Artif Intell Rev (2013) 39:133–149
DOI 10.1007/s10462-011-9262-6

Solution approaches to the course timetabling problem

S. A. MirHassani · F. Habibi

Published online: 29 May 2011

Abstract University course timetabling is one of the most important administrative activities that take place in all academic institutions. In this work, we go over the main points of recent papers on the timetabling problem. We concentrate on university timetabling and introduce hard and soft constraints as well as most currently used objective functions. We also discuss some solution methods that have been applied by researchers. Finally, we raise more questions to be explored in future studies. We hope the directions lead to new researches that cover all aspects of the problem and result in high-quality timetables.

Keywords Course timetabling · Hard and soft constraint · Combinatorial optimization · Heuristic

1 Introduction

Timetabling problems have attracted the attention of the scientific community from a number of disciplines (including (OR) and Artificial Intelligence (AI)) since the 1960s for about 45 years and over the last decade; there has been an increased interest in the field. The timetabling problem is defined as "allocation, subject to constraint, of given resources to objects being placed in space-time, in such a way to satisfy a set of desirable objectives" (Luis et al. 2009). Timetabling problems arise in various forms including educational timetabling, nurse scheduling, sports timetabling, transportation timetabling and entertainment timetabling. It is one of the most important and time-consuming tasks occurring periodically (i.e. annually, quarterly, etc) in all academic institutions. Yet in many real-world situations, particularly where resources (such as people, space or time) are not abundant, the problem of constructing workable and memorable timetables can be a very challenging one, even for the experienced timetable designer. However, timetables will often have a large effect on the everyday lives of the people who use them. Timetable creation is certainly a problem that

S. A. MirHassani (✉) · F. Habibi
Faculty of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran
e-mail: a_mirhassani@aut.ac.ir

we should try to solve in the best possible fashion. Moreover, a timetable will often need to be updated or completely remade (e.g. school timetables will often be redesigned at the beginning of each academic year, bus timetables will need to be modified to accord with new road layouts and bus stops, etc.). Preparing timetables is also a problem that people will have to face regularly. Among instances of the multifarious problem of timetabling, educational timetabling is one of the most widely studied one. An educational timetabling problem can be defined to be the problem of assigning a number of events (course, exams, etc) or activities into a limited number of time periods (time-slots) in such a way to satisfy almost all constraints. In computing terms, timetabling problems are often modeled as Combinatorial Optimization Problems (COP). The overall objective in a COP is to find an assignment of discrete values to variables (e.g. timeslots for each event which needs to be timetabled) so that the solution be optimal according to some criteria. In other words, the problem is to find the best possible solution from among all possible solutions. The manual solution of the timetabling problem usually requires several days of work. In addition, the solution obtained may be not satisfactory in some respect. Thus, a considerable attention has been devoted to automated time tabling. Many papers related to automate timetabling have been published in conferences proceedings and journals. In addition, several applications have been developed and employed which have been quite successful. For example see Burke et al. (1994a,b, 1997, 2003, 2004, 2010), Newall et al. (1996b), Burke and Ross (1996), Carter and Burke (1998), Burke and Petrovice (2002), Carter (1986a), Carter (1986b), Carter et al. (1996), Carter and Laporte (1998), Carter and Laporte (1996a), Birbas et al. (1977) and Daskalaki et al. (2004). Also a extensive review of timetabling problems is done by Carter and Laporte (1996a, 1998) and Schaerf (1999). Carter and Laporte (1998), reorganized the approaches from 1986 to 1996. Later Burke and Petrovice (2002) review the latest works and research directions about timetabling problem until 2002. We are going to complete their work from 2002 to present. Burke et al. (2004), offered a definition of general timetabling, which covers many cases. They state: “A timetabling problem is a problem with four parameters: T, a finite set of times; R, a finite set of resources; M, a finite set of meetings; and C, a finite set of constraints. The problem is to assign times and resources to the meetings so as to satisfy the constraints as much as possible.

The rest of this paper is organized as follows: Sect. 2 describes the timetabling problem in general and concentrates on university course timetables. In Sect. 3 we review modeling based methods that have been used by researchers so far. Section 4 describes the heuristic methods. Directions for future work and a list of new ideas comprise Sect. 5. Finally, conclusions are given in Sect. 6.

Some of the most recent papers have been included below. But undoubtedly, there are many that we have omitted. Despite these omissions, we believe that the bibliography below will be a valuable summary of recent research in this area.

2 Problem description

Educational timetabling problems need to be solved in schools, colleges and universities regularly. There are some similarities between these problems, for example, teacher/lecturers can only teach one group of students at a time and rooms have size limitations. However, there are also some key differences. Overall different educational timetabling problems can be divided into two mains categories:

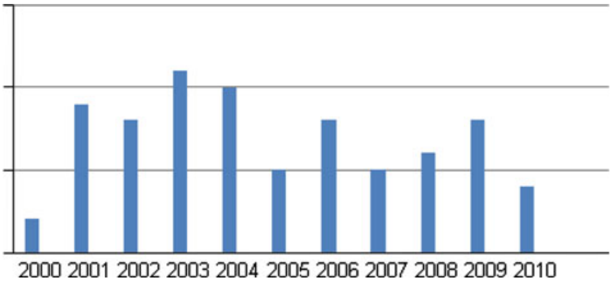

Fig. 1 No. of works on university course timetabling problem (2000–2010)

- *Exam timetabling*: most educational institutions need to schedule a set of examinations at the end of each semester or year. In its simplest form, the problem can be defined as assigning a set of examinations to a fixed number of time periods so that no student is required to take more than one exam at any time.
- *Course timetabling*: Course timetabling often involves conditions in which students have requested a set of courses (seminars, tutorials, laboratories, etc), and the aim is to minimize the total number of conflicts and satisfy all constraints that is discussed in educational institutions.

