Ann Oper Res
DOI 10.1007/s10479-013-1321-8

A survey of school timetabling research

Nelishia Pillay

Abstract Although there has been a fair amount of research in the area of school timetabling, this domain has not developed as well as other fields of educational timetabling such as university course and examination timetabling. This can possibly be attributed to the fact that the studies in this domain have generally been conducted in isolation of each other and have addressed different school timetabling problems. Furthermore, there have been no comparative studies on the success of different methodologies on a variety of school timetabling problems. As a way forward this paper provides an overview of the research conducted in this domain, details of problem sets which are publicly available and proposes areas for further research in school timetabling.

Keywords School timetabling · Educational timetabling

1 Introduction

Educational timetabling encompasses university course timetabling, examination timetabling and school timetabling. A lot of progress has been made in university course and examination timetabling research. This can be attributed to the variety of problems publicly available which has enabled a comparative study of different methodologies for these domains. School timetabling has been defined as a type of classroom assignment (Carter and Tovey 1992) or course timetabling (Carter and Laporte 1997) problem. While there have been numerous studies focusing on solving the school timetabling problem, research in this area has not advanced as rapidly as university course and examination timetabling (Nurmi and Kyngas 2008; Post et al. 2008). This is possibly due to studies being done in isolation of each other for specific schools (Santos et al. 2008) and the lack of a variety of problems which can be accessed publicly (Schaerf 1999a; Smith et al. 2003; Jacobsen et al. 2006; Post et al. 2008). Furthermore, methods implemented to solve this problem have not been widely tested on a variety of school timetabling problems (Jacobsen et al. 2006;

N. Pillay (✉)
School of Computer Science, University of KwaZulu-Natal, Pietermaritzburg, South Africa
e-mail: pillayn32@ukzn.ac.za

Published online: 15 February 2013

Santos et al. 2005) which is essential in order to ascertain how well the methodology can generalize. Due to the unavailability of a general set of benchmark problems, there have not been many comparative analyses of different methodologies in solving the school timetabling problem. In cases where such studies have been performed, different methods are compared for a single school and not a variety of problems.

This paper serves as a starting point for further development in the field by conducting a survey of research in the domain of school timetabling and suggesting directions for future research. The school timetabling problem is an NP-complete or NP-hard problem depending on the constraints associated with a specific problem. Various studies have been conducted into the complexity of this problem, illustrating why it is NP-complete. Research into the complexity of the school timetabling problem began as early as 1976 with the study conducted by Even et al. (1976) and includes more recent analyses by Carter and Tovey (1992), Cooper and Kingston (1996), Eikelder and Willemen (2001) and Willemen (2002).

Previous surveys on educational timetabling have been conducted by Carter and Laporte (1997), Schaerf (1999a) and Burke et al. (2004). Carter and Laporte (1997) report on course timetabling which class-teacher programming (i.e. school timetabling) is a subproblem of. Schaerf (1999a) provides an overview of educational timetabling, namely, school timetabling, course timetabling and examination timetabling. The paper describes the constraints of these problems and methods used to solve these three educational timetabling problems. Burke et al. (2004) define the class-teacher/school timetabling problem and outline the role that graph theory has played in solving this problem, amongst other timetabling problems. Junginger (1986) has conducted a survey of school timetabling specifically for German schools. The school timetabling problem and the methods employed to solve this problem have evolved over time. This paper attempts to provide an up-to-date account of research in this domain.

The definition of the school timetabling problem and the hard and soft constraints associated with each problem instance differs from one study to the next. The following section provides a standardized definition of the problem. Section 3 presents an overview of research in the field. Data formats for school timetabling problems and problem sets that are publicly available for research are described in Sect. 4. Section 5 takes a look at what is happening in the software industry with respect to school timetabling software and provides an overview of commercial software and freeware available for school timetabling. Future research directions for school timetabling are proposed in Sect. 6.

2 The school timetabling problem (STP)

This section describes the general school timetabling problem. The school timetabling problem and the terminology used in defining the problem differs drastically from one study to the next. For consistency the section firstly defines the terms used in the subsections to follow. The STP can be defined in terms of the requirements of the problem, the hard constraints and soft constraints. Section 2.2 defines the school timetabling problem and Sect. 2.3 describes the hard and soft constraints that must be catered for in a standardized definition of the school timetabling problem. Objective functions that are generally used to evaluate school timetables are discussed in Sect. 2.4.

2.1 Terminology

This section defines terms to be used in the subsections that follow. A *class* refers to a group of students that will be taught a particular subject, e.g. Mathematics.

The term *grade* refers to the level of schooling, e.g. twelfth grade is the last year of secondary schooling. Each grade contains one or more classes.

A *subject* refers to the actual content being taught, e.g. English or Mathematics. The subjects to be taught to each grade are determined by a curriculum which is usually set by a governing educational body.

A *lesson* refers to a particular subject being taught to a class by a teacher. A number of lessons are taught for each subject. These lessons are also referred to as *meetings* between a teacher and a class.

A *period* is a timetable slot which a lesson can be scheduled in. The duration of periods differs from school to school.

An *idle* or *free* period for a teacher is a period in which the teacher does not teach. Similarly, an *idle* or *free* period for a class is a period in which the class is not taught a lesson. A timetable without free or idle periods is described as *compact*. Some studies also distinguish between an idle and a free period. A free period is defined as one in which a teacher or a class is not involved in a lesson while an idle period is defined to be a free period with a lesson before and after it for the teacher or class. In this case a compact timetable is one that does not contain any idle periods.

A teacher may be required to serve one or more *relief* periods as a result of another teacher being absent. This essentially involves the teacher overseeing the absent teacher's class.

It may be required that some lessons be taught over two, i.e. a *double*, or three, i.e. a *triple*, consecutive periods.

In some cases classes are *split* into subgroups and each subgroup is taught a different subject simultaneously in different venues. Similarly, it may be necessary to *merge* classes for lessons and teach these in one large venue. The merger can occur over more than one grade.

A *tuple* is comprised of a class, teacher and a room which must be scheduled in a timetable period. In some cases a tuple may consist of only a class and a teacher. In addition to this a class may be the result of a split and/or merge. There may also be more than one teacher in a particular tuple.

A *resource* refers to any entity involved in a lesson. The standard resources are a class, teacher and the room in which the lesson is held.

2.2 Problem definition

Each school timetabling problem is defined in terms of a number of grades, available teachers and rooms, number of lessons to be taught by teachers to specific classes and a set of constraints. Solving the school timetabling problem essentially involves allocating class, teacher and room tuples to timetable slots so as to satisfy the hard and soft constraints of the problem (Abramson and Abela 1991; Beligiannis et al. 2008; Post et al. 2008). Post et al. (2008) describe classes, teachers and rooms as resources to be allocated. Students are usually grouped into classes prior to the timetable construction process (Abramson 1991). The school timetabling problem differs for each country due to the characteristics and regulations specific to the particular education system (Alvarez-Valdes et al. 1996; Post et al. 2008).

In some cases the room does not form part of the tuple and each class-teacher pair has to be allocated to a period and a venue (Wilke and Ostler 2008; Post et al. 2008). In this case class sizes must be provided. In addition to this a list of rooms and their corresponding capacities must be specified. In some versions of the problem classes have fixed rooms and teachers move from one class to another. In this case classes will only move to specialized rooms or in the case of splits or mergers for lessons. In some schools certain facilities, such as the sports ground, are shared with other schools.

In most cases teachers are pre-assigned to teach particular lessons to certain classes. However, other versions of the problem require the allocation of teachers to classes (Alvarez-Valdes et al. 2002; Kingston 2008). In this case more than one teacher is qualified to teach subjects to the classes at different levels (de Haan 2004). The number and duration of lessons can also differ for grades in a school (Post et al. 2008). Furthermore, some schools extend over more than one site (Schaerf 1996).

The hard and soft constraints also differ drastically from one problem to the next. Hard constraints are constraints that must be met by a timetable in order for it to be operable. A timetable meeting all the hard constraints of a problem is called a feasible timetable. Soft constraints are usually used to define the quality of a timetable. Soft constraints can be contradictory, and as such it is impossible to satisfy all the soft constraints. Thus, we attempt to minimize the soft constraint cost.

