A Multi-phase Approach to the University Course Timetabling Problem ⋆

Shahadat Hossain * Minhaz F. Zibran

Department of Mathematics and Computer Science
University of Lethbridge, Canada.

Key words: Constraint Programming, Course Scheduling, Graphical User Interface.

1 Introduction

Course timetabling is a well-known combinatorial problem that is needed to be solved regularly at educational institutions. There are a number of general constraints applicable to course timetabling problem in any organization. Depending on the institution, additional constraints and preferences may also apply.

This paper presents a multi-phase implementation of the timetabling problem using constraint programming technology. The software implementation incorporates a commercial solver ILOG's CPLEX, Java technology for graphical user interface, and C++ for piecing together of software components.

We solve the problem in a sequence of four phases. Phase-1 assigns instructors to courses (lectures, labs and tutorials), phase-2 assigns lectures to days, phase-3 assigns lectures to time-slots and finally phase-4 assigns labs and tutorials to days and available time-slots in the days. Each phase is solved using constraint programming with suitable heuristics for ordering the decision variables and maximizes an objective function over a given set of constraints and preferences. A useful feature of our implementation is that it allows the user to customize constraints as well as to generate new solutions that may incorporate partial solutions from perviously generated feasible solutions.

⋆ This research was supported in part by NSERC discovery grant.

* Corresponding Author.
Email addresses: shahadat.hossain@uleth.ca (Shahadat Hossain), minhaz.zibran@uleth.ca (Minhaz F. Zibran).

2 The Problem and Solution Approach

Timetabling problems have been traditionally solved using greedy heuristics, meta-heuristics such as hill climbing and tabu search, integer linear programming[1], and more recently using constraint programming techniques[2]. Cambazard et al.[3] proposed an algorithm for solving course timetabling problem using constraint programming that allows automatic relaxations of constraints. Lotfi, Vahid and Cerveny[4] introduced a multi-phase algorithm to solve a final exam scheduling problem. In our work we propose a multi-phase algorithm to solve the course timetabling problem using constraint programming. In what follows we use the following terminologies to describe the problem and solution strategies.

Topic: A topic is a set of related lectures, labs and tutorials. For example, "CS1000: Computer Basics" is a topic, which may include two lectures, three labs and two tutorials.

Course: A course refers to a single lecture or lab or tutorial.

Instructor: An instructor is a person who teaches/conducts a course. There may be different types of instructors, such as Professors, Academic Assistants and Graduate Assistants (Teaching Assistants).

Week-day: A week-day is a working day of a week from Monday through Friday.

Day-Sequence: A day-sequence is a set of week-days. Week-days are divided into two day-sequences: MWF (Monday, Wednesday and Friday) and TR (Tuesday and Thursday).

Time-slot: A time-slot refers to the unit of time-span specified by starting and ending time. Each week-day is divided into a fixed number of time-slots. Each time-slot may be allotted to courses to be taught during that period. Duration of 3 time-slots in MWF equals to the duration of 2 time-slots in TR.

One year prior to the start of an academic semester, each academic department at the University of Lethbridge determines the courses to be offered in that semester. Instructors mention which courses they would like to teach with a numerical preference level for each course. Instructors also mention the day or day-sequence and time of day (morning or afternoon) they prefer to teach. The department then prepares a timetable assigning courses to instructors, days or day-sequences and time-slots respecting the individual preferences as much as possible. The constraints and preferences specific to our problem include the following.

- Professors conduct only lectures, academic assistants conduct labs and tutorials, and graduate students conduct only labs. Each instructor has an upper limit on the number of courses the instructor may teach.

