African Journal of Science and Technology (AJST)
Science and Engineering Series Vol. 5, No. 2, pp 34-40

MATHEMATICAL PROGRAMMING FORMULATIONS FOR THE EXAMINATIONS TIMETABLE PROBLEM: THE CASE OF THE UNIVERSITY OF DAR ES SALAAM

Mushi, A.R.
Department of Mathematics, University of Dar es salaam

ABSTRACT:- *Examinations Timetabling Problem (ETP) is the problem of assigning courses to be examined and candidates to time periods and examination rooms while satisfying a set of constraints. Every University has a different set of constraints and structure of examinations. Thus there is no general ETP model for all Universities around the world [1]. ETP is NP-Hard [2] and therefore no optimal algorithm is known for this problem which can solve a general problem within reasonable time. However, exact methods can be used to provide a benchmark for the heuristic methods. There is no general model for University Timetabling Problems because the problem feature differs from one University to another. In this paper we focus in the formulation of the ETP for the University of Dar as salaam. We formulate, test and compare three Integer Programming models. It is concluded that, although exact methods cannot give a solution to a real-size problem, these models give a good benchmark for testing the performance of other approaches. This paper also gives a direction for better exact models for the University of Dar es salaam's ETP.*

INTRODUCTION

The Examination-Timetabling Problem (ETP) essentially involves the assignment of courses to be examined and their candidates to time periods and examination rooms while satisfying a given set of constraints. These constraints are divided into hard and soft [i], [ii]. Hard constraints must be satisfied such as avoiding student-examination collision and room over-sizing. Soft constraints may be tolerable but must be minimized as much as possible, such as scheduling large candidate examinations first and minimizing the evening sessions. ETP is an NP-Hard problem [iii],[iv] and therefore no optimal algorithm is known which can give a solution within reasonable time. Despite of this fact, exact methods have been very useful to provide benchmarks for heuristic algorithms and in other cases have been used in combination with heuristic methods to give good solutions.

There are quite a lot of versions of the timetabling problem, differing from one University to another [iv]. A lot of work has been done on this type of problem basing their studies on specific Universities [v]. The first attempts were based on the graph coloring concept [vi]. In this case the vertices represent courses, and an arc joins two vertices only if they cannot be scheduled at the same time. The problem is therefore to find the chromatic number of the resulting graph. However, like the Timetabling Problem, the Chromatic Number problem is also NP-Hard [vii]. Due to the complexity of the ETP, most of the work done concentrates on the heuristic algorithms which try to find a good approximate solution. Some of these include Genetic Algorithms [viii]], Tabu Search [ix], Simulated Annealing [iii], and recently the use of scatter search methods [x]].

No mathematical models have been developed for the timetable problems at the University of Dar es salaam. In this paper, we focus our attention in formulating mathematical models for the examinations timetable at the University of Dar es salaam. This will act as a benchmark for testing heuristic algorithms and help future reformulations of the problem mathematical models.

The ETP differs considerably from the University course-scheduling problem. The following are some of these differences;

- An examination occupies only one slot throughout the planning period, while the course timetable may require several slots and with different categories such as lectures, tutorials and practical sessions.
- An examination may span several weeks while course timetable must fit within a week, which repeats for the whole semester.
- An examination room can occupy more than one examination while course schedules cannot.

Examinations timetable at the University of Dar es salaam

The examinations period is currently fixed to 2 weeks, with 3 examination sessions a day except on Fridays where we have 2 sessions. A week is made up of five consecutive days i.e. Monday to Friday. Some examinations have more candidates than a single room can hold, thus some examinations are scheduled in more than one room. On the other hand, a room can have more than one examination scheduled in it if sufficient room space is available.

To optimize examination space, lecture theatres are also used for examinations. However, a color scheme has been designed in such a way that, such a room can be occupied by four different examinations with each neighboring student doing a different examination.