2.3 Problem constraints

Post et al. (2008) categorize constraints for the school timetabling problem into three groups, namely, basic scheduling constraints, event constraints and resource constraints. This paper subdivides this taxonomy further into seven categories, namely, problem requirements constraints, no clashes constraints, resources utilization constraints, workload constraints, period distribution constraints, preference constraints and lesson constraints.

Problem requirement constraints refer to constraints resulting from the specific school or education system rules, such as a teacher must teach a class a specific number of lessons, a lunch break must be scheduled.

The no clashes constraint category includes those constraints that require resources not to be utilized at the same time, e.g. a teacher cannot be scheduled to teach two lessons in the same period.

Resource utilization constraints prevent resources from being over-utilized or under-utilized and these constraints must be met in order for a timetable to be operable. For example, teachers should be scheduled when available. Room capacities must not be exceeded and all rooms should be utilized. Time for commuting between buildings and sites must be allowed for both classes and teachers if necessary.

Workload constraints refer to minimum and maximum limits placed on certain resources. For example, teachers may be required to teach a minimum and maximum number of lessons per week. Similarly, different classes may have a different number of maximum lessons per day.

Period distribution constraints are those that specify the period layout for resources such as classes and teachers. For example, lessons must be uniformly distributed for classes and teachers. Similarly, classes must have compact timetables, i.e. no free periods, while teacher timetables may include free periods.

Preference constraints describe preferences for the different resources, such as teachers wishing to teach in certain periods only, or scheduling lessons for particular subjects during certain parts of the day.

Lesson constraints specify requirements that must be met for certain lessons, e.g. some lessons must take place either before or after other lessons. In some cases classes must be split and/or merged for certain lessons. It may also be compulsory for certain lessons to take place during particular parts of the day, e.g. drama in the afternoons.

The following sections describe the constraints in each of these categories.

2.3.1 *Problem requirement constraints*

These constraints are defined by the requirements of the school timetabling problem and as such are usually hard constraints. These include:

- PR1—Classes must be scheduled for the required number of meetings for each subject.
- PR2—Teachers must be scheduled for the required number of meetings with each class.
- PR3—One period, in a specific range, should be allocated as a lunch break for pupils (Colorni et al. 1998; Lion 1967; Wilke and Ostler 2008). If a lunch break is not built into the timetable, and the corresponding period/s is not included in the unavailable periods, a specification indicating the need for a lunch break and the duration and range of periods during which it should be scheduled must be provided. Alternatively, it may be necessary for classes/grades to take a lunch break at different times to prevent overcrowding of the lunch room (Di Stephano and Tettamanzi 2001).

2.3.2 *No clashes constraints*

In order for a timetable to be operable there must be no clashes, i.e. each resource must not be scheduled more than once in a timetable period. Thus, the constraints in this category are hard constraints:

- NC1—A class must not be scheduled more than once during a period.
- NC2—A teacher must not be scheduled more than once during a period.
- NC3—A room must not be scheduled more than once during a period.

2.3.3 *Resource utilization constraints*

Constraints in this category are generally hard constraints that prevent the over-utilization or under-utilization of resources, i.e. classes, teachers and rooms:

- RU1—Time for travel—In some cases a school may be spread over a large area or more than one site (Di Stephano and Tettamanzi 2001). Some schools utilize facilities available at other schools, such as sports grounds (Nurmi and Kyngas 2007). In both these cases the timetable must make provision for time needed for travel between the different venues. In some cases the school may be spread over one site with large distances between buildings and again time must be allowed for travel between buildings (Di Stephano and Tettamanzi 2001).
- RU2—Teachers must only be scheduled when available (Beligiannis et al. 2008; Birbas et al. 1997; Post et al. 2008; Santos et al. 2008; Valouxis and Housos 2003; Wilke and Ostler 2008)—Teachers may be unavailable during certain periods due to administration tasks, teaching in another school, allocated free periods or days off. Teachers must not be scheduled to teach during these periods.
- RU3—Room capacities must not be exceeded (Wilke and Ostler 2008).
- RU4—All rooms must be used (Grobner et al. 2003). This is a soft constraint in some problems.
- RU5—Certain lessons require specialized rooms, e.g. science labs, computer lab, the gymnasium. These requirements must be met. In some problems the specialized rooms are highly utilized, making the STP more difficult (Wilke and Ostler 2008; Wright 1996; Wood and Whitaker 1998).
- RU6—Some schools are located over more than one site. In this case all the lessons to a class on a particular subject must be scheduled in a room on the same site (Post et al. 2008).

- RU7—Rooms may not be available at certain times due to maintenance, etc. Lessons must not be scheduled in the venues during these times.
- RU8—Classes may not be available during certain periods (Carrassco and Pato 2004).

2.3.4 *Workload constraints*

These constraints govern the workload of different resources:

- W1—Teacher workload must be adhered to—Teacher workload is defined in terms of a minimum and maximum number of teaching lessons or hours per week or per day (Birbas et al. 2009; de Haan et al. 2007b; Nurmi and Kyngas 2008; Santos et al. 2005; Wilke and Ostler 2008). Teachers may also be required to serve a certain number of relief periods (Bufe et al. 2001).
- W2—Classes for a particular grade cannot have more than a preset number of periods per day, e.g. the timetable for lower grades may be comprised of 9 periods while that for higher grades may schedule 11 periods (Raghavjee and Pillay 2010b).
- W3—A limit may be set on the maximum and minimum number of lessons a class may have per day, e.g. certain grades can no more than three Mathematics lectures in a day (Marte 2002).

W1 and W3 are defined as hard constraints for some problems and soft constraints for others. W2 is generally a hard constraint.

2.3.5 *Period distribution constraints*

These constraints place certain restrictions on the period distribution for resources in the timetable and are defined as hard constraints for some problems and soft constraints for others:

- PD1—Idle or free periods for classes—This constraint differs from one STP to the other. In some cases free periods are not allowed at all (Alvarez-Valdes et al. 1996; Jacobsen et al. 2006; Melicio et al. 2006; Wilke and Ostler 2008), i.e. there are no free periods between lessons in a school day. In other problems only some grades or levels, usually higher grades, can have free periods (Filho and Lorena 2001). Some problems also stipulate when free periods are permitted, e.g. last two periods of the day (Schaerf 1996; Valouxis and Housos 2003; Beligiannis et al. 2009).
- PD2—Idle or free periods for teachers—Teachers may be allowed some free periods (Wilke and Ostler 2008; Wright 1996). However the number of free periods for each teacher should usually be minimized when constructing the timetable (Birbas et al. 2009; de Haan et al. 2007b; Post et al. 2008; Wilke and Ostler 2008). Alternatively, it may be necessary for teachers to have compact timetables.
- PD3—Lesson spread for classes—Different STPs have different spread requirements for lessons. For example, there may be a restriction of at most one lesson for a subject per day (Alvarez-Valdes et al. 1996; Filho and Lorena 2001; Melicio et al. 2006; Wright 1996). Alternatively, the teaching of lessons on consecutive days over a certain period may not be favored (Alvarez-Valdes et al. 1996; Beligiannis et al. 2009). Some school timetabling problems require the lessons for each subject to be distributed uniformly throughout the week (Alvarez-Valdes et al. 1996; Birbas et al. 2009; Colorni et al. 1998; Wright 1996). Other restrictions include at most two daily lessons with the same teacher (Birbas et al. 1997; Santos et al. 2005) and the same subject should not be taught in the last period of one day and the first period of the following day (Wright 1996).

- PD4—Lesson spread for teachers—Some problems require the lessons taught by a teacher to be well-spaced throughout the week (Valouxis and Housos 2003). Alternatively, the lessons taught by a teacher should be concentrated over a limited number of days (Birbas et al. 1997; Filho and Lorena 2001; Santos et al. 2008) instead of a few lessons being dispersed over an entire week (Landman 2005).