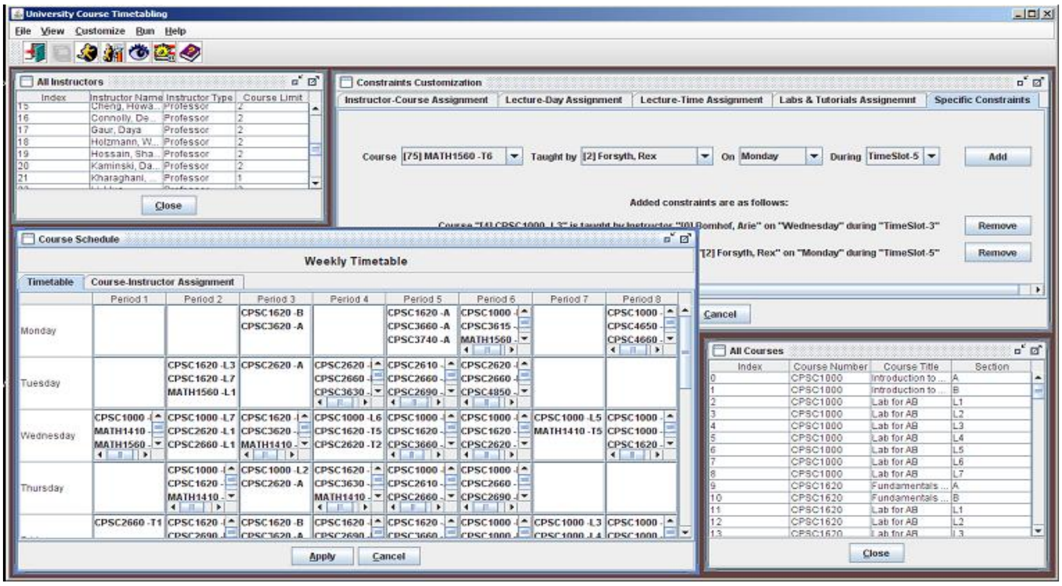

Fig. 1. A snapshot of user interface of the timetabling implementation.

- A lecture has to be scheduled in one of the two day sequences. When a lecture is assigned to a day-sequence, it is taught at the same time in each day of the day sequence.
- A lab or tutorial has to be assigned to only one of the five week-days.
- Lectures and associated labs and tutorials of a single topic have to be scheduled in such a way that for each lecture at least one lab and tutorial are available in non-overlapping time-slots.
- There should be a gap of at least one time-slot between courses taught by the same instructor.
- The preference of instructors on course, day, and time should be satisfied as much as possible.

We decompose the course scheduling problem into four phases and each phase is solved separately using constraint programming solvers. In phase-1, courses (lectures, labs, and tutorials) are assigned to the instructors (professors, academic assistants and graduate students). phase-2 assigns lectures to one of the two day-sequences. In phase-3, lectures of a single day-sequence are assigned to the time-slots available on that day-sequence. And finally phase-4 assigns labs and tutorials to week-days and available time-slots within the week-days. At each phase, the objective is to maximize the preferences (given as an objective function) subject to the given constraints. One of the objectives of this work is to allow the users enough flexibility so that customized schedules can be produced in a user-friendly way. The customization allowed in the current implementation includes addition or removal of constraints on the fly, loading and modifying a previously saved solution, and computing a new solution from a partial solution. In such cases, the new solution is attained quickly as it is not necessary to solve the problem from scratch[5].

3 Experiments and Evaluation

We experimented with our timetabling package in Windows XP environment on a AMD Athlon(tm) 64 processor 3500+ with 512 MB RAM. Our test data included 120 courses (35 lectures and 85 are labs and tutorials), 34 instructors (22 Professors, 9 Academic Assistants and 3 Graduate Students), 20 classrooms, 8 time-slots in each day of MWF and 6 time-slots in each day of TR. At each run on the data a feasible solution for each phase was obtained in less than two minutes.

4 Concluding Remarks

Course timetabling problem is a constraint satisfaction optimization problem. Therefore, constraint programming techniques are well-suited for solving such problems, as it exploits the structure of the problem. Multi-phase approach divides the complexity of the original problem into several subproblems of reduced complexity. Since the phases are solved separately, partial solutions may be generated, examined and amended. New solutions can be computed quickly from the partial solutions. Furthermore, the graphical user interface (see Figure 1) and the provision for dynamically adding or removing constraints make our timetable package flexible and user-friendly.

References

[1] MirHassani, S.A. A Computational Approach to Enhancing Course Timetabling with Integer Programming. Applied Mathematics and Computation, 179(2): 814 – 822, 2006.

[2] Hentenryck, Pascal Van and Saraswat, Vijay. Constraint Programming: Strategic Directions. Constraints: An International Journal, Vol. 2, No. 1, pp. 7 – 33 (1997). Kluwer Academic Publishers.

[3] Cambazard, Hadrein and et al. Interactively Solving School Timetabling Problems Using Extensions of Constraint Programming. E. Burke and M. Trick (Eds.): PATAT 2004, LNCS 3616, pp. 190 – 207, 2005.

[4] Lotfi, Vahid and Cerveny, Robert. A final Exam Scheduling Package. J. Opl. Res. Soc. Vol. 42, No. 3, pp. 205 – 216, 1991.

[5] T. Müller, R. Bartàk, H. Rudovà. Minimal Perturbation Problem in Course Timetabling. Practice and Theory of Automated Timetabling, Selected Revised Papers, pp. 126 – 146. Springer-Verlag LNCS 3616, 2005.