This paper is based on the following assumptions;

- An examination can be scheduled in any of the available rooms.
- All examinations are three hours long. In practice, some examinations are 2½ hours and some are 3½ hours long but can be considered to fit within the 3 hours timeslots.
- No open book examinations. This case can be dealt with manually.
- Once assignment of courses to rooms and timeslots is done, a division of courses in lecture theatres into different groupings can be finished up manually.
- Maximum mixing of courses in a room is four, due to the four colors arrangement. However, small exams can be grouped to form the four parts. The exercise can be done manually after solving the problem.
- Walking distance between examination rooms is irrelevant, as there is an interval of at least one and half hours between examinations

Hard Constraints

- No candidate can be assigned to more than one examination at the same time
- A room cannot be assigned more candidates than its capacity
- Only rooms with standby generators can be used for the evening session of a day.

Soft Constraints

- Minimize room space wastage during examinations. Each room must be utilized as much as possible.
- We discourage continuous examinations for a candidate in a day. Since there are three sessions in a day, we prefer to have at least one gap between examinations per candidate in a day.
- We prefer the case where a lecturer has at least one gap between invigilation sessions.

Mathematical Programming Formulation

Many possible models exist for the solution of a particular problem. Different models might have different features including the size, number of non-zeros, characteristics of variables and most importantly closeness to the Integer Polytope. We present three models with different features and compare their performance.

Model 1: A single 0/1 variable

Let $x_{jkr} = \begin{cases} 1 & \text{If course j is assigned to time slot k in room r} \\ 0 & \text{Otherwise} \end{cases}$

Note that, the time slot *k* takes care of both time and day. The examinations take two weeks, Monday to Friday of each week, amounting to ten examination days in a semester. These are numbered in increasing order from Monday of first week to Friday of the second week. See Table 1 for illustration;

Table 1: Time Slot numbers

Week	Week 1								Week 2	
Day	Monday			Tuesday			…	…	Friday	
Session	1	2	3	1	2	3	…	…	1	2
Time slot (k)	1	2	3	4	5	6	…	…	27	28

Note that Fridays have only two examination sessions. This is to allow for Friday Muslim Prayers and Seventh Day Adventists who start their Sabbath day on Friday 18:00 hours in the evening. This gives a total of 28 timeslots for the whole examination period.

Constraints

To prepare a timetable, the following data needs to be supplied:

- Student registration details, where every student must register for courses that he/she will take before the beginning of the semester.

  o Let $s_{ij} = \begin{cases} 1 & \text{if student } i \text{ registered for course } j \\ 0 & \text{Otherwise} \end{cases}$

- Lecturers are assigned courses from their respective Departments. Important details here for the examination timetable are the lecturer identification and the courses assigned to each lecturer.

  o Let $l_{ij} = \begin{cases} 1 & \text{if lecturer } i \text{ is assigned to course } j \\ 0 & \text{Otherwise} \end{cases}$

- Examinations are scheduled into rooms whose examination capacity is known.
  - o Let R_r = Examination capacity of room *r*. The examination capacity of a room is normally less than the teaching capacity so as to disperse candidates. These capacities are estimated beforehand.
  - o Let G = Set of all indices of rooms in R which have standby generators
  - o Let E = Evening time slots i.e. E = {3, 6, 9, 12, 17, 20, 23, 26}.
- We assume that there are *n* courses to be scheduled in *m* rooms and *t* time slots.

The problem constraints are divided into hard and soft. The hard constraints must be satisfied and these are modeled as constraints of the problem. Soft constraints are to be minimized and these are modeled as the objective function of the problem.

- No student can be assigned to more than one examination session at a time. In other words, no student can be assigned to two courses j_1 and j_2 at the same time slot *k*.

1) $s_{ij_1} x_{j_1kr_1} + s_{ij_2} x_{j_2kr_2} \le 1 \quad k = 1, 2, ...30;$

$\forall r_1, \forall r_2, \forall i, \forall r_1, \forall j_1, j_2$ such that $j_1 \ne j_2$

- No examination can be assigned to a room which has less capacity than the course size. That is, courses assigned to time slot *k* at room *r*, must not exceed the room examination capacity.