2.3.6 *Preference constraints*

These constraints are usually soft constraints and describe resource preferences:

- P1—Lesson preferences—This refers to period preferences for lessons (Melicio et al. 2006; Wright 1996). There may be preferences for some subjects to be taught in morning sessions, e.g. mathematics or afternoon sessions (Colorni et al. 1998) or a particular subject should not be taught in the first period of a day (Nurmi and Kyngas 2008).
- P2—Teacher preferences—A teacher may prefer to teach in certain periods and not in others (Schaerf 1999b; Nurmi and Kyngas 2008; Valouxis and Housos 2003).

2.3.7 *Lesson constraints*

Constraints in this group define requirements that need to be met for certain lessons and are usually hard constraints:

- L1—Lessons should either be scheduled or not scheduled in certain periods. Some lessons may be pre-scheduled in particular periods, i.e. it may be compulsory for the lessons to take place during certain periods. For example, some lessons may require the use of facilities at other institutions which can only be utilized at certain times (Marte 2002). Alternatively, certain lessons have to be scheduled during particular parts of the day, e.g. theater and ballet in the afternoons (de Haan 2004). Similarly, physical education should not be scheduled just after lunch (Appleby et al. 1961). Some classes may start the school day later than other classes or finish earlier and thus cannot be scheduled in certain periods (Nurmi and Kyngas 2007).
- L2—Lesson sequences—It may be necessary for some lessons to occur before or after other lessons (Di Stephano and Tettamanzi 2001; Melicio et al. 2006; Nurmi and Kyngas 2007) on the same day or in the same week, e.g. practical work should be held after the theoretical aspects are taught (Srndic et al. 2009). Certain sequences of subjects may not be permitted (Di Stephano and Tettamanzi 2001), e.g. three language lessons in a row (de Haan 2004).
- L3—Double or triple lessons (De Haan et al. 2007a, 2007b; Filho and Lorena 2001; Jacobsen et al. 2006; Kingston 2004; Lion 1967; Melicio et al. 2006; Santos et al. 2008; Schaerf 1999b)—It may be necessary to schedule a double (two consecutive lessons) or triple lesson (three consecutive lessons) with a class for a particular subject.
- L4—Splitting and merging of classes (Beligiannis et al. 2008; Jacobsen et al. 2006; Kingston 2004; Kwok et al. 1997; Lion 1967; Marte 2006; Post et al. 2008; Wilke and Ostler 2008)—Classes may be merged together for a lesson. In some cases classes may have to split into subgroups with each subgroup being taught a different subject simultaneously. The split subgroups may also need to be merged differently from the original configuration. The splitting and merging may take place for the same grade or across grades.
- L5—Certain lessons must take place simultaneously (Birbas et al. 2009). Due to large student numbers, it may be necessary to repeat lessons. The repeat lesson would be taught simultaneously by different teachers (Hertz and Robert 1998).

2.4 Objective function

The literature cites two objective functions that are commonly used to evaluate school timetables. The first function is basically the sum of the hard and soft constraint violations (Abramson and Abela 1991; Valouxis and Housos 2003; Wood and Whitaker 1998). The second function used is the weighted sum of the hard and soft constraint violations which allows for some constraints to have a higher priority than others (Schaerf 1996; Wright 1996).

3 Solving the school timetabling problem

This section examines some of the methodologies that have been used to solve the STP. This survey of techniques is a work-in-progress and is by no means exhaustive. Early research into solving the school timetabling problem focused on heuristic methods that mimicked the manual approach to solving the school timetabling problem (Schaerf 1999a). Appleby et al. (1961) and de Gans (1981) employ such methods. Earlier methods also include improvement algorithms, such as that used by Aust (1976). In this study a three phase approach is taken, with each phase focusing on one of the problem constraints. Improvements are achieved by swapping timetabling entries. Graph theory methods (Burke et al. 2004; Cooper and Kingston 1993; Lion 1966) and network flow methods (Hertz and Robert 1998; Cooper and Kingston 1995; de Werra 1971) have also played an important role in solving the school timetabling problem. As the field of school timetabling developed further certain methods were used frequently over the years and new algorithms were derived to solve the school timetabling problem. These include simulated annealing, evolutionary algorithms, tabu search, integer programming, constraint programming, GRASP (Greedy Randomized Search Procedure), tiling algorithms, threshold accepting, bee algorithms, cyclic transfer algorithms and the walk down jump up algorithm. In some cases hybrid approaches, combining two or more methodologies are implemented. Comparative studies, comparing the performance of two or more techniques in solving a particular STP have also been conducted. Solving the school timetabling problem is generally computationally expensive. In order to reduce the runtimes associated with solving the school timetabling problem, distributed methods have also been investigated. This section provides an overview of these different methods and studies. A lookup table, summarizing the different methods, the problem/s that the method has been used to solve and the problem constraints, is provided in Appendix A.

3.1 Bee algorithms

In the study conducted by Lara et al. (2008) a bee algorithm is used to solve the school timetabling problem. Scout bees are used to find feasible solutions. The feasible solutions are then improved by employing collector bees to search the neighborhood for better solutions. The algorithm was used to solve the problem for two real-world school timetabling problems.

3.2 Constraint programming

Marte (2002) applies constraint programming to solve the German gymnasium type school timetabling problem. A constraint model for school timetabling is presented. A hybrid method incorporating chronological backtracking, constraint propagation, depth-first search with restarts and dead-end driven learning is used to solve the problem.

Valouxis and Housos (2003) use constraint programming (CP) in combination with local search to solve the school timetabling problem for Greek high schools. CP is used to find a feasible timetable. The quality of the timetable is then improved using local search until further improvement is not possible. The stopping criterion is a runtime of one hour.

3.3 Constraint satisfaction methods

Meisels et al. (1997) present a constraint network solution to the high school timetabling problem. The problem is formulated as a binary constraint network. A combination of CSP (constraint satisfaction problem) algorithms, namely, the conflict-based backjump and the forward check algorithms, is applied to the network to find a solution.

Abbas and Tsang (2001) show how a constraint satisfaction algorithm, namely, Gashchnig's BackJumping algorithm, can be use to solve the school timetabling problem. Courses that are more "difficult" to schedule are allocated to timeslots first. Course difficulty is a function of various factors such as the duration of the course and the number of students enrolled for the course, amongst others.

3.4 Cyclic transfers

Post et al. (2010a) present a neighborhood structure and a cyclic transfer algorithm for the high school timetabling problem. The algorithm implemented finds negative cycles in the improvement graph. Cyclic exchanges in the form of insertions and ejections are performed. Four data sets from Dutch and English high schools were used to test the algorithm.

3.5 Evolutionary algorithms

Colorni et al. (1990) have applied a genetic algorithm to solve an Italian high school timetabling problem. Each element of the population is a two-dimensional array with the columns representing timetable periods and the rows teachers. The fitness of each timetable is a weighted sum of the constraint violations. Reproduction, mutation, day mutation and crossover are used to create offspring. Filtering is performed to remove any infeasibilities introduced into the timetable by the genetic operators.

Abramson and Abela (1991) use a genetic algorithm to solve the school timetabling problem. A parallel algorithm is applied to speed up the process. Each chromosome consists of n periods and each period contains m tuples. The mutation operator changes the period of a tuple. The crossover operator returns a single child which contains the first fragment of the first parent and the second fragment of the second parent. Crossover may result in the "label replacement problem", i.e. the offspring may contain some duplicated and/or missing genes. A label replacement algorithm is used to rectify this problem. The GA was used to solve nine highly constrained school timetabling problems.

Calderia and Ross (1997) evaluate the use of genetic algorithms (GAs) to solve the STP by applying a GA to a small randomly generated school timetabling problem. An initialization procedure is used to create an initial population of feasible timetables. This procedure creates timetables by allocating lessons to the timetable according to the number of timetable periods available for the lesson and lessons with fewer available periods are scheduled first. If necessary a repair algorithm is used to remove infeasibilities. A GA is used to improve the quality of the initial population. Roulette-wheel selection and an ultra-elitism method are used for selection. Reproduction, mutation and crossover are applied to parents to create the offspring of the next generation.