There are, of course, substantial differences between two broad categories of the university timetabling problem, namely, course and exam timetabling. For example, a number of exams may be scheduled in one room or an exam may be split across several rooms, while in course timetabling one course (usually) has to be scheduled into exactly one room. Exam timetabling often aims to decrease the number of students who have exams in adjacent periods while in course timetabling it is usually desirable for students to have two or more courses in a row. Course timetabling problems will generally involve assigning events to a fixed set of timeslots (e.g. those occurring in exactly one week), whereas exam timetabling problems might sometimes allow some flexibility in the number of timeslots being used. MirHassani (2006b) predefined examination schedule that must be modified in order to maximize paper spread in course of examination. Azimi (2005), applied four techniques (Simulated Annealing, Tabu Search, Genetic Algorithm and Ant Colony) and three novel hybrid combinations of them to the classical examination timetabling problem, as an NP complete problem.

We concentrate on the university course timetabling problem. The aim is to find where and when a course has to be scheduled subject to available resources. The works published by researchers on this subject in the recent years are depicted in Fig. 1.

2.1 University course timetabling

University course timetabling problems tend to be large scale because of involvement in lots of courses, classes, lecturers and other resources. Also, the interplay between resources such as students and lecturers increase problem complexity. Timetabling in schools is usually more constrained. For example pupils have specified start, finish and time lunch. A compact timetable, in which there are no free periods, is normally a constraint that does not apply to a university context.

Altogether, there are many constraints that enforce the university requirements and students preference. These constraints are usually divided into two categories:

Table 1 Hard constraints frequently used by researchers

Hc1	Capacity of classroom is limited
Hc2	A class cannot have the same subject for more than two lecture periods a day.
Hc3	A class cannot have a lecture with more than one lecturer
Hc4	A lecturer can only deliver one lecture at a time
Hc5	Each lecture is exactly one period long
Hc6	Students can only have one lecture at a time
Hc7	Each lecturer must deliver a specified number of lectures per week
Hc8	A classroom can only be used for one lecture at a time
Hc9	Lessons can be blocked, if required
Hc10	Lectures can be pre-scheduled to a specific time
Hc11	Lecturers' unavailability is considered
Hc12	Allocated rooms must be large enough to accommodate the students
Hc13	Only one lecture for a particular course is allowed
Hc14	Double lectures must include two consecutive periods
Hc15	A set of precedence requirements stating that certain events should occur before others
Hc16	We have different time slots of a lecture day that some of them are overlap

I. *Hard constraint*: These constraints must be satisfied in order to produce a feasible timetable. Perhaps the most usually hard constraint in timetabling is the so-called "event-clash" constraint. Examples of such constraints are:

- A lecturer or a student cannot be assigned to more than one lecture in the same time slot; (Causmaecker et al. 2009).
- A room cannot hold more than one lecture at the same time; (Causmaecker et al. 2009).

II. *Soft constraint:* Soft constraints are desirable but not absolutely essential. Such constraints often determine the quality of a timetable. In real-world situations, it is usually impossible to satisfy all soft constraints. Certainly, the quality of a solution is depending on these constraints.

For Example:

- Class timetabling should be as compact as possible, eliminating idle times for students; (Aladag et al. 2009).
- If possible, students should not have a day with a single class; (Aladag et al. 2009) Table 1, identifies 16 hard constraints and Table 2 classifies 22 soft constraints that have been mostly considered by researchers. Some hard constraints (e.g. Hc1, Hc2, Hc3, Hc4, Hc5 and Hc6) have been widely used, whereas others (e.g. constraints Hc9, Hc7, Hc13 and Hc14) are only considered by a few researchers.

Some constraints are considered as hard constraints by a group of researchers and soft by others. For example, room capacity (Hc1) was considered as a hard constraint by (Luis et al. 2009), but a soft constraint (Sc7) by (Pongcharoena et al. 2008) and, (Deris et al. 2000). An interesting constraint is Hc9 that packs together the grade lessons in a specific style and reduces the problem size. So we can solve the problem more easily (for example students that have math 1 cannot have math 2; therefore, we can spot these as one lesson). Hc11 is related to cases that teachers teach in more than one university or are part-time, and their

Table 2 Soft constraints frequently used by researchers

Sc1	Some lecturers require special facilities. (special tools)
Sc2	Students must not have spare periods or a day with a single session
Sc3	Conflicts between optional subjects chosen by students should be avoided
Sc4	Lecturers can specify times when they prefer not to lecture
Sc5	Some lectures should not take place late in the evening
Sc6	Lecturers' timetables should avoid gaps
Sc7	Rooms should be fully occupied whenever possible, but capacity constraints should not be violated
Sc8	The timetables for rooms should be as compact as possible
Sc9	Lectures should be spread uniformly over the whole week
Sc10	An hour lunch break must be scheduled between 12:00 and 14:00
Sc11	The students provide a sorted list of preferred course
Sc12	Teaching load of a faculty member must be observed
Sc13	The number of lessons per day can be limited
Sc14	The number of students having lunch at a given time should be controlled
Sc15	Lecture rooms should be close to the host department
Sc16	Students should have consecutive lectures in the same building (to avoid moving students)
Sc17	Classes should have lectures either in the morning or in the afternoon
Sc18	Some classes may be split into smaller groups (for example seminar, laboratory, etc) the students need to be equally distributed.
Sc19	All sessions of the course should be scheduled in the same room and the same time-slot, but in different days.
Sc20	Some classes are offered jointly with tutorials, or lab sessions, or both of them
Sc21	Some courses have more than one lecturer
Sc22	Some groups of courses must allocate to special time slots

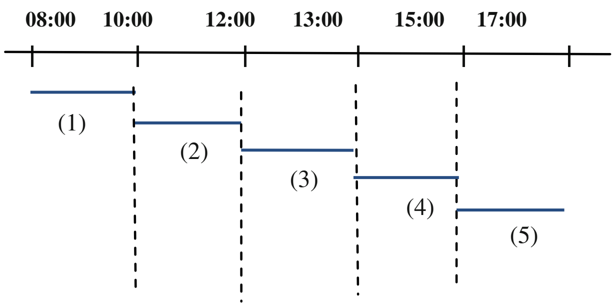

Fig. 2 Separate time slots

particular situation has to be taken into account. Hc12 enforces classroom capacity. Hc15 concerns courses that have prerequisite, so this course should first survive before the course has a prerequisite. For example, the prerequisite of math2 is math1, so if the students do not pass math 1, they cannot take math 2. Hc16 takes into account circumstances when time-slots overlap (see Figs. 2 and 3). In both situations the timetable should not involve conflict.