2) $\sum_j \sum_i (s_{ij} x_{jkr}) \le R_r \quad k = 1, 2, ...30; r = 1, 2, ...m$

This says that, if any of the courses *j* = 1,2, …, *n* are assigned to time slot *k* at room *r*, then the sum of all students taking courses *j* = 1, 2, …, *n* must not exceed the room capacity.

- Courses cannot be scheduled in evening slots *p*, in a room without a standby generator, i.e.

  $x_{jpr} = 0 \quad \forall r \notin G, \ \forall j, \text{ and } p \in E.$

  Thus we have

  $$\sum_{j \in J, p \in E, r \notin G} x_{jpr} = 0, \quad where \ \mathrm{J} = \{1,2,...,\mathrm{n}\}$$

- A course must be scheduled exactly once

3) $\sum_k \sum_r x_{jkr} = 1 \quad \forall j$

Soft Constraints

As mentioned earlier, the soft constraints make up the objective function.

- We would like to minimize room space wastage during examinations. Each room *r* must be utilized as much as possible.

  i.e. minimize $z = \sum_r \sum_k \left(R_r - \sum_j \sum_i s_{ij} x_{jkr} \right)$

- Minimize the case where a student *i* can have consecutive examinations j_1 and j_2 in time slot *k* and *k+1* at any day.

  i.e. minimize $s_{ij_1} x_{j_1kr_1} + s_{ij_2} x_{j_2(k+1)r_2}$

  $\forall i, \forall k, \forall r_1, \forall r_2, \forall j_1, j_2$ such that $j_1 \ne j_2$

- Minimize the case where lecturer *l* may invigilate more than one consecutive examinations j_1, j_2 in time slots *k* and *k+1*.

  i.e. minimize $\left(l_{ij_1} x_{j_1kr_1} + l_{ij_2} x_{j_2(k+1)r_2} \right)$

  $\forall k, \forall i, j$ such that $i \ne j, \ \forall r_1, \forall r_2$

The Pure 0/1 Integer Programming model then looks as follows;
minimize

$$z = \sum_r \sum_k \left(R_r - \sum_j \sum_i s_{ij} x_{jkr} \right)$$

$$+ \sum_i \sum_{j_1 j_2} \sum_k \sum_{r_1 r_2} (s_{ij_1} x_{j_1kr_1} + s_{ij_2} x_{j_2(k+1)r_2})$$

$$+ \sum_i \sum_{j_1 j_2} \sum_k \sum_{r_1 r_2} (l_{ij_1} x_{j_1kr_1} + l_{ij_2} x_{j_2(k+1)r_2})$$

Subject to:

1) $s_{ij_1} x_{j_1kr_1} + s_{ij_2} x_{j_2kr_2} \le 1 \; k = 1, 2, ...30;$

$\forall r_1, \forall r_2, \forall i, \forall j_1, j_2,$ such that $j_1 \ne j_2,$

2) $\sum_j \sum_i (s_{ij} x_{jkr}) \le R_r \;\; k = 1, 2, ...30; r = 1, 2, ...m$

3) $\sum_{j \in J, p \in E, r \notin G} x_{jpr} = 0, \;\; where \; \mathrm{J} = \{1,2,...,\mathrm{n}\}$

4) $\sum_k \sum_r x_{jkr} = 1 \; \forall j \;\; z \ge 0, x \in \{0,1\}$

Given a timetable with *n* courses, *m* rooms, and *t* timeslots, we have a problem with *ntm* variables. A typical examinations timetable at UDSM involves 2,400 courses, 100 rooms, 5,000 students and 700 Lecturers on a 28 slots time interval. This gives a problem with *ntm* = 2400x28x100 = 7,200,000 variables. Obviously the problem is too big to solve by any available Mathematical Programming software.

We attempt to minimize the number of variables by breaking the variable definition into two separate variables. This gives the second model as presented below;

Model 2: Two 0/1 Variables

Let $x_{jk} = \begin{cases} 1 & \text{If course j is assigned to time slot k} \\ 0 & \text{Otherwise} \end{cases}$

Let $y_{jr} = \begin{cases} 1 & \text{If course j is assigned to room r} \\ 0 & \text{Otherwise} \end{cases}$

Hard Constraints

No student *i* can have more than one courses j_1, j_2 scheduled on the same time slot *k*.

