally, we want to avoid as much as possible that students have two exams in adjacent period. It follows from the outcome of the problem of minimizing the number of time slots that there is no solution possible in which no student has two exams per day. Hence, we treat it as a soft constraint, and we include the number of students with two exams in adjacent period in the objective function.

We introduce the following notation:

N is the number of exams.

M is the number of student.

K is the number of student enrollment.

S is the size of classrooms in one period.

$C = [c_{nm}]$ NxN is the symmetrical matrix which represents the conflicts of exams.

cnm is the number of students taking both exam n and exam m where $n \in \{1... N\}$

And $m \in \{1... N\}$.

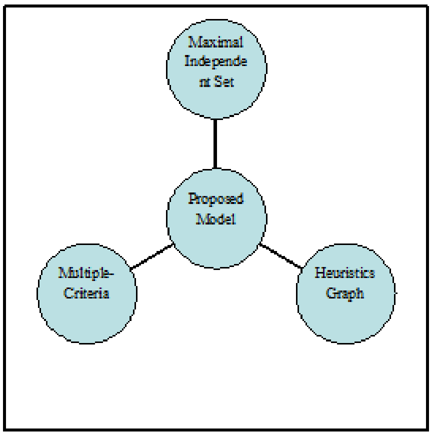
G= 0.75 * S (where the 0.75 is the percent of full the size of classrooms, and this percent we compute from the result from recent papers)

P = K / G is the number of period that can by achieve as 3 period in one day, and must by minimize the number of student has two exam in adjacent period.

And the remaining exam that not scheduling in previous periods P that scheduling as Q periods and that two periods in one day.

VI. NEW ALGORITHM

Our idea is combine of three technique (Maximal Independent Set, Multi Criteria and Heuristic Graph), as figure 4.1. The first technique Maximal Independent Set is used to get set of exams without conflict by used Greedy Algorithm that show in section 4.4.1, and this technique used in different part of main algorithm. The second technique Multicriteria are several criteria simultaneously optimized in a feasible set of a finite number of explicitly given alternatives, it used in second part of main algorithm in subroutine 2. The third technique in my proposed model is heuristics graph, The heuristic order the events based on an estimation of how difficult it is to schedule them as Largest degree first (Events that have a large number of conflicts with other events). Largest weighted degree (This is a modification of the largest degree first which weights each conflict by the number of students involved in the conflict). Saturation degree (In each step of the timetable construction an event which has the smallest number of valid periods available for scheduling in the timetable constructed so far is selected). Color degree (This heuristic prioritizes those events that have the largest number of conflicts with the events that have already been scheduled). Our proposed model includes 6 subroutines:



VII. GREEDY ALGORITHM

The simple greedy algorithm for finding a large independent set works as follows:

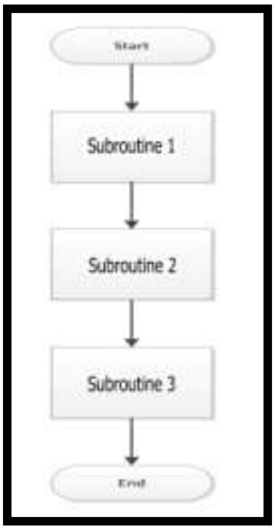
Greedy (u is node) {

1. Initialize I as I = {}.
2. I = I ∪ {u} and delete u and all neighbors of u from G.
3. Find a vertex u with the smallest possible number of neighbors in G. Set I = I ∪ {u} and delete u and all neighbors of u from G.
4. Repeat step 3 until there are no vertices left in G.
5. Return I is our large independent set in G.