Now we are back to the soft constraints in Table 2. Sc2 represents that there are no gaps between time-slots that students have courses. Sc19 corresponds to the case that courses require more than one session, (e.g. have 2 sessions), in this case, both sessions must be

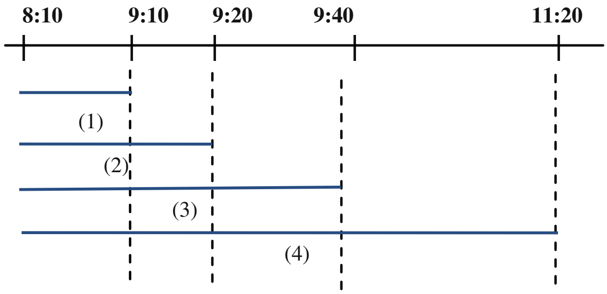

Fig. 3 Partially overlapped time slots

Table 3 Objective functions frequently used by researchers

Obj1	Maximize the students' preferences
Obj2	Minimize the total number of non-assigned subjects
Obj3	The obtained timetable must be as compact as possible
Obj4	Minimize the number of student left without a seat at a class
Obj5	Minimize the number of distinct course-room allocation on the top of a single course-room allocation per course
Obj6	Maximize the number of assigned course with urgency
Obj7	Minimize to offer sessions of one course in continuous days (the obtained timetable must have one day off between two sessions of each course)
Obj8	Minimize the conflict of days and time slots for different course
Obj9	Minimize the total dissatisfaction and inequity cost associate with assignment faculty members to classes (Maximize the faculty preference)
Obj10	Minimize the empty periods of each teacher (Compact timetable for lecturers)
Obj11	Minimize the empty periods of students between two classes (Compact timetable for students)
Obj12	Minimize the conflicts between co-requisites courses that can take in the same term

scheduled in the same time slot (e.g. both of them in 13:00–15:00) and in the same class but in different days. Sc20, is about some courses, theoretical or practical or both, that must be offered jointly with tutorials, or lab sessions. Sc21 is related to courses that have more than one lecturer, for example in medical branches, more than one lecturer will evaluate a course. Sc22 shows the case in which a group of courses must be offered in a special time slot. For example, courses with 2 and 4 (time) units must be offered in special time slots and courses with 3 units should be offered in other time slots that are perhaps different.

Beyond this constraint, however, there are many other kinds of constraints—hard and soft—which can affect a timetable. In the real world, it is usually the case that most universities will have their own specific probationary set of constraints that makes their particular timetabling problem different to others.

The aim of creating a timetable can be different with administrators. Table 3 identifies 12 objective functions have been considered by previous research. Attend in more occasions, soft constraints are spot at mould objective function. In this case, some soft constraints stand with a penalty coefficient in objective function. For example Obj.8, in fact this function is a soft constraint that its violation will be measured by penalty coefficient and the objective is to minimize the penalties.

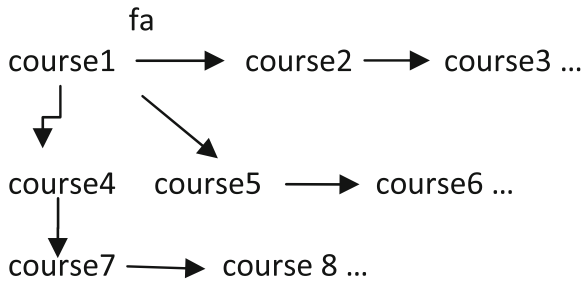

Fig. 4 Prerequisite and co-requisite courses

Object 1, represents the preferences of student, for example, the days in which they prefer to have classes, should maximize. Obj.2 shows how the courses, teachers, and students have assigned to classes. Obj.4 represents whether the class allocated to the course has enough space. To best understand Obj.8, see Fig. 4. Here we see course 3, has only 2 courses (course1 and course2) as prerequisites; therefore, if the student passed course1 and course 2 he/she can take course 3 and course 4 and course5 (courses that only have course1 as prerequisite). So, courses 3, 4 and 5 must not conflict. Under this condition all students can take all courses that are not prerequisite (Note that courses that have prerequisite may conflict). Obj.12 describes co-requisite courses that must be taken together. Therefore, these courses must not have conflict.

3 Model based methods

Over the last 30 years, many OR techniques ranging from the use of mathematical programming to heuristics are adopted to solve the timetabling problem.

Asratian and Werra (2002), considered a theoretical model which extends the basic class-teacher model of timetabling. This model corresponds to some situations which occur frequently in the basic training programs of universities and schools. (Nergiz et al. 2005), consider a sub-problem of the general timetabling problem in the form of faculty–course–time slot (FCT) assignments in a single stage. They develop a linear 0–1 multi-objective model for this problem in which objective functions are related to the administration's total preferences on instructor–course and course–time slot assignments, and the instructors' total preferences on instructor–course–time slot assignments would be maximized simultaneously. Moreover, the model includes the administration's objective functions to minimize the total deviation from the instructors' upper load limits. To demonstrate the features of the model, a special example has been constructed. Because of the multi-objective nature of the FCT model, the solution process of this problem has been considered in two stages: secularization of the given problem and, then, solving. Because of the 0–1 nature of the problem, a special scalar formed approach called conic Scalarization is applied. Also the problem is formulated as a multi-objective model by Nergiz et al. (2005), and (Ozdemir and Gasimov 2004). Ozdemir and Gasimov (2004), also constructed a multi objective 0–1 nonlinear model for the problem, and explained an effective way to solve it. The Analytic Hierarchy Process (AHP) and the Analytic Network Process (ANP) are used to determine the weights of conflicting objectives. Efficient solutions corresponding to both sets of weights have been calculated, and the results compared.