$s_{ij_1} x_{j_1k} + s_{ij_2} x_{j_2k} \le 1 \;\; k = 1, 2, ...28;$

$\forall i, \forall j_1, j_2$ such that $j_1 < j_2$

No room *r* can have more students *i* than its capacity R_r.

$\sum_j \sum_i (s_{ij} y_{jr}) \le R_r \quad r = 1, 2...m$

Evening slots cannot be scheduled in rooms which have no standby generator;

$\sum_{j \in J, p \in E, r \notin G} (x_{jp} + y_{jr}) = 0, \;\; J = \{1, 2, ...n\}$

An examination *j* must be scheduled in exactly one slot *k*.

$\sum_k x_{jk} = 1 \qquad \forall j$

Soft Constraints

We discourage the case where a lecturer *i* can invigilate consecutive examinations j_1, j_2; we prefer to have at least one gap [*k*, *k+1*] of an examination slot between.

i.e. Minimize $l_{ij_1} x_{j_1k} + l_{ij_2} x_{j_2k+1} \; \forall k, \forall i, \forall j_1, j_2$ such that $j_1 < j_2$.

Minimize the case where a student *i* can have two or more consecutive examinations j_1, j_2.

i.e. $s_{ij_1} x_{j_1k} + s_{ij_2} x_{j_2k+1} \; \forall k, \forall i, \forall j_1, j_2$ such that $j_1 < j_2$.

Minimize room wastage; i.e. minimize

$R_r - \sum_j \sum_i (s_{ij} y_{jr}) \quad r = 1, 2, ...m$

Objective function

Minimize $z = \sum_r \left(R_r - \sum_j \sum_i s_{ij} y_{jr} \right)$

$+ \sum_i \sum_k \sum_{j_1 j_2} (s_{ij_1} x_{j_1k} + s_{ij_2} x_{j_2k+1})$

$+ \sum_i \sum_k \sum_{j_1 j_2} (l_{ij_1} x_{j_1k} + l_{ij_2} x_{j_2k+1})$

In this model we have *n*(*t*+*m*) = 2400(28 + 100) = 312,000 variables. This is a considerable reduction from the previous model. However, the number of variables is still too high to be able to solve a real timetable problem. The number of variables can be further reduced if we minimize the number of 0/1 variables. Let us consider the case of mixed pure integer and 0/1 variables as shown in model 3.

Model 3

Let T_c = Time slot in which course *c* is slotted

C_i = Room which is assigned to course *i*

$G_{si} = \begin{cases} 1 & \text{If student } s \text{ takes course } i \\ 0 & \text{Otherwise} \end{cases}$

$A_{li} = \begin{cases} 1 & \text{If lecturer } l \text{ is assigned course } i \\ 0 & \text{Otherwise} \end{cases}$

R_i = Capacity of room *i*

Objective function

- Minimize the room space wastage;

minimize $\sum_{r}(R_r - \sum_{i \ni c_i = r} \sum_{s} G_{si})$

- Consecutive examinations;
- We would like the gap between two examinations for a student to be at least 2.

i.e. $\left| T_i G_{si} - T_j G_{sj} \right| \succ 2 \quad \forall s, \ \forall i, j \ni i \prec j$

maximizing the gap is the same as:

minimize $\left(2 - \pm\left(T_i G_{si} - T_j G_{sj}\right)\right) \forall s, \quad \forall i, j \ni i \prec j$

Since we are minimizing, we can ignore the negative and consider only the positive component to get;

Minimize $\left(2 - \left(T_i G_{si} - T_j G_{sj}\right)\right) \forall s, \quad \forall i, j \ni i \prec j$

Thus the objective component becomes:

Minimize $\sum_{s} \sum_{i \prec j} \left(2 - \left(T_i G_{si} - T_j G_{sj}\right)\right)$

Constraints

- Student collision

Courses *i* and *j* registered by the same student *s* cannot be scheduled on the same slot;

$T_i G_{si} \neq T_j G_{sj} \quad \forall s, \ \forall i, j \ \ni i \prec j$

- Room size;

The count of all students *s* registered for courses *i* which have been slotted in room *r* must not exceed the capacity of that room.