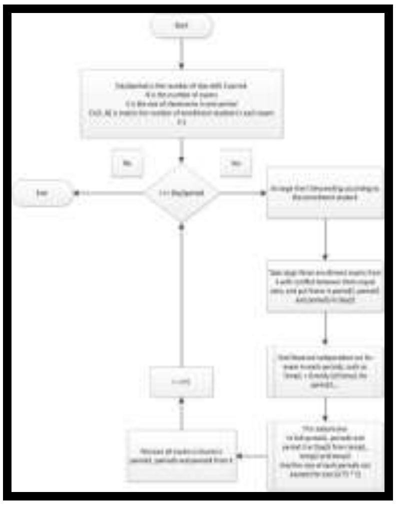
VIII. MAIN ALGORITHM

The Main algorithm is consist of two part, first part is to fill the P period where this period can put 3 period in one day and this part consist of two section, first section is to fill P period without students that have two exams in adjacent period, the second section is to minimize the number of courses that unscheduled by adding the unscheduled courses to the first section with minimum number of student that have two exams in adjacent period. The second part is to fill the Q period, where this period can by achieve two exams in one day, where the unscheduled courses in first part must schedule as two period in one day.

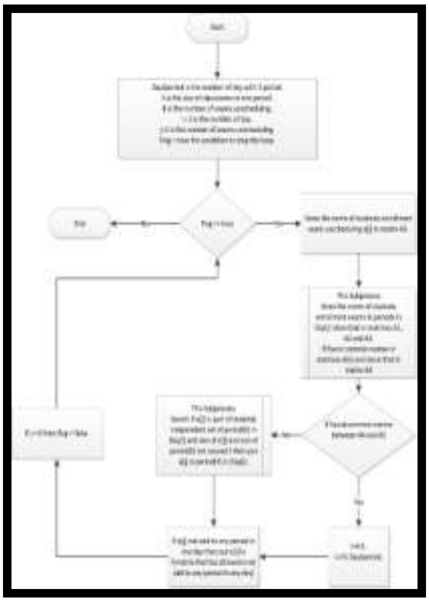
International Journal of Advance Research in Science and Engineering
Vol. No. 5, Special Issue No. 01 , May 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354



Subroutine 1

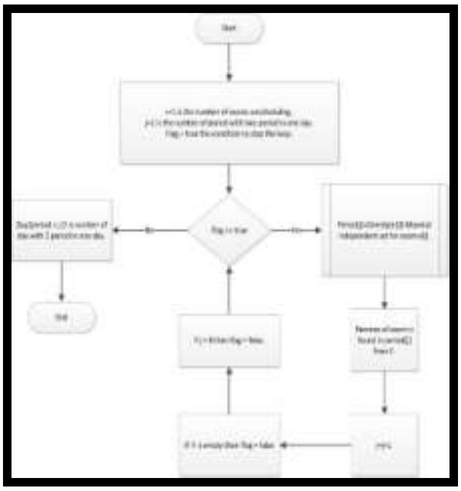


Subroutine 2



IJARSE
ISSN 2319 - 8354

Subroutine 3



IX. ALGORITHM FOR DISTRIBUTED STUDENTS ON CLASSROOMS

This algorithm is to distribute students in each period on available classrooms:

Begin

While (Flag = true)

- H = size of classroom[i]
- K1= H/2
- K2 = H/2
- m= size of exam[j1]
- m1=m-K1
- If (m1>0)

    - K1 students from exam[j1] in classroom[i]

    else

    - (-1*m1) students from exam[j1] in classroom[i]

- n= size of exam[j2]
- n1=n-K2
- If (n1>0)

    - K2 students from exam[j2] in classroom[i]

    else

International Journal of Advance Research in Science and Engineering
Vol. No. 5, Special Issue No. 01 , May 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

- (-1*n1) students from exam[j2] in classroom[i]

1. i=i+1

End

X. APPLICATION

In this section we give an example to describe how our algorithms solve the examination timetabling table as following:

Let suppose we have that

N =10 is the number of exams.

M = 230 is the number of student.

K = 600 is the number of student enrolled.

S = 100 is the size of classrooms in one period.