Daskalaki and Birbas (2005) and Nergiz et al. (2005), decomposed the problem into a series of easier sub-problems. One of the main advantages provided by the decomposition of timetabling problems is that the solution process becomes easier than that of the whole problem. Compared to a solving approach that solves the problem in a single stage, computation time for decomposed problems is significantly reduced; nevertheless there may be some loss in the quality of the solution. Ozdemir and Gasimov (2004), developed a two-stage relaxation procedure that solves the integer programming formulation of a university timetabling problem. Relaxation is performed in the first stage and concerns constraints that ensure consecutiveness in multi-period sessions of certain courses. These constraints, which are computationally more complex than others, are recovered during the second stage, and a number of sub-problems, one for each day of the week, are solved for local optima.

In recent years, because of the advancements in computer software and hardware, IP and MIP formulations have again started being an acceptable approach for many combinatorial problems. The new technologies in information systems, the availability of reliable software and the ability to solve relatively large problems in a relatively short time are the main reasons for making this traditional modeling approach attractive for solving realistic problems. Two decades back the problems that were solvable by classical IP techniques, mainly branch-and-bound, carried tens of integer variables. Now a problem with many thousands and in special occasions having millions of binary variables is not necessarily a trouble. Regarding the timetabling problems, IP models have been presented for the university timetabling problem: (Burke et al. 2010; Daskalaki and Birbas 2005; Daskalaki et al. 2004; Al-Yakoob et al. 2006, 2007; Boland et al. 1977; Papoutsis et al. 2003; MirHassani 2006a) and (Broek et al. 2009) among people who worked on MIP formulation.

Daskalaki and Birbas (2005) represented a two-stage relaxation procedure that efficiently solves the integer programming formulation of a university timetabling problem. The relaxation is performed in the first stage and concerns the constraints that warrantee consecutiveness in multi-period sessions of certain courses. These constraints, which are computationally heavier than the others, are recovered during the second stage and a number of sub problems, one for each day of the week, are solved. Comparing to a solution approach that solves the problem in a single stage, computation time is reduced significantly without any loss in quality for the resulting timetables. The new solution approach gives a chance for further improvements in the final timetables, as well as for a certain degree of interaction with the users during the construction of the timetables. Daskalaki et al. (2004), presented a novel 0–1 integer programming formulation of the university timetabling problem. The model provides constraints for a great number of operational rules and requirements found in most academic institutions. Treated as an optimization problem, the objective is to minimize a linear cost function. With this objective, it is possible to consider the satisfaction of expressed preferences regarding teaching periods or days of the week or even classrooms for specified courses. Moreover, with suitable definition of the cost coefficients in the objective function, it is possible to reduce the solution space and make the problem tractable. The model is solvable by existing software tools with IP solvers, even for large departments. The case of a five-year Engineering Department with a large number of courses and teachers is presented along with its solution as resulted from the presented IP formulation.

Al-Yakoob et al. (2006, 2007), presented mathematical programming models for assigning faculty members to classes including, among typical academic class scheduling issues, certain specialized central policies at Kuwait University. The time-slots for classes are initially assumed to be given and an integer programming model (CFAM) is constructed to solve the resulting problem, which aims to minimize the individual and collective dissatisfaction of faculty members in a fair fashion, where dissatisfaction is measured by a function of the assignment of faculty members to time-slots and specific classes. In order to enhance the quality of results obtained in practice, the model is modified (ECFAM) so that the time-slots for the classes can be changed; however, with restrictions related to efficient facility utilization and permitting an administratively regulated maximum number of changes. Gender-based modeling considerations are also introduced in order to maintain desirable class offering patterns. Computational results are presented based on solving the models directly by the CPLEX-MIP (version 7.5) package and also using a specialized LP-based heuristic. The faculty schedules generated via the proposed approach based on a number of case studies related to the Department of Mathematics and Computer Science at Kuwait University reveal that this approach yields improved schedules in terms of fairness and enhanced satisfaction levels among faculty members. MirHassani (2006a) stated that the integer programming approach is well-suited for solving the timetabling problem. A novel 0–1 integer programming formulation of the university timetabling problem is presented. Its implementation is immediate by using a mathematical programming language and an integer programming solver. Broek et al. (2009) discussed the solution of this real-world timetabling problem. They presented a complete mathematical formulation and explained all the constraints resulting from the situation in Eindhoven. They solved the problem using lexicographical optimization with four sub problems. For all the four sub problems, an elegant integer linear programming model was given which can be easily solved. Finally, they reported on their computational experiments and results around the Eindhoven real-world data.

However, the effort required for modeling complicated operational rules, as well as the computational difficulties that result from the size of real problems have discouraged researchers and forced them to turn their interest to other approaches.

4 Heuristic methods

Timetabling has been proved to be a NP-hard problem. This means that the amount of computation required to solve problems increases exponentially with problem size; (Paker et al. 1988) and (Asratian and Werra 2002).This makes it time-consuming and hard to manually build timetables that satisfy the objectives and constraints, especially for large problems. Therefore, it is immediate to use efficacious search methods to produce optimal or near-optimal solutions that satisfy the constraints.

A wide variety of approaches to timetabling problems have been described in the literature and tested on real data. They can be almost divided into four types. Carter (1986a), Carter et al. (1996) and Carter and Laporte (1996a), describe the major components of the course timetabling problem. They discuss some of the initial types of algorithms that have been applied to these problems. They also provide a series of tables listing papers in refereed journals that have either implemented a solution or tested their algorithm on real data. They made no attempt to provide a qualitative comparison. They restricted their presentation to a description of the types of technique used and the size of problem solved not including commercial software vendors. Carter (1986a), did a survey of the actual applications of timetabling at several universities and a tutorial guide for practitioners on selecting and/or designing an algorithm for their own institutions.