- $\sum_{i \ni C_i = r} \sum_{s} G_{si} \leq R_r \quad \forall r$ (i)

Since C_i is a variable, we need to remove it from the bounds. We introduce a 0/1

variable $\delta_{ir} = \begin{cases} 1 & \text{if } C_i = r \\ 0 & \text{Otherwise} \end{cases}$

Then (i) can be written as follows:

$\sum_{i} \sum_{s} \delta_{ir} G_{si} \leq R_r \quad \forall r$

To enforce the 0/1 variable, we introduce the following constraint [xiii];

$C_i - \delta_{ir} \prec r \quad \forall i, \ \forall r$

Given a course slotted in room r (i.e. C_r), where r has no standby generator, then the time in which this course is slotted cannot be a multiple of 3 (i.e. cannot be evening hours).

Thus $T_{C_r} \neq p, \ p \in E, \ r \notin G$

Model 3 then looks as follows;

Minimize $\sum_{r}(R_r - \sum_{i} \sum_{s} \delta_{ir} G_{si})$

$+ \sum_{s} \sum_{i \prec j} \left(2 - \left(T_i G_{si} - T_j G_{sj}\right)\right)$

Subject to:

1. $T_i G_{si} - T_j G_{sj} \neq 0 \quad \forall s, \ \forall i, j \ \ni i \prec j$
2. $\sum_{i} \sum_{s} \delta_{ir} G_{si} \leq R_r \quad \forall r$
3. $C_i - \delta_{ir} \prec r \quad \forall i, \ \forall r$
4. $T_{C_r} \neq p, \ p \in E, \ r \notin G$
5. $\delta \in \{0,1\}, T, C$ integers

There are three types of variables in this case;

T, C, and δ giving a total of $t + n + nm$ = 28+2,400 +2,400x100 = 242,428 variables. This is a better reduction in the number of variables. However, the number of variables is still too large to be solved by the available mathematical programming packages. This is expected, since the problem is NP-Hard. As pointed earlier, our interest is to find small instances of test problems which can be solved to optimality and play the role of benchmarks for testing the performance of heuristic procedures. We present a summary of results for data obtained from four departments in the faculty of science.

Summary of results

We use test data from departments of Computer Science, Mathematics, Physics and Chemistry. The results are as summarized;

Table 2: Results summary for model 1

Model 1				
Department	**Variables**	**Constraints**	**Iterations**	**Time (Sec)**
Computer Science	448	113,012	316	27.41
Mathematics	336	72,664	270	24.5
Physics	336	84,760	123	20.03
Chemistry	448	161,396	301	28.09

Table 3: Results summary for model 2

Model 2				
Department	**Variables**	**Constraints**	**Iterations**	**Time (Sec)**
Computer Science	128	7,064	217	25.44
Mathematics	124	8,071	117	16.74
Physics	124	9,415	157	19.02
Chemistry	128	10,088	362	72.31

Table 4: Results summary for model 3

Model 3				
Department	**Variables**	**Constraints**	**Iterations**	**Time (Sec)**
Computer Science	48	272	102	12.44
Mathematics	44	303	98	10
Physics	44	351	53	11.49
Chemistry	48	380	68	11

The data set used was significantly small due to the explosion of problem size as we increase the size of the problem. The number of constraints grows exponentially.