An evolutionary algorithm is used by Fernandes et al. (1999a, 1999b) to solve the Portuguese high school timetabling problem. This algorithm introduces a new "bad genes" operator which improves the performance of the algorithm. Bad gene mutation and crossover perform swaps and change bad genes, i.e. genes causing hard constraint violations, respectively.

An evolutionary algorithm, which uses local search with mutation operators to further improve timetables, is implemented by Bufe et al. (2001) to solve a German high school timetabling problem.

Di Stephano and Tettamanzi (2001) apply an evolutionary algorithm to the Italian school timetabling problem. An initial population is created by a greedy algorithm which randomly places lessons so as to maintain feasibility. If a feasible period can no longer be found, the lesson is placed in a randomly chosen period. Tournament selection is used to choose parents. The evolutionary algorithm uses uniform crossover and a perturbation operator to produce offspring. The perturbation operator combines the use of an intelligent mutation and an improvement operator. The algorithm was used to generate solutions to an artificial problem modeled on the Italian school system and two real-world Italian school problems.

Filho and Lorena (2001) use a constructive genetic algorithm to solve the school timetabling problem for two Brazilian high schools.

Wilke et al. (2002) use a genetic algorithm to solve the German school timetabling problem. The initial population is comprised of potential timetable solutions, i.e. timetables are directly represented. Each chromosome represents the class timetables for the school. Roulette-wheel selection is used to choose parents. An elitist strategy, namely, copying the best two individuals into the next generation is also employed. In addition to crossover and mutation operators, a number of hybrid operators are applied. If there is no improvement in the fitness of an offspring, a reconfiguration step is performed during which the parameters of the GA are reset.

Bedoya and Santos (2003) evaluate genetic algorithms in solving a test school timetabling problem. Each individual is a two-dimensional array with the rows corresponding to classes and the columns to timetable periods. Roulette wheel selection is used for selection purposes and the mutation operator is used to create the offspring of each generation.

Ciscon et al. (2006) use a memetic algorithm to eliminate open and isolated classes in a timetable for a Brazilian school.

Nurmi and Kyngas (2007) use the h-HCGA genetic algorithm to find solutions to the school timetabling problem. The genetic algorithm uses a greedy hill-climbing mutation operator for regeneration purposes. The mutation operator moves lessons between periods. Periods are chosen so as to reduce the cost of the timetable. An extension of this algorithm employs simulated annealing to choose a period "intelligently". The algorithm has been tested on the data sets in the OR-Library (Beasley 2010) and data obtained from Finnish schools.

A hybrid genetic algorithm (HGA) is used by Yigit (2007) to solve the school timetabling problem for a Technical and Vocational High School in Turkey. In addition to the standard mutation and crossover operators the HGA employs a repair operator. The HGA produced better results than the standard GA.

Beligiannis et al. (2008) use an adaptive evolutionary algorithm to solve the school timetabling problem. Each element of the population is a matrix with the rows corresponding to classes and the columns to periods. Each cell in the matrix stores the teacher that will teach the class in the particular period. Initial studies indicated that crossover was not effective and time consuming and hence it was not used. The period mutation operator swaps the teachers between two timetable periods for a class. The periods chosen for swapping are randomly selected. The bad period mutation operator does not randomly choose the periods, instead the two "most costly" periods in the corresponding teacher timetable are selected. Linear ranking selection is used to choose parents. The best chromosome of each generation is copied into the next generation. This algorithm successfully generated solutions to the Greek high school timetabling problem. This system was extended to include a user-friendly interface so that it could be utilized by non-experts such as school timetable planners (Moschopoulos et al. 2009). In later work Beligiannis et al. (2009) implement a genetic algorithm to solve the same school timetabling problem. The algorithm is adaptive in that it allows the user to assign weights to the different constraints. The fitness of each timetable is the weighted sum of the constraint violations. Roulette wheel selection is used to choose parents to which the crossover and mutation operators are applied, to create the next generation. After crossover is performed, checks are conducted to ensure that co-teaching and subclass violations have not occurred as a result of crossover and corrections are made if necessary.

A genetic algorithm with co-evolution is implemented by Mohammadi and Lucas (2008) to solve the school timetabling problem. This algorithm treats teacher timetables as a different species in the population.

Nurmi and Kyngas (2008) convert the curriculum-based university course timetabling problem for the 2nd International Timetabling Competition (ITC'07) into one for school timetabling and use a genetic algorithm to solve this problem. The GA uses a greedy hill-climbing mutation operator to solve the problem.

Raghavjee and Pillay (2008) apply a genetic algorithm to five generated school timetabling problems (Beasley 2010). The algorithm firstly creates an initial population of timetables using a sequential construction method employing the largest degree heuristic. The mutation operator is used to iteratively refine the initial population. A variation of tournament selection is used to choose the parents of each generation. The algorithm found solutions for all five problems and produced better results than other methodologies applied to the same set of problems. In Raghavjee and Pillay (2009) this system is extended to cater for both hard constraints and soft constraints. A two-phased approach is taken to solve the school timetabling problem. The first phase produces feasible timetables while the second phase focuses on minimizing the soft constraint cost, without introducing infeasibilities. The system was tested on the Greek high school problems made available by Beligiannis et al. (2008). The GA approach was also used to evolve timetables for a South African primary and high school (Raghavjee and Pillay 2010a, 2010b). In this implementation one or more of the low-level construction heuristics, namely, largest degree, split degree, and saturation degree, was/were used to construct timetables during initial population generation of the first phase. The GAs of both phases used a sports tournament selection method to choose parents. Mutation operators with hill-climbing were implemented to create offspring.

Srndic et al. (2009) implement a parallel genetic algorithm, on a Beowulf cluster, to solve a highly constrained test high school timetabling problem. A coarse-grained parallel model is employed. Each element of the population is a two-dimensional array. Tournament selection is used to choose parents. The reproduction, mutation and crossover operators are used to create offspring. The fitness function is the sum of the penalty violations.

3.6 GRASP

Moura and Scaraficci (2010) use GRASP with path-relinking to solve the STP for three Brazilian high schools. GRASP takes a three stage approach to the problem. The first phase ranks lessons. During the second phase the ranking is improved using local search. In the third phase a path-relinking strategy is used to identify optimal solutions. These three phases are repeated a number of times.

3.7 Integer programming

One of the earlier studies employing integer programming to solve the school timetabling problem, is that conducted by Lawrie (1969). A linear programming model of the problem is formulated and a variation of Gomory's method of integer forms is used to produce a timetable outline in which there are no teacher violations. The use of integer programming to solve the school timetabling problem has grown since this early study.

Earlier work by Birbas et al. (1997) uses integer programming to solve the school timetabling problem for Greek high schools. This work is extended further in Birbas et al. (2009) which takes a hybrid approach to solving the problem. The first phase solves the shift assignment problem in which teachers are allocated to shifts. The second phase solves the school timetabling problem. Integer programming is used in both phases. The approach was successfully applied to a secondary Hellenic school.

Papoutsis et al. (2003) use integer programming to represent the school timetabling problem and solve the problem using column generation. The method has been applied to generating timetables for Greek high schools.

Boland et al. (2008) present two integer linear programming models to solve the high school timetabling problem. In this study integer linear programming is used to solve the class blocking problem. This method was tested on data from Australian high schools.

Santos et al. (2008) use mixed integer programming to solve the STP for Brazilian high schools. A cut and column generation algorithm is implemented. The algorithm uses Fenchel cuts.

Ribic and Konjicija (2010) solve the school timetabling problem in two phases, both of which employ integer programming. The first phase focuses on day allocations and the second phase solves the rest of the problem. The approach was applied to a test problem.

3.8 Neural networks

Carrassco and Pato (2004) use neural networks to solve the school timetabling problem. Two neural network simulations are tested, namely, Potts mean-field annealing simulation with continuous Potts neurons and a discrete neural network simulation with discrete user-take-all neurons. Both neural network implementations were tested on the five hard problems made available by Abramson and Smith (Beasley 2010). The discrete neural network performed better than the Potts neural network and a multi-objective genetic algorithm in solving the five problems.