The approaches can be roughly divided into four types (Burke and Petrovice 2002):

(1) sequential methods
(2) clustering methods
(3) constraint-based methods
(4) meta-heuristic methods

4.1 Sequential methods

These methods order events using domain heuristics and then sequentially assign the events into valid time slots so that no events in the period are in conflict with each other (Carter 1986a). In sequential methods, timetabling problems are usually represented as graphs where events (courses/exams) are represented as vertices, and conflicts between the events are represented by edges, this method was used by Burke et al. (2007), Werra (1985) and Haan et al. (2007).

Burke et al. (2007), presented an investigation of a simple generic hyper-heuristic approach upon a set of widely-used constructive heuristics (graph coloring heuristics) in timetabling. Within the hyper-heuristic framework, a Tabu search approach is employed to search for permutations of graph heuristics which are used for constructing timetables in exam and course timetabling problems. For example, if some students have to attend two events there is an edge between the nodes which represents this conflict. The construction of a conflict-free timetable can, therefore, be modeled as a graph coloring problem. Each time slot in the timetable corresponds to a color in the graph coloring problem and the vertices of a graph are colored in such a way so that no two adjacent vertices are colored by the same color. A variety of graph coloring heuristics for constructing conflict-free timetables is available in the Carter and Laporte (1996a). These heuristics order the events based on an estimation of how difficult it is to schedule them. The heuristics that are often used are: Burke and Petrovice (2002).

Largest degree first: Events that have a large number of conflicts with other events (i.e., a large degree) are scheduled early. The fact is that the events with a large number of conflicts are more difficult to schedule and so should be tackled first.

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: These heuristic prioritizes those events that have the largest numbers of conflict with events that have already been scheduled.

4.2 Clustering methods

In these methods, the set of events is split into groups which satisfy hard constraints and then the groups are assigned to time periods to fulfill the soft constraints. An early paper to describe this approach was written by Papoutsis et al. (2003). Beligiannis et al. (2007) and Fisher and Shier (1983) used different optimization techniques to solve the problem of assigning the groups of events to time slots. The main drawback of these approaches is that the clusters of events are formed and fixed at the beginning of the algorithm and that may result in a poor quality timetable. Haan et al. (2007), used this method in a 4-phase approach to a timetabling problem in secondary school as it is common in the Netherlands. The problem has been stated as a graph coloring problem with extra conditions on the availability of resources (rooms, teachers). The size of the graph involved, and the extra efforts to improve the quality are the main reasons for the 4-phase approach. They try to control the quality by a preprocessing phase, and a post-processing phase. In the preprocessing phase, they cluster events in so-called cluster-schemes. These clustered events can be considered as the new events to be scheduled. In the second and third phase a feasible timetable is constructed.

In the fourth phase a Tabu Search is used to improve the best schedule found. The developed approach is tested by using data from the Kottenpark, in the Netherlands.

4.3 Constraint-based methods

In these methods a timetabling problem is modeled as a set of variables (i.e., events) to which values (i.e., resources such as rooms and time slots) have to be assigned to satisfy a number of constraints. Usually a number of rules are defined for assigning resources to events. When no rule is practicable to the current retail solution, a backtracking is enforced until a solution is found that satisfies all constraints. Abdennadher and Marte (2000), showed how to model their timetabling problem as a partial constraint satisfaction problem and gave a concise finite domain solver implemented with Constraint Handling Rules that, by performing soft constraint propagation, allows for making soft constraints an active part of the problem-solving process. Furthermore, they improved efficiency by reusing parts of the timetable of the previous year. Their prototype needs only a few minutes to create a timetable while manual timetabling usually takes a few days. Valouxis and Housos (2003), have talked about the timetabling problem for a typical high school environment, that was modeled and solved using a constraint programming (CP) approach. The previous timetabling problem was defined as a constraint satisfaction problem (CSP), consisting of a set of variables and a set of constraints. For each variable, a finite set of possible input values is defined and the constraints main role is to restrict the values that the problem variables can simultaneously take.

4.4 Meta-heuristic methods

Over the last two decades, a variety of meta-heuristic approaches such as Simulated Annealing, Tabu Search, Genetic Algorithms, Ant Colony and hybrid approaches have been investigated for the timetabling problem. Also, some very good results have been reported. Meta-heuristic methods begin with one or more initial solutions and employ search strategies that try to avoid local optima. Most of these search algorithms can produce high quality solutions (Burke and Petrovice 2002).

4.4.1 Genetic and memetic algorithm

The grouping genetic algorithm (GGA) is a class of evolutionary algorithm mainly modified to tackle grouping problems, i.e. problems in which a number of parts must be assigned to a set of predefined groups. Dimopoulou and Miliotis (2001), presents a novel application of the hybrid grouping genetic algorithm in a problem related to university timetabling.

Specifically, the assignment of students to laboratory groups is tackled. It was proposed by Falkenauer (1992), when realized that traditional genetic algorithms had some drawbacks while they were applied to grouping problems (mainly, the traditional encoding increases the space search size in this kind of problems). Thus, in the GGA, the encoding, crossover and mutation operator of traditional GAs are modified, to obtain a compact algorithm with very good performance in grouping problems. The GGA has been successfully applied to a number of problems, in different fields such as (Brown and Vroblefski 2004) in telecommunications, (James et al. 2007b) in manufacturing, (James et al. 2007a) in planning the problem and (Hung et al. 2003) in industrial engineering. Ross et al. (2003), used genetic algorithms to solve the timetabling problem. Beligiannis et al. (2007), used this algorithm for school timetabling. The proposed genetic algorithm is used in Beligiannis et al. (2007), to create feasible and efficient timetables for high schools in Greece.

Carrasco and Pato (2001), began by presenting the timetabling problems that emerge in the context of educational institutions. This is followed by a description of the basic characteristics of the class/teacher timetabling problem. A multi-objective genetic algorithm was proposed for this timetabling problem, incorporating two distinct objectives. They concern precisely the minimization of the violations of both types of constraints, hard and soft, while respecting the two competing aspects—teachers and classes. A brief description of the characteristics of a genetic multi objective meta-heuristic is presented, together with the non-dominated sorting genetic algorithm. This approach represents each timetabling solution with a matrix-type chromosome and is based on special-purpose genetic operators of crossover and mutation developed to act over a secondary population and a fixed-dimension main population of chromosomes. The paper concludes with a discussion of the favorable results obtained through an application of the algorithm to a real instance taken from a university establishment in Portugal.