G= 0.75 * S = 0.75 * 100 =75

P = K / G =600/ 75 = 4 is the number of period that achieve as 3 period in one day. As following table:

Exam	Enrollment
E1	95
E2	87
E3	83
E4	82
E5	72
E6	61
E7	40
E8	34
E9	25
E10	21

Table .1 Show the exams and enrollment in example

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
E1	0	60	43	49	39	18	0	16	0	14
E2	60	0	0	0	37	13	14	0	0	0
E3	43	0	0	44	0	0	12	11	0	0
E4	49	0	44	0	0	0	11	16	0	16
E5	39	37	0	0	0	0	10	13	0	0
E6	18	13	0	0	0	0	0	0	3	0
E7	0	14	12	11	10	0	0	0	0	0

International Journal of Advance Research in Science and Engineering
Vol. No. 5, Special Issue No. 01 , May 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

E8	16	0	11	16	13	0	0	0	0	0
E9	0	0	0	0	0	3	0	0	0	0
E10	14	0	0	16	0	0	0	0	0	0

Table .2 Show the exams and their conflict

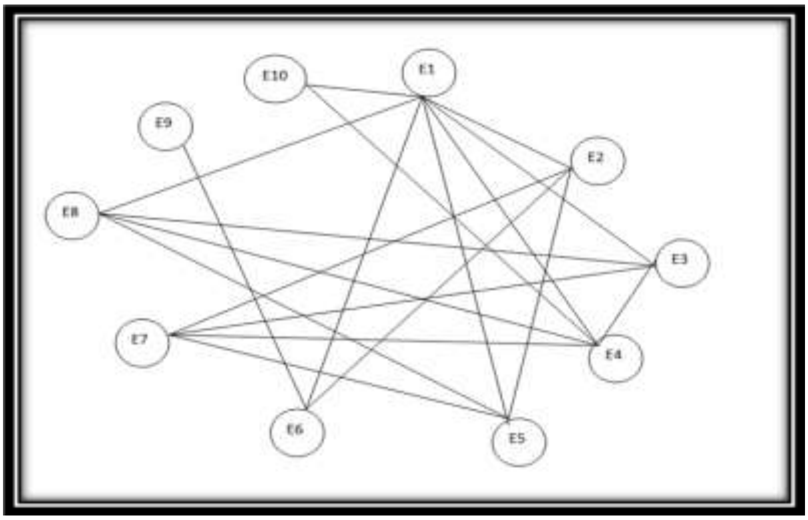


Figure 1 Graph that show conflict between Exams

The result from Algorithm 1 as following:

- period1 [E1]
- period2 [E7]
- period3 [E9]

The result from Algorithm 2 as following:

- period1 [E1]
- period2 [E7, E8, E10]
- period3 [E9, E6]

The result from Algorithm3 as following:

- period1 [E2]
- period2 [E3]
- period3 [E4]
- period4 [E5]

The timetabling of that example as following

Day 1
period1 [E1]
period2 [E7, E8, E10]
period3 [E9, E6]

Day 2
period1 [E2]
period2 [E3]

Day 3
period1 [E4]
period2 [E5]

XI. CONCLUSION

In this paper, we show the proposal model from definition and notation and algorithms and flowchart, and provide example that show how algorithms solve the problem. In the future work, we will apply our new algorithm to experimental data taken from some universities.

REFERENCE

[1] Malek Alzaqebah and Salwani Abdullah, "The Bees Algorithm for Examination Timetabling Problems", International Journal of Soft Computing and Engineering, 2011.

[2] Oluwasefunmi T. Arogundade, Adio T. Akinwale and Omotoyosi M. Aweda, "A Genetic Algorithm Approach for a Real-World University Examination Timetabling Problem", International Journal of Computer Applications, 2010.

[3] Tero Harju, “Lecture Notes on GRAPH THEORY” Turku, Finland, 2011.

[4] Can Berk Kalayci, Aşkıner Gungor, "A Genetic Algorithm Based Examination Timetabling Model Focusing on Student Success for the Case of the College of Engineering at Pamukkale University, Turkey", 2012.

[5] Ahmad Sharieh, Wagdi Al_Rawagepfeh, Mohammed H. Mahafzah and Ayman Al Dahamsheh,"An Algorithm for Finding Maximum Independent Set in a Graph", EuroJournals Publishing, 2008.