3.9 Simulated annealing

Abramson (1991) applies simulated annealing to the school timetabling problem. The atoms correspond to elements of the timetable and energy to the cost of the timetable. In order to allow for scheduling to be more flexible, assignments are made to room groups instead of individual rooms. If a group of classes must always take place at the same time, the classes should be scheduled as a group instead of individually. The system was tested on randomly generated problems and data from an Australian school. This work is extended further (Abramson et al. 1999) to test six cooling schedules. In this case the SA swaps the contents of timetable periods. Nine generated problems were used for testing purposes. The scheme producing the best results was "reheating as a function of cost".

Melicio et al. (2006) developed the THOR school timetabling tool to solve the STP for Portuguese schools. THOR firstly creates an initial solution using a heuristic construction algorithm. This solution is then improved using fast simulated annealing.

Yongkai et al. (2009) also apply a simulated annealing algorithm with a new neighborhood structure to two real-world high school timetabling problems. The structure performs a sequence of moves instead of a single move.

3.10 Tabu search

One of the earlier studies employing tabu search to solve the school timetabling problem is that conducted by Wright (1996) to induce a timetable for a large comprehensive English school. Since this early study, numerous variations of the tabu search have been implemented to solve the school timetabling problem.

Alvarez-Valdes et al. (2002) use a tabu search to assign teachers to courses when solving the Spanish secondary school problem. A two-phased approach is taken to perform teacher assignments. During the first phase teachers are assigned to subjects and groups. This initial solution is improved using tabu search. The teacher, subject, and group tuples are then allocated to timetable periods in the second phase. The HORES program was used to solve the problem in the second phase. The system was applied to 12 Spanish high school timetabling problems. Teacher allocations made by the tabu search were found to be better than the manual allocations made by the school and random teacher allocations.

Santos et al. (2004, 2005) employ a tabu search with two memory diversification schemes, to derive solutions to the Brazilian high school timetabling problem. A "greedy randomized construction" method is used to create an initial solution. The method schedules tuples with a higher number of unscheduled lessons and fewer available timetable periods first. The initial solution is improved by tabu search. The diversification strategy was tested with transition based long term memory and residence based long term memory. The study showed that the use of a diversification strategy improved the quality of the timetable produced by the tabu search.

In the approach taken by Jacobsen et al. (2006), an initial solution is firstly created using a construction heuristic with a graph coloring algorithm. The initial solution is then improved using tabu search. The system was tested on data from German high schools.

Ohtsubo et al. (2006) use a tabu search to solve the school timetabling problem for junior high schools. Three tabu search algorithms, each differing with respect to the mechanism used to allocate subjects, were implemented and tested.

Bello et al. (2008) treat the school timetabling problem as a graph coloring problem. An adjunct graph is created and colored using an adaptation of the tabu search algorithm for graph vertex coloring (Tabucol), namely, Modified Tabucol (MT). The system was applied to five instances from Brazilian high schools.

Minh et al. (2010) also employ a tabu search to induce school timetables for three Vietnam high schools. Firstly, a greedy search is used to create an initial timetable. This timetable is then improved using tabu search. The moves performed by the tabu search include single moves, swaps, and block moves.

3.11 Threshold accepting

Abboud et al. (1998) describe threshold accepting (TA) as a variation of simulated annealing in which the search is driven by a threshold rather than a temperature. The authors implement a variation of the standard TA algorithm to solve the school timetabling problem. A sequential construction algorithm with backtracking is used to create an initial solution which is improved with threshold accepting. The TA implements a swap operator which swaps timetable periods only if the swap improves the cost of the timetable. The algorithm was tested on 6 generated problems modeled on a real world school timetabling problem. The variation of the TA, the standard TA and simulated annealing produced the same quality timetables for all six problems. However, the variation of the TA had the lowest runtimes, taking less than half the time required by simulated annealing to solve a problem.

3.12 Tiling algorithms

Kingston (2004, 2006) uses a tiling algorithm in combination with hill-climbing to allocate meetings (teacher and class tuples) and an alternating path algorithm for assigning resources to meetings after times are fixed. Meetings are firstly placed onto tiles and then the tiles are timetabled. Resources are then allocated to meetings. Later research conducted by Kingston (2008) investigates the use of a bipartite matching model, namely, global tixel matching, to assign resources such as teachers and rooms to meetings. These algorithms have been applied to Australian high schools.

3.13 Walk down jump up algorithm

Wilke and Killer (2010b) present the walk down jump up algorithm, which combines hill climbing, a jump operator and great deluge to solve the school timetabling problem. This local search algorithm performs two phases. In the first phase an initial solution is created and a fast descending acceptance rate is used to reduce cost. When the cost cannot be reduced further the acceptance rate is increased and the first phase is started again. The algorithm was used to solve a real-world German school timetabling problem.

3.14 Hybrid approaches

Yoshikawa et al. (1994, 1996) take a constraint-based approach, incorporating the use of hill-climbing, to solve the school timetabling problem. A constraint relaxation problem solver is applied to the school timetabling problem. An initial timetable is created using arc consistency. The initial timetable is improved using hill-climbing.

The method employed by Schaerf (1996, 1999b) firstly constructs an initial timetable by randomly assigning teacher-class pairs according to the requirements matrix. The RNA (Randomized Non-Ascendant) search is then applied to improve the initial timetable until no further improvement is possible. At this point the tabu search is applied until there is no improvement. During the RNA phase the hard constraint cost is weighed higher than the soft constraint cost. During the tabu search this weight is "adjusted dynamically". This is referred to as adaptive relaxation. Adaptive relaxation was found to be essential for finding feasible solutions. The RNA and tabu phases are repeated sequentially until there is no further improvement in the quality of the timetable. This hybrid system was used to solve the STP for a randomly generated data set and data sets obtained from two Italian schools.

Alvarez-Valdes et al. (1996) take a three-phase approach to solving the school timetabling problem. In the first phase a parallel heuristic algorithm with priority rules is used to create an initial timetable which is not usually feasible. Phase 2 applies a variation of the standard tabu search to the initial timetable created in phase 1 to produce a feasible timetable. Phase 3 improves the quality of the feasible timetable developed in phase 2. A graph theory approach, using the Floyd-Warshall algorithm, is taken in this phase. This approach was tested on randomly generated problems and data sets from 14 Spanish schools.

Monfroglio (1996) combines the use of genetic algorithms and constrained heuristic search to solve the school timetabling problem.

Drexl and Salewski (1997) take a distributed approach, consisting of two phases, to create a timetable for a German high school. The first phase uses greedy randomized algorithms followed by the application of genetic algorithms in the second phase.

Lohnertz (2002) uses a combination of tabu search and a graph vertex coloring algorithm, namely, a weighted matching algorithm, to solve the German gymnasium school timetabling problem. The tabu search uses the matching algorithm to determine how good a placement each period is. The timetable is displayed for the user to make changes. Exhaustive search is used to identify moves leading to a dead-end and the user is warned accordingly.

Willemen (2002) solves the school timetabling problem for Dutch schools by implementing a tree search algorithm to perform subject-group assignments and a tabu search for timeslot allocation. The tree search algorithm implements a constructive heuristic and dead end handling and uses backtracking and backjumping if necessary.

Kwan et al. (2003) use a hybrid approach to solve a school timetabling problem involving co-teaching and split classes. The method employed combines constraint technology, heuristics, local search and a tabu-list. This approach was found to solve problems effectively within minutes.

De Haan (2004) uses a combination of a beam search and a branch and bound algorithm to solve the Dutch secondary school timetabling problem. The first phase uses a beam search, without backtracking, to assign lessons to day and room parts. In the second phase the problem is formulated as a graph (vertex) coloring problem and the branch and bound algorithm is used to assign lessons to timetable periods and rooms. Landman (2005) extends this research conducted by deHaan to cater for consecutive periods and produce better quality timetables. The use of a shifting algorithm in combination with a day-parts recoloring (vertex) algorithm and a reschedule algorithm is employed to achieve this.

Post and Ruizenaar (2004) use a combination of clustering and branch and bound algorithms to solve the school timetabling problem for a Netherlands secondary school, namely, Kottenpark. Clustering produces clusterschemes composed of clusterlines consisting of optional subjects that can be taught simultaneously.