In model 3, the solution was found after less iteration than the other two models. The number of variables and constraints is significantly lower. However, the solution times are not directly proportional to the size of the problem. Chemistry data set from model 2 took longer than computer science data set on the same model, despite of the fact that the two departments have the same number of variables. This is due to the fact that the performance of the algorithm also depends on other factors such as the deeper the constraints are towards the optimal polytope. Using polyhedral combinatorics, it would be interesting to find the facets (deepest cuts) associated with such a model. This will trim down infeasibilities and make it easy for the branch and bound procedure to finish up the remaining problem. Such an approach might bring down the size of the problem that could be solved by the mathematical programming procedures. This has been applied with success to several Linear Programming models. See Grötschel et al [xiv] for the application of Polyhedral combinatorics to the Linear Ordering Problem. Generally, the three models give the same solution, but model 3 is smaller in size and could be used to solve bigger problem sizes. We don't expect model 3 to solve real-size problem either, but it gives a set of optimal solutions which can be used to test the performance of heuristics.

CONCLUSION

Many possible models exist for the same timetabling problem. In this particular case, we have demonstrated the existence of various models for the timetabling problem. The quality of a model depends on the closeness of the problem to the integer polytope. Reformulation and preprocessing can be used to reduce the problem even further without jeopardizing the optimality of the solution [xv].

Since this is an NP-Hard problem, we don't expect to get a mathematical programming formulation that will give us an optimal solution to the general problem. However, efforts in this direction can help to give a good benchmark for the heuristic approaches. Small-scaled instances can be used to compare with solutions obtained from heuristic approaches and thereby give a clue about the quality of these heuristics. We have been able to develop and test three models which can be used for benchmarking purposes. As observed, model 3, which is a mixed pure and 0/1 variables, performs better than the other two models.

REFERENCES

[i] De Werra, D (1985):"*An introduction to Timetabling*", European Journal of Operations Research, 19, 151-162.

[ii] Tim B. Cooper & J. H. Kingston (1996): "*The Complexity of Timetable Construction Problems*", Springer Lecture Notes in Computer Science 1153, pages 283-295.

[iii] Saleh M. A., Coddington P. (1998):"*A Comparison of Annealing techniques for Academic Course Scheduling*", Lecture Notes in Computer Science, vol. 1408, 92-114.

[iv] Slim A & Michael Marte (1998):"*University Timetabling using Constraint Handling Rules*", Journees Phrancophones de Programmation, par contraintes, Nates, France

[v] D. de Were (1985):"*An Introduction to Timetabling*", European Journal of Operations Research, 19, 151-162.

[vi] Iain Phillips (2000):"*Department of Computing Examination Timetable Problem*", Imperial College, http://citeseer.ist.edu/291030.html

[vii] A. Schaef (1995):"*A survey of Automated Timetabling*", Artificial Intelligence Review, *13(2), 87-127*.

[viii] Welsh D.J.A., Powell M.B. (1967):"*An Upper Bound for the Chromatic Number of a Graph and Its Applications to Timetabling Problems*", Comp. Journal. 10, 85-86.

[ix] Karp K.M. (1972): "*Reducibility among Combinatorial Problems*", In Complexity of Computer Computations, Plenum Press, New York.

[x] Corne D., Fang H.S., Mellish C., (1993*):"Solving the Modular Exam Scheduling Problem with Genetic Algorithms*", Proceedings of the sixth International Conference of Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Edinburgh.

[xi] Mooney L. M. (1991):"*Tabu Search Heuristics for Resource Scheduling with Course Scheduling Applications*", PhD Dissertation, Purdue University, 179 pages.

[xii] Marti R., Lourenco H., Laguna M. (2001):"*Assigning Proctors to Exams with Scatter Search*", Economics Working Paper Series No 534, Department of Economics & Business, Universitat Pompeu Fabr

[xiii] Williams, H. P. (1997):"*Model Building in Mathematical Programming*", John Wiley & Sons, New York

[xiv] Grotschel M. Junger M. & Reinelt G. (1983):"*A Cutting Plane Algorithm for the Linear Ordering Problem*", European Journal of Operations Research, 6(1984), 1195-1220.

[xv] Crowder, H, Johnson,E.L and Padberg, M (1982): "*Solving Large-Scale Zero-One Linear Programming Problems*", Operations Research 31, No. 5 (1983), 803 – 834