A Hungarian Based Algorithm for the Academic Scheduling Problem

Asma M A Bahurmoz
bahurmoz@kaau.edu.sa
King Abdul Aziz University
Jeddah, Saudi Arabia

Every educational institute faces the general problem of scheduling faculty members to courses, to classrooms and to time slots, and repeatedly every academic period. Its purpose is to satisfy and optimize the following constraints:

1- All scheduled sections (will be referred to as courses thereafter) must be assigned.

2- A faculty member cannot be assigned more than one course per time slot.

3- A room cannot be assigned to more than one course per time slot.

4- Consistency with respect to students must be observed.

5- Room assignment must insure course requirement (e.g. lecture halls, labs, etc).

6- Room assignment must insure efficient space utilization.

7- Faculty preferences for certain time slots and rooms must be satisfied.

8- Other administrative and educational requirements.

The first five constraints are known in the literature as hard ones and must be satisfied to obtain a feasible time table. The rest are known as soft and are usually incorporated in the objective function to be optimized as much as possible.

The academic scheduling problem (ASP) is well known in the literature, and it has been proven to be NP-Complete [9]. However, certain versions of the problem can be solved easily [4]. The problems' complexity is due to its large size, variety of variables and constraints, which could differ from one college to another within the same university.

The ASP has been of challenging nature to both operations research and computer science experts. Several comprehensive bibliographies and surveys on the practice and theory of timetabling were published during the last twenty five years [8, 6, 5, 7, 2). Most early research cited in the literature has concentrated on developing heuristic algorithms based on mathematical programming [3]. Modern heuristic algorithms, known as metaheuristic, based on

evolutionary algorithms have been successfully applied; Several examples can be found in [1] but only few of these algorithms in spite of their theoretical appeal are implemented. Carter ET. al.,[5] concluded "we were somewhat surprised to discover that there are very few course timetabling papers that actually report that the methods have been implemented and used at an institution".

Except for few educational institutes worldwide one can say that most academic institutions solve the problem manually. Such solutions are manpower and time consuming. Solutions are characterized by poor quality in terms of resource utilization and faculty satisfaction. Nevertheless, with more research addressing the scheduling problem in general, more advances in information technology, and hybrids of methodologies employed it is expected to see breakthroughs in the quality and quantity of automated scheduling.

This research addresses the design and implementation of a decision support system (DSS) for the ASP. A Hungarian based heuristic algorithm is designed to solve the problem in phases. It has been incorporated into a friendly interface not only to facilitate its usage but to allow the user to interact with the program and change parameters to secure feasibility if necessary, improve the quality of the resulting time table and increase its flexibility in responding to sudden changes in resources.

The decision support system's strategy is to fulfill professors' preferences for time slot assignments to courses. It consists of two phases:

Phase 1: A Hungarian based algorithm:

The first three sets of constraints with the seventh set incorporated in the objective function represent a 3-dimension assignment problem and by excluding the third set of constraints the problem is reduced further to a two-dimension assignment problem; courses to be assigned to time slots. The cost matrix consists of professor's preferences for time slots for each course they are teaching. Hungarian algorithm will iterate several times until a feasible schedule is obtained where all courses are assigned into time slots and required rooms insuring feasible choice for students to be able to attend all required courses from each. After every iteration the resulted assignments are checked for time slots overlapping and conflict of courses in terms of students' schedules (two courses must not be scheduled in the same time where a student must attend both). Then a compatible room, in terms of type and size, will be assigned to that course. Such courses will be labeled as assigned and will be removed from the unassigned courses list whose cost matrix is reset and resolved by the Hungarian algorithm and so on. So far the fourth, fifth and sixth sets of the above mentioned constraints are satisfied. The algorithm is stopped either when all courses are assigned or it reaches a certain number of iterations. The two sets of side

constraints that represent time overlapping and students grouping spoiled the beauty and simplicity of the Hungarian and drive the algorithm after a set of iterations into a loop. This calls for a second phase via a GUI.

Phase 2: Graphical User Interface:

The interface will prompt the user with the list of courses that have not been assigned. The user can retrieve information on professor availability and his/her preferences, students, and rooms available at these preferred time slots so he can select the most appropriate assignment and moves on to the second course. By examining the outcome a third phase is needed.

Phase 3: Further improvements:

The resulted solution is checked for maintaining a consistent schedule with respect to students' schedules and efficient use of resources. Further changes might be required to satisfy administrative regulations (the eighth set of constraints) such as a faculty member must be available for a department meeting at a given time every week. And some times just to increase faculty satisfaction by changing a room or time slot.

Computational experience:

The DSS has been rigorously tested on real-world instances provided by four faculties at king Abdulaziz University in addition to a small private college (Effat College). These instances represent: data of four consecutive semesters (Jan 2003- Sep. 2004). One of the faculties has implemented it successfully for the current semester 2004 and it will be repeated for the coming one. The others are considering it as well.

Quality of the resulting time tables indicates high degree of faculty satisfaction and more effective space utilization, not to mention saving on academic and administrators' time who are involved in generating the time table. However, some draw backs are noticed which require more improvements:

1- The fact that the algorithm is based on faculty priorities made it difficult to collect their priorities every semester and entering it.

2- Faculties who have many labs and work longer days needed some adjusting to the algorithm to accommodate certain issues such as two teachers can teach one course or a class must attend one lecture but divided into two sections when attending labs.

The author is currently testing the algorithm to produce the time table for the fall semester 2004 and comparing it against the manual one. Promising results are expected and detailed results will be presented at the conference.

References

1. Burke, E, Erben W. (Eds.): Practice and Theory of Automated Timetabling III (PATAT 2000, Konstanz, Germany, August, selected papers). Lecture Notes in computer Science, Vol. 2079. Springer –Verlag, Berlin Heidelberg New York (2001).
2. Burke, E., Petrovic, S.: Recent Research Directions in Automated Timetabling. Eur. J. Oper.Res. 140 (2002) 266-280.
3. Carter, M.W.: A Lagrangian Relaxation Approach to the Classroom Assignment Problem. IFOR 27 (1986) 230-246.
4. Carter and Tovey: When is the classroom Assignment Problem Hard? Operations Research, vol. 40 (1992) s28-s39.
5. Carter, W.M., Laporte, G.: Recent Developments in Practical Course Timetabling. In: Burke, E, Carter M. (Eds.): Practice and Theory of Automated Timetabling II (PATAT 1997, Toronto, Canada, August, selected papers). Lecture Notes in Computer Science, Vol.1408. Springer-Verlag, Berlin Heidelberg New York (1998) 3-19.
6. Kingston, J. H. (eds.): Bibliography on practice and theory of automated time-tabling. (1995) http://liinwww.ira.uka.de/bibliography/Misc/timetabling.html.
7. Schaef, A.: A survey of Automated Timetabling. Artif. Intell. Rev. 13 (1999) 87-127.
8. Schmidt, G., and Strohlein, T.: Timetable Construction – An annotated bibliography. Computer Journal, Vol. 23 (1980) 307-316.
9. De Werra: An Introduction to Time-tabling. European Journal of Operational Research, Vol. 19 (1985) 151-162.