Souza (2004) firstly applies GRASP to create an initial solution to the school timetabling problem. This initial solution is then improved using tabu search. This hybrid approach was tested on a Brazilian high school problem.

Avella et al. (2007) implement a hybrid system combining simulated annealing and very large neighborhood search (VLNS) to generate a solution for the Italian high school timetabling problem. Simulated annealing creates an initial solution which is improved using VLNS. The VLNS solves an integer linear programming formulation of the problem. The system was tested on the data set made available by Beasley (2010). The hybrid system was found to outperform each of the methods applied separately to the same set of problems.

De Haan et al. (2007a) take a three-phase approach to generating a timetable for a Netherlands secondary school. The first phase creates cluster schemes of subjects, the second phase focuses on assigning lessons to timeslots and rooms, while the third phase employs a tabu search to improve the timetable obtained in phase 2. In later work de Haan et al. (2007b) take a four-phase approach to solving the STP. A preprocessing phase is implemented to cluster events into clusters schemes using a branch-and-bound algorithm. The second and third phases focus on constructing feasible timetables. The second phase assigns lessons to day-parts using a dynamic priority rule. The cluster with the lowest availability is scheduled first. If this leads to unscheduled lessons the heuristic value is recalculated. During the third phase day-parts are allocated to timeslots. A graph (vertex) coloring first-fit heuristic is used for this. The fourth phase uses a tabu search to improve the feasible timetable. The system was successfully applied to a data set from a Netherlands high school.

A genetic algorithm, incorporating the use of neural networks to calculate fitness, is implemented by Zuters (2007) to solve the school timetabling problem. The neural network is used for timetable evaluation and is trained on timetables used in previous years by the school. The neural network is a multilayer perceptron with 600 inputs, one hidden layer with five neurons and one output layer with one neuron. The hybrid system was tested on eight school timetabling problems.

Cedeira-Pena et al. (2008) combine the use of the non-random ascent method (RNA) and genetic algorithm (GA) as a means of solving the school timetabling problem for high schools. The hybrid approach was found to perform better than the RNA and GA when individually applied to the same problem.

Liu et al. (2009) implement two phases to solve the school timetabling problem. The first phase uses simulated annealing to produce a feasible timetable. The SA creates new neighborhoods by performing a series of swaps between timeslots instead of just one swap. The second phase employs a tabu search to minimize the soft constraint cost of the feasible timetable created in the first phase.

3.15 Comparative studies

Colorni et al. (1998) compare the performance of simulated annealing, tabu search with local search and genetic algorithms in solving the school timetabling problem for two Italian high schools. The GA uses reproduction, mutation and crossover. Mutation swaps a set of contiguous genes in the same row. Day mutation swaps two days in the same row. A filtering algorithm is used to convert an infeasible offspring to a feasible one. Instead of creating timetables from scratch the previous year's handmade solution was used as a starting point. Tabu search produced the best results followed by genetic algorithms and simulated annealing.

Smith et al. (2003) use a Hopfield neural network to solve the school timetabling problem. The neural network is used to solve the problem for nine highly constrained school timetabling problems made available by Abramson (Beasley 2010). The performance of the Hopfield neural network on this data set is compared to that of greedy search, simulated annealing and tabu search. The neural network performed better than the other methods, followed by simulated annealing.

Wilke and Ostler (2008) compare the performance of tabu search, simulated annealing, genetic algorithms and branch and bound in solving the school timetabling problem for a German high school. The comparison is performed with respect to computation time and solution quality. Tabu search had the best runtimes but was unable to find feasible solutions. Simulated annealing found feasible solutions. The GA was not able to find feasible solutions. Branch and bound had the highest runtimes and did not produce valid solutions. This work is extended further (Wilke and Killer 2010a) to compare the performance of genetic algorithms, immune systems, harmony search, tabu search, simulated annealing, great deluge, and the walk down jump up algorithm in solving the school timetabling problem. Simulated annealing was found to perform the best in terms of both timetable quality and runtimes.

3.16 Distributed methods

Slechta (2005) proposes a model for decomposing the high school timetabling problem into subproblems, and solving each subproblem in parallel, namely, the multi-resource timetabling problem (MRTP) model. The problem is represented as a graph and a decomposition algorithm is used to divide the graph into subgraphs representing subproblems. Each subproblem is run on a different machine.

Kingston (2007) presents a hierarchical approach that recursively combines smaller timetables into a larger timetable representing the solution to a school timetabling problem. Kingston (2010) extends this work further to produce the KHE general problem solver for school timetabling problems. KHE employs the coarse grained parallel processing model and facilitates the sharing of instances and the independent creation of multiple solutions in parallel.

3.17 Critical summary

This section provides a critical summary of the methods described above for solving the school timetabling problem.

From the above discussion it is evident that various "traditional" optimization and artificial intelligence techniques have been applied to solve the school timetabling problem. Evolutionary algorithms and tabu search appear to be the most popular techniques used to solve this problem. One of the drawbacks of using evolutionary algorithms is the long runtimes needed to find a solution (Abramson and Abela 1991). In addition to these methods, less conventional approaches such as bee algorithms have been successfully applied to solving this problem. A new algorithm, namely, the walk up jump down algorithm developed by Wilke and Killer (2010b) specifically for educational timetabling problems, was used to solve the German school timetabling problem. Neural networks, which are generally more suitable for performing low-level processing rather than high-level reasoning needed to solve school timetabling problems, have not only been used to successfully solve the school timetabling problem, but have also produced better results than other methods, including a genetic algorithm, tabu search and simulated annealing, in solving five "hard" artificially generated school timetabling problems (Smith et al. 2003; Carrassco and Pato 2004).

A majority of the techniques presented employ some construction method to create an initial solution which is then improved by the optimization technique. These construction methods usually allocate tuples or lessons to timetable periods with lessons and tuples that are more difficult to schedule being allocated first. A heuristic is usually used to measure this difficulty. The two heuristics commonly used for this purpose are the number of lessons of a tuple that still have to be scheduled, tuples with more lessons are given priority, and the number of available timetable periods that the lesson or tuple can be allocated to. In the latter case a lower value indicates a tuple or lesson that is more difficult to schedule. In some cases the construction method employs backtracking or a repair mechanism to improve the initial timetable with respect to feasibility. The use of such a construction method appears to be necessary to produce feasible and/or better quality timetables.

As mentioned above, there have been numerous studies employing evolutionary algorithms to solve the school timetabling problem. These studies have revealed that a construction method such as that described above, without backtracking or the use of a repair mechanism, is needed to create the timetables of the initial population. A majority of the studies using evolutionary algorithms have represented timetables directly using a two dimensional array which the reproduction, mutation and crossover operators are applied to, to create offspring. Variations of the genetic operators either incorporating the use of hill-climbing or moving tuples or teachers causing constraint violations have been necessary in some studies to produce solutions. Application of genetic operators, especially crossover, can lead to offspring, not meeting the requirements of the problem, being produced. A repair or filtering algorithm is usually employed to correct this. Elitism has also been used to improve the performance of the algorithm in some cases.

The runtimes of some of these methods are high and multiprocessing or distributed computing is needed in order to reduce runtimes thus enabling researchers to test the different methods more thoroughly. Some models, which can be built on, have already been developed for this purpose (Abramson and Abela 1991; Slechta 2005; Kingston 2007; Srndic et al. 2009; Kingston 2010).

A majority of the research conducted has focused on implementing methods to produce feasible, good quality timetables for specific timetabling problems. However, there has been very little work (Moschopoulos et al. 2009) on the development of interactive systems that can be deployed in schools and can be easily used by administration or teaching staff to create timetables.

In all of the studies described above the techniques employed have successfully solved the school timetabling problem they have been applied to. However, a criticism that can be leveled against a majority of these studies, without in any way undermining the contribution made by them, is that the methods have been applied to one problem only and have not been widely tested. Furthermore, in each of the three comparative studies different methodologies have produced the best results. This can again possibly be attributed to the fact that in each of the studies a single problem was used to test the performance of the different methods. Thus, conclusions cannot be drawn from these comparative studies and the methods need to be evaluated further on additional problems.