Rossi-Doria and Paechter (2004), tried the memetic algorithm to improve the performance of a genetic algorithm by incorporating local neighborhood search. The main idea of the memetic algorithm is to explore the neighborhood of the solutions obtained by a genetic algorithm and to navigate the search toward the local optima (for each solution) before passing back to the genetic algorithm and continuing the process. The main drawback of these approaches is that they need initial solution that generate randomly. If initial solution is not chosen fitly it may result in a poor quality timetable or too many algorithm iterations.

4.4.2 Simulated annealing

Simulated Annealing (SA) has its origins in statistical physics, where annealing involves the slow cooling of a solid until it reaches a low-energy ground state. In general, the SA performs a stochastic search of the neighborhood space. SA starts with an initial state (s), which includes a point randomly selected from the search (solution) space and an initial temperature. A next point (s*) is randomly selected from a set of neighbors to the current point. If the objective function associated with the selected neighbor, E(s*), is better than the objective value of the initial point E(s), it is accepted as the new point for the next neighborhood search (NS). Otherwise, it is accepted with a probability where Δ e is the change to the objective value. The next step is another search for a new set of neighborhoods for the selection process. The temperature is then reduced to focus on a specific region. A cooling rate (r) is specified that determines the amount of computation required and the quality of solutions. The initial temperature value (t_0), the number of iterations to be performed at each temperature, the cooling rate and the termination criterion are specified by the "SA cooling schedule".

SA has been used to solve many types of combinatorial optimization problems including course scheduling and examination timetabling. Among persons who have adopted this approach are, (Grigorios et al. 2008), and, (Kustoch 2003).

4.4.3 Tabu search

One of the most efficacious algorithms for the solution of the problem is the Tabu Search (TS) algorithm. TS has proved its efficiency in solving the combinatorial optimization problems. A Tabu search algorithm consists of using advanced strategies and usual components such as Tabu list, various memories, neighborhood structures, and so on. One of the most important factors which affect the sufficiency of the algorithm is a defined neighborhood structure pertained to the nature of the problem. The idea behind TS is to start from a random solution and successively move to one of its neighborhoods, see Aladag et al. (2009), Cordeau et al. (2003), Causmaecker et al. (2009), and Jaumard et al. (2002), among the other.

Causmaecker et al. (2009), presented a decomposed meta-heuristic approach to solve a real-world university course timetabling problem. Essential aspects in this problem are the overlapping time slots (see Fig. 3) and the irregular weekly timetables. A first stage in the approach reduces the number of subjects through the introduction of new structures. The next stages involve a meta-heuristic search that attempts to solve the constraints one by one, instead of trying to find a solution for all the constraints at once. Test results for a real-world instance are presented since their main concern was the automated generation of real-world university course timetables and not the construction of yet another new search algorithm. They applied Tabu search which has been proven to be very successful in a variety of timetabling problems.

Aladag et al. (2009), examined the four different neighborhood structures based on types of move such as simple, swap. Two of the four neighborhood structures used in their study were used by Aladag and Hocaoglu (2007a), and Alvarez et al. (2002). The two neighborhood structures differ in terms of the used moves which are simple and swap. Other used neighborhood structures proposed in this paper compose of combining the simple and swap moves. By doing so, they aimed at constituting a diversification effect in the used TS algorithm. Based on the usage of four neighborhood structures, the fall semester of course timetabling problem of the department of statistics at Hacettepe University is solved utilizing the TS algorithm introduced. According to the results obtained, multiple comparisons among all neighborhood structures are statistically conducted.

As mentioned, the approach needs to start from a random solution, so a fit initial random solution must be picked and moved toward a high quality timetable. It is the main drawback of this approach.

Zhipeng et al. (2009), present an Adaptive TS algorithm (ATS) for solving a problem of curriculum-based course timetabling. The proposed ATS algorithm integrates several distinguished features such as an original double Kempe chains neighborhood structure, a penalty-guided perturbation operator and an adaptive search mechanism. Computational results show the high effectiveness of the proposed ATS algorithm compared with five reference algorithms as well as the current best known results. This paper also gives an idea about the essential ingredients of the ATS algorithm.

Burke et al. (2010), studied an approach to such problems, which can be thought of as multi-phase exploitation of multiple objective/value—restricted sub-models. In this approach, only one computationally difficult component of a problem and the associate subset of objectives are considered at first. This produced a partial solution, which defines interesting neighborhoods in the search space of the complete problems. Often, it is possible to pick the initial component so that variable aggregation can be performed at the first stage, and the neighborhoods to be explored next are guaranteed to contain feasible solutions. The goal is to find an assignment of events to slots and rooms, so that the assignment of events to slots is a feasible bounded coloring of an associated conflict graph and the linear combination of the number of violation of soft constraint is minimized. In the proposed heuristic, an objective restricted neighborhood generator product assignment of periods to events, with decreasing number of violations of two period-related soft constraints. Those are relaxed into assignment of events to days, which define neighborhoods that are easier to search with respect to all four soft constraints, integer programming formulation for all sub problems are given and evaluated.

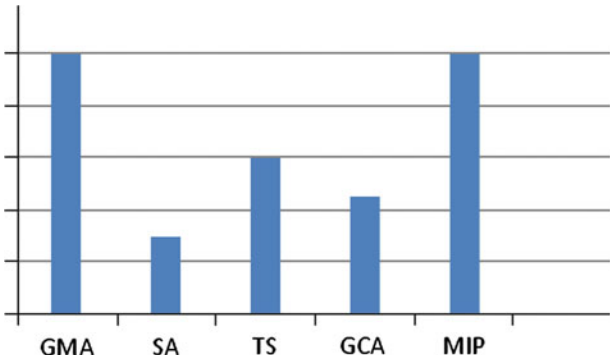

Fig. 5 Employed methods (2002–2010)

The most frequently used heuristic methods are Genetic and Memetic Algorithm (GMA), Simulated Annealing (SA), Tabu Search (TS), Graph Coloring Algorithm (GCA) and Mixed Integer Programming (MIP). In Fig. 5, we see that across methods said to solve timetabling problems, in recent years, employed the MIP and the genetic algorithm more than other approaches.