4 Data formats and benchmark data sets for the STP

For further advancement in school timetabling research, it is essential that there is a variety of school timetabling data sets publicly available to test and compare the performance of different methodologies in solving the STP. As part of the initiative to develop a publicly available repository of school timetabling problems, the school timetabling research community have also being investigating the use of a uniform language or data format to express school timetabling problems and hence facilitate the exchange of problems between researchers. Section 4.1 presents an overview of research in this area. School timetabling data sets that are already publicly available for use are described in Sect. 4.2.

4.1 Data format for the STP

A fair amount of research has gone into identifying an XML format for expressing data sets. The main aim behind this initiative is to provide a standard format to represent school timetabling problems thereby facilitating public use of these problems. Reis and Oliveira (2000) present the UniLang language for defining school, university course and examination timetabling problems. UniLang provides an evaluator to test whether a given timetable meets all the requirements and constraints of the defined problem. More recently, Ostler and Wilke (2010) have presented an XML format for both timetabling and scheduling problems.

Kingston (2000) has created a language, namely, STTL for specifying high school timetabling problems and evaluating solutions to these problems. Post et al. (2008, 2010b)

Table 1 OR-library data sets

Data set	No. of classes	No. of teachers	No. of venues	No. of periods	No. of tuples to schedule
hdtt4	4	4	4	30	120
hdtt5	5	5	5	30	130
hdtt6	6	6	6	30	180
hdtt7	7	7	7	30	210
hdtt8	8	8	8	30	240

take a similar approach in defining an XML data format and evaluator for high school timetabling problems. To date this format is the most widely used by the school timetabling research community. Twenty problem instances from eight different countries have been expressed in this format and form part of a publicly available archive (details are given in the next section).

4.2 Benchmark data sets

Five of the data sets used in the study conducted by Smith et al. (2003) are available from the OR-Library maintained by Beasley (1990, 2010) at http://people.brunel.ac.uk/~mastjjb/jeb/orlib/tableinfo.html. The data sets have been artificially generated. These problems have been described as "hard" timetabling problems and are more highly constrained than real-world school timetabling problems. The characteristics of problems are listed in Table 1.

These problems have the following hard constraints:

- All class-teacher meetings must be scheduled the required number of times.
- No class clashes.
- No teacher clashes.
- No room clashes.

Methods that have been used to solve this set of problems include greedy search, simulated annealing, tabu search, neural networks (Smith et al. 2003; Carrassco and Pato 2004), a hybrid approach combining simulated annealing and very large neighborhood search (Avella et al. 2007) and evolutionary algorithms (Abramson and Abela 1991; Nurmi and Kyngas 2007; Raghavjee and Pillay 2008).

The seven data sets from Greek high schools used by Beligiannis et al. (2008) are available at http://prlab.ceid.upatras.gr/timetabling/. Each data set contains a requirements matrix specifying how many times each teacher must meet each class. These data sets are described in Table 2.

The hard constraints for the problem are:

- No class clashes.
- No teacher clashes.
- No room clashes.
- All class-teacher meetings must be scheduled the required number of times.
- Split and merger requirements must be met.
- Teachers must be scheduled when available.
- Idle periods for classes should be avoided. If they do occur in a class timetable, they must be scheduled in the last hour of each day.

Table 2 Beligiannis data set

Problem	No. of classes	No. of teachers	No. of venues	No. of periods
1	11	34	–	35
2	11	35	–	35
3	6	19	–	35
4	7	19	–	35
5	6	18	–	35
6	10	34	–	35
7	13	35	–	35

The soft constraints for this problem are:

- Idle periods in teacher timetables must be minimized.
- Lessons taught by teachers should be uniformly distributed over the week.
- If idle periods occur in the teacher timetable, these must also be uniformly distributed.

Evolutionary algorithms (Beligiannis et al. 2008, 2009; Raghavjee and Pillay 2009) have been used to solve this problem.

Post et al. (2008, 2010b) have initiated a project to facilitate the easy exchange of benchmark school timetabling data sets and so promote research in this domain. Post et al. propose an XML format to describe school timetabling problems and have setup a website for access to and the submission of problems, namely, http://www.utwente.nl/ctit/hstt/. There are 16 data sets from 8 different countries, namely, Australia (3), Brazil (5), England (1), Finland (3), Greece (3), Italy (1), the Netherlands (3), and South Africa (1) available from the website. In addition to this the OR-Library data sets and other artificially created problems used by Abramson and Abela (1991) can also be downloaded from this site. The school timetabling problem for each country, based on the specific education system implemented, is also presented. The website provides the following for each data set:

- Characteristics of the data set, e.g. number of teachers, number of rooms, number of classes, etc.
- Hard and soft constraints for the problem represented by each data set.
- Known solutions and the hard and soft constraint cost of these solutions.

A description of these data sets is provided in Table 3. All these data sets and the solutions provided are represented in the XHSTT format.

A list of hard and soft constraints for each of these data sets can be accessed from http://www.utwente.nl/ctit/hstt/datasets/.

5 An industry perspective

A survey was conducted into the current state of software development for school timetable construction. The survey has revealed that there are numerous packages developed internationally for creating school timetables. Focus is on providing an easy to use, interactive piece of software that can be used as a management tool, rather than developing intelligent optimization techniques for solving the school timetabling problem. The main constraint catered for is no clashes (class, teacher and room). Additional constraints addressed by some of these software packages include:

Table 3 Post data set

Problem	Country	Times	Teachers	Rooms	Classes
TES99	Australia	30	37	26	13
BGHS98	Australia	40	56	45	30
SAHS96	Australia	60	43	36	20
Instance 1	Brazil	25	8	–	3
Instance 4	Brazil	25	23	–	12
Instance 5	Brazil	25	31	–	13
Instance 6	Brazil	25	30	–	14
Instance 7	Brazil	25	33	–	20
St. Paul	England	27	68	67	67
Finland Artificial School	Finland	20	22	12	13
Finland High School	Finland	35	18	13	102
Finland Secondary School	Finland	35	25	25	14
Greek High School 1	Greece	35	29	–	66
Third High School Patras 2010	Greece	35	29	–	57
Third High School Preveza 2008	Greece	35	29	–	53
Italy Instance 1	Italy	36	13	–	3
Genpro	Netherlands	37	78	42	26
Kottenpark 2003	Netherlands	38	75	41	18
Kottenpark 2005	Netherlands	37	78	42	26
Lewitt 2009	South Africa	148	19	2	16

- Catering for class and teacher unavailabilities.
- Minimizing the number of idle periods in teacher timetables.
- Catering for travel between school locations.
- No consecutive periods on the same day for lessons on the same subject.
- The use of specialized rooms.
- Catering for teacher preferences.
- Catering for splits and mergers.
- Uniform distribution of lessons for teachers over the teaching period.
- Uniform distribution of lessons for classes over the teaching period.
- Catering for teacher workload limits.
- Consecutive periods for certain subjects.
- Pre-scheduled lessons, i.e. certain lessons must take place during specified periods.
- Free periods for teachers.

- Room preferences.
- Daily and/or weekly lesson limits for classes and teachers.

Appendix B lists some of these software packages, together with the country of origin, and whether the product caters for all educational timetabling or school timetabling only. While academia focuses on developing robust, intelligent methods to solve the school timetabling problem, industry appears to aim at developing interactive, easy to use tools that can utilized by teaching and administration staff at schools to construct timetables that meet their needs. This gap between industry and academia needs to be bridged in order to produce efficient and robust school timetabling software that can be easily used by non-experts.

6 Future research directions for school timetabling

It is evident from Sect. 5 that a variety of school timetabling data sets is now available. This will facilitate a comparison of different methodologies in solving the school timetabling problem and promote further development of the school timetabling domain. Furthermore, a more thorough evaluation of the ability of techniques in solving this problem can be performed.