The timetabling problem can also be represented as stochastic optimization. Pongcharoena et al. (2008), described the Stochastic Optimization Timetabling Tool (SOTT) that has been developed for university course timetabling. GA, SA and random search are embedded in the SOTT. The algorithms include a repair process, which ensures that all infeasible timetables are rectified. This prevents clashes and ensures that the rooms are sufficiently large to accommodate the classes. The algorithms also evaluate timetables in terms of soft constraints; minimizing student movement; avoiding fragmentation in the timetables for students and lecturers; and satisfying lecturers' preferences for the timing of classes. The algorithms were tested using two sets of timetabling data from a collaborating university. Both GA and SA produced very good timetables, but the results obtained from SA were slightly better than those using GA.

5 Directions for future work

We believe that there is still considerable scope for further work in this general direction. Because the university course timetables produced in this way involves some problems.

- One of the problems with many existing university course timetabling systems is that the system is often tailored to the needs of a particular institution or user. So, it is necessary to provide a method that easily reformulated or customized to support change.
- Also, there is no timetable that spots all constraints (hard and soft) and objectives. In the most university course timetabling tasks, only some of these constrains, not all, are enforced, and to compliance some objectives, jar on other objectives. Therefore a more general timetable is essential.
- As mentioned, the general course timetabling problem is known to be NP-hard and also, in practice, a challenging computational task; therefore, the need for an algorithm that can solve the whole timetabling problem (relevant to each institute), and generate high quality timetables in the shortest computational time is evident. Many successful university timetabling systems are often applied only in the institutions where they were designed.

- It pointed out that manual techniques for course timetabling problem are very time-consuming. It is clear that a well-established method could potentially lead to a valuable tool in solving these problems. The MIP models are solvable by current software tools but special software that utilize problem characteristics can lead to a powerful tool, even for large departments.

6 Conclusion

This paper surveys approaches to the solution of university course timetabling problem. An overall conclusion is that there are considerable benefits to be gained from studying the course timetabling problem. Different methods are discussed and a separate literature review is provided for each one. It seems that decomposing large problems into smaller ones and hybridizing the heuristics methods to produce advanced search methods could lead to powerful tools that function well not only on the particular problem but also create high quality timetables in general. These tools would be the appropriate facilities for providing valuable help to the decision maker to implement a good course and examination timetable. This approach seems to present a promising direction in the development of efficient algorithms that deal with large scale problems.

References

Abdennadher S, Marte M (2000) University course timetabling using constraint handling rules. J Appl Artif Intell 143:311–326

Aladag CH, Hocaoglu G (2007) A tabu search algorithm to solve course timetabling problem. Hacettepe J Math Stat 36(1):53–64

Aladag CH, Hocaoglu G, Basaran MA (2009) The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem. Expert Syst Appl 36:12349–12356

Alvarez R, Crespo E, Tamarit JM (2002) Design and implementation of a course scheduling system using tabu search. Eur J Oper Res 137:512–523

Al-Yakoob SM, Sherali HD (2007) A mixed-integer programming approach to a class timetabling problem: a case study with gender policies and traffic considerations. Eur J Oper Res 180:1028–1044

Al-Yakoob SM, Sherali HD (2006) Mathematical programming models and algorithms for a class–faculty assignment problem. Eur J Oper Res 173:488–507

Asratian AS, Werra D (2002) A generalized class teacher model for some timetabling problems. Eur J Oper Res 143:531–542

Azimi Z (2005) Hybrid heuristics for examination timetabling problem. Appl Math Comput 163:705–733

Beligiannis GN, Moschopoulos C, Likothanas SD (2007) A genetic algorithm approach to school timetabling. Journal of the Operational Research Society 1:1–20

Birbas T, Daskalaki S, Housos E (1977) Timetabling for Greek high schools. Journal of the Operational Research Society 48:1191–1200

Boland N, Hughes BD, Merlot LT, Stuckey PJ (1977) Timetabling for Greek high schools. Journal of the Operational Research Society 48:1191–1200

Broek JVD, Hurkens C, Woeginger G (2009) Timetabling problems at TU Eindhoven. European Journal of Operation Research 192:877–885

Brown EC, Vroblefski M (2004) A grouping genetic algorithm for the microcell sectorization problem. Engineering Applications of Artificial Intelligence 17(6):589–598

Burke Ek, Kingston J, de Werra D (2004) Applications to timetabling. In: Gross J, Yellen J (eds) Handbook of graph theory. Chapman Hall/CRC Press, London, pp 445–474

Burke EK, Kingston J, Jackson K, Weare R (1997) Automated university timetabling. The state of the art. Comput J 40(9):565–571

Burke EK, Elliman DG, Weare RF (1994a) A genetic algorithm for university timetabling. AISB Workshop on Evolutionary Computing, University of Leeds, UK, Society for the Study of Artificial Intelligence and Simulation of Behaviour, pp 35–40

Burke EK, Elliman DG, Weare RF (1994b) A university timetabling system based on graph colouring and constraint manipulation. J Res Comput Educ 27(1):1–18
Burke EK, Ross P (1996) The practice and theory of automated timetabling. Selected papers from the 1st international conference on the practice and theory of automated timetabling, Napier University, August/September, Springer Lecture Notes in Computer Science Series, pp 309–324
Burke EK, Kendall G, Soubeiga E (2003) A tabu search hyper heuristic for timetabling and rostering. J Heuristics 9(6):51–70
Burke EK, McCollum B, Meisels C, Petrovic A, Rong Q (2007) A graph-based hyper-heuristic for educational timetabling problems. Eur J Oper Res 176:177–192
Burke EK, Mrecek J, Parkes AJ, Rudova H (2010) Decomposition, reformulation, and divingin university course time tabling. Comput Oper Res 37:582–597
Burke EK, Petrovice S (2002) Recent research directions in automated timetabling. Eur J Oper Res 140: 266–280
Carrasco MP, Pato MV (2001) A multiobjective genetic algorithm for the class/teacher timetabling problem. Lect Notes Comput Sci 2079:3–17
Carter M (1986) A lagrangian relaxation approach to the classroom assignment. INFOR 27(2):230–246
Carter M (1968) A survey of practical applications of examination timetabling algorithms. Oper Res 34: 193–202
Carter MW, Laporte G (1998) Recent developments in practical course timetabling. In: Selected and revised papers of the second international conference on practice and theory of automated timetabling (PATAT 1997), LNCS. Springer, Toronto, vol 1408, pp 3–19
Carter M, Burke EK (1998) The practice and theory of automated timetabling II, selected papers from the 2nd international conference on the practice and theory of automated timetabling. University of Toronto o, August 278 Burke EK and Petrovic S J Oper Res 140, pp 266–280
Carter MW, Laporte G (1996a) Recent developments in practical examination timetabling. In: Burke EK, Ross P (eds), the practice and theory of automated timetabling. Springer, Berlin, Selected papers from the 1st international conference. LNCS 1153. Springer, Berlin, Heidelberg, pp 3–21
Carter MW, Laporte G, Lee SY (1996) Examination timetabling: algorithmic strategies and applications. J Oper Res 74:373–383
Causmaecker DBB, Demeester PA, Berghe GV (2009) A decomposed metaheuristic approach for a real-world university timetabling problem. Eur J Oper Res 195:307–318
Cordeau JF, Jaumard, Morales R (2003) Efficient timetabling solution with tabu search. International timetabling competition results, http://www.idsia.ch/Files/ttcomp2002/jaumard.pdf
Daskalaki S, Birbas T (2005) Efficient solutions for a university timetabling problem through integer programming. Eur J Oper Res 160:106–120
Daskalaki S, Birbas T, Housos E (2004) An integer programming formulation for a case study in university timetabling. Eur J Oper Res 153(1):117–135
Deris S, Omatu S, Ohta Hiroshi (2000) Timetable planning using the constraint-based reasoning. Comput Oper Res 27:819–840
Dimopoulou M, Miliotis P (2001) Implementation of a university course and examination timetabling system. Eur J Oper Res 130:202–213
Falkenauer E (1992) The grouping genetic algorithm–widening the scope of the GAs. In: Proceedings of the Belgian journal of operations research, statistics and computer science, vol 33, pp 79–102
Fisher JG, Shier DR (1983) A heuristic procedure for largescale examination scheduling problems. Technical Report, Department of Mathematical Sciences, Clemson University 417
Grigorios N et al (2008) Likothanassisa applying evolutionary computation to the school timetabling problem. Greek Case Comput Oper Res 35:1265–1280
Haan pd, Landman r, Post G, Ruizena H (2007) A four-phase approach to a timetabling problem in secondary schools. 7500 AE Enschede, The Netherlands Department of applied Matematics, University Twente, P.O. Box 217
Hung C, Sumichrast RT, Brown EC (2003) A grouping genetic algorithm for material cutting plan generation. Comput Ind Eng 44(4):651–667
Ismayilova NA, Sagir M, Rafail N (2005) A multiobjective faculty–course–time slot assignment pr with preferences. Math Comput Modell 46:1017–1029
James T, Vroblefski M, Nottingham Q (2007) A hybrid grouping genetic algorithm for the registration area planning problem. Comput Commun 30(10):2180–2190
James TL, Brown EC, Keeling KB (2007) A hybrid grouping genetic algorithm for the cell formation problem. Comput Oper Res 34:2059–2079
Jaumard J, Cordeau R, Morales (2002) Efficient timetabling solution with tabu search, Working Paper, Available from Metaheuristics Network. International Timetabling Competition

Kustoch PA (2003) Timetabling competition SA-based heuristics. International timetabling competition results

Luis E, Blas A, Salcedo-Sanz S, Emilio G, Portilla A (2009) A hybrid grouping genetic algorithm for assigning students to preferred. Expert Syst Appl 36:7234–7241

MirHassani SA (2006) A computational approach to enhancing course timetabling with integer programming. Appl Math Comput 175:814–822

MirHassani SA (2006) Improving paper spread in examination timetables. Appl Math Comput 179:702–706

Newall JP, Weara RF, Burke E (1996b) A memetic algorithm for university exam timetabling Practice and theory of automated timetabling. Lecture notes in computer science. Springer, Berlin, pp 241–250

Ozdemir MS, Gasimov RN (2004) The analytic hierarchy process and multiobjective 0–1 faculty course assignment problem. Eur J Oper Res 157(2):398–408

Paker RG, Rardin RL, Diego S (1988) Discrete optimization. Academic Press Inc, USA

Papoutsis K, Valouxis C, Housos E (2003) A column generation approach for the timetabling problem of Greek high schools. J Oper Res Soc 54:230–238

Pongcharoena P, Promtetb W, Yenradeec P, Hicksd C (2008) Stochastic optimization timetabling tool for university course scheduling. Int J Prod Econ 112:903–918

Ross P, Hart E, Corne D, Chosd, Tsutsui S (2003) Genetic algorithms and timetabling. Advances in evolutionary computation, theory and applications, pp 755–771

Rossi-Doria O, Paechter B (2004) A memetic algorithm for university course timetabling, in Combinatorial Optimisation. Book of Abstracts

Schaerf A (1999) A survey of automated timetabling. Artif Intell Rev 13(2):87–127

Valouxis C, Housos E (2003) Constraint programming approach for school timetabling. Comput Oper Res 30:1555–1572

Werra D (1985) An introduction to timetabling. Eur J Oper Res 19(2):151–162

Zhipeng Lu AB, Jin-Kao Hao A (2009) Adaptive Tabu Search for course timetabling. Eur J Oper Res 200(1):235–244