A majority of the methods described in this paper firstly implement a construction algorithm which uses a heuristic to sort class-teacher (or class-teacher-room) tuples in order of difficulty to schedule, and allocates each tuple in sequence. An area which has not been investigated as thoroughly as in the domain of university examination timetabling (Carter et al. 1996) is different heuristics that can be used to estimate the difficulty of scheduling a tuple. Section 3.17 describes two such heuristics that have generally been used by construction methods. The derivation and evaluation of other heuristics for this domain needs to be examined.

A majority of the methods implemented have been used to solve a particular school timetabling problem and there does not appear to be work done in developing methods that generalize well. The aim of hyper-heuristics is to generalize well over the problems in a particular domain, rather than producing the best result for one or more problem sets (Burke et al. 2003). Hyper-heuristics search a heuristic space rather than a solution space. The heuristic space is usually comprised of combinations of low-level heuristics which can be constructive or perturbative (Pillay and Banzhaf 2009). While there has been research into the effectiveness of hyper-heuristics for university course and examination timetabling (Burke et al. 2007), this has not been studied for school timetabling. An initial investigation into the use of constructive hyper-heuristics for school timetabling has been conducted by Pillay (2010). This preliminary study implements an evolutionary algorithm based hyper-heuristic for a generated problem. However, a lot more work still needs to be done before any significant contribution can be made.

Building school timetabling systems that can be deployed in schools and are not just research tools is important to the development of the field. Such a system must allow for timetable reconstruction without much effort. The user must be able to easily make minor changes to the constraints, change the weighting of constraints, make manual changes and request a new timetable taking these into consideration. This is referred to as post-publication timetabling and was introduced by Cumming and Paetcher (2000). School timetable construction software developed by industry appears to focus on these aspects. Links need to be fostered between industry and academia in order to develop robust and efficient timetable software that meet the needs of non-experts.

The application of some search techniques, e.g. evolutionary algorithms, can be time consuming. There have been earlier studies investigating the use of parallel processing to decrease the runtime of timetabling systems (Abramson 1991; Abramson and Abela 1991). Slechta (2005) and Kingston (2010) propose distributed approaches to the school timetabling problem. Given the emergence of multi-core processors, the effectiveness of parallel processing in improving runtimes of school timetabling systems needs to be studied further.

7 Conclusion

Research in the domain of school timetabling has not advanced as rapidly as other spheres of educational timetabling. This has been attributed to most studies in this domain being done in isolation of each other and the lack of a variety of benchmark problems to evaluate methods thoroughly and perform comparative studies on. The definition of the school timetabling problem varies drastically from one study to the next. This paper has attempted to provide a standardized definition of the problem in terms of problem requirements, hard constraints and soft constraints. The paper provides an overview of methodologies employed to solve the school timetabling problem. In addition to this the paper provides details of publicly available school timetabling data sets. Finally, the paper proposes future directions of research in this field, namely, the derivation of new heuristics, an evaluation of hyper-heuristics in this domain, collaboration between industry and academia in developing usable systems that promote timetable reconstruction and the use of parallel processing to improve the runtimes of school timetabling systems.

Acknowledgements The author would like to thank the reviewers for their helpful comments and suggests.

Appendix A: Solving the school timetabling problem

Table 4 lists the different methods that have been applied to the school timetabling problem, the studies implementing these methods, the problems the methods were used to solve and the constraints for these problems. Note that for both the hybrid methods and comparative studies, column 2 also lists the methods used.

Table 4 An overview of methods used to solve the school timetabling problem

Method	Studies	Problems	Constraints
Bee algorithms	Lara et al. (2008)	Two real-world high school problems	PR1, PR2, NC1, NC2
Constraint programming	Marte (2002)	German gymnasium school	PR1, PR2, NC1, NC2, RU2, W1, W3, PD1, PD3, L1, L3
	Valouxis and Housos (2003)	Greek high school	PR1, PR2, NC1, NC2, NC3, PD1, PD3, PD4, P2
Constraint satisfaction methods	Meisels et al. (1997)	–	PR1, PR2, NC1, NC2
	Abbas and Tsang (2001)	–	PR1, PR2, NC1, NC2
Cyclic transfers	Post et al. (2010a)	German and English secondary schools	PR1, PR2, NC1, NC2, RU2, RU8, W1, W3, PD1, PD2, L5

Table 4 (*Continued*)

Method	Studies	Problems	Constraints
Evolutionary algorithms	Colorni et al. (1990)	Italian high school	PR1, PR2, NC1, NC2, NC3,
	Abramson and Abela (1991)	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, NC3
	Calderia and Ross (1997)	Generated problem	PR1, PR2, PR3, NC1, NC2, NC3, RU2, PD1, PD2, PD3, L1
	Fernandes et al. (1999a)	Portuguese high school	PR1, PR2, NC1, NC2, NC3, PD3
	Fernandes et al. (1999b)	Portuguese high school	PR1, PR2, NC1, NC2, NC3, PD3
	Bufe et al. (2001)	German high school	PR1, PR2, NC1, NC2
	Di Stephano and Tettamanzi (2001)	Italian school	PR1, PR2, NC1, NC2, RU1, RU2, RU4, RU6, RU7, L1, L2, PD1, PD2, PD3, L1, L3
	Filho and Lorena (2001)	Brazilian high school	PR1, PR2, NC1, NC2, NC3, RU5, PD2, P2
	Wilke et al. (2002)	German school	PR1, PR2, PR3, PR4, NC1, NC2, RU2, PD1, P2, L1, L4
	Bedoya and Santos (2003)	–	PR1, PR2, NC1, NC2
	Ciscon et al. (2006)	Brazilian school	PR1, PR2, NC1, NC2, PD1, PD3
	Nurmi and Kyngas (2007)	Generated problems (Beasley 2010) Finnish schools	PR1, PR2, NC1, NC2, NC3, W1, W2, W3, PD1, PD2, PD4, L3
	Yigit (2007)	Technical and vocational Turkish high schools	PR1, PR2, NC1, NC2
	Beligiannis et al. (2008)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Mohammadi and Lucas (2008)	–	PR1, PR2, PR3, NC1, NC2
	Nurmi and Kyngas (2008)	ITC'07 data set (McCollum et al. 2009)	PR1, PR2, NC1, NC2, NC3, RU2, PD1
	Raghavjee and Pillay (2008)	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, NC3
	Beligiannis et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Moschopoulos et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Raghavjee and Pillay (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Srndic et al. (2009)	–	PR1, PR2, NC1, NC2
	Raghavjee and Pillay (2010a)	South African high school	PR1, PR2, NC1, NC2, P2, L1, L4
	Raghavjee and Pillay (2010b)	South African primary school	PR1, PR2, NC1, NC2, P2, L1, L4
		South African high school	PR1, PR2, NC1, NC2, RU5, PD3, L1, L3

Table 4 (*Continued*)

Method	Studies	Problems	Constraints
GRASP	Moura and Scaraficci (2010)	Brazilian high school	PR1, PR2, NC1, NC2
Integer programming	Lawrie (1969)	–	PR1, PR2, NC1, NC2
	Birbas et al. (1997)	Greek high school	PR1, PR2, NC1, NC2, W1, W3, RU2, PD1, PD2, PD3, L1
	Papoutsis et al. (2003)	Greek high school	PR1, PR2, NC1, NC2, W1, PD1, PD2, PD3, PD4, P2
	Boland et al. (2008)	Australian high school	PR1, PR2, NC1, NC2, RU2, RU4, PD1
	Santos et al. (2008)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, L3
	Birbas et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, P2, L1, L3, L5
	Ribic and Konjicija (2010)	–	PR1, PR2, PR3, NC1, NC2, RU2, RU5, RU8, PD1, L1
Neural networks	Carrassco and Pato (2004)	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2
Simulated annealing	Abramson and Abela (1991)	Generated problems Australian school	PR1, PR2, NC1, NC2, NC3, RU2, W3, L3
	Abramson et al. (1999)	Generated problems	PR1, PR2, NC1, NC2, NC3
	Melicio et al. (2006)	Portuguese school	PR1, PR2, NC1, NC2
	Yongkai et al. (2009)	Two real-world high school problems	PR1, PR2, NC1, NC2
Tabu search	Wright (1996)	English comprehensive school	PR1, PR2, NC1, NC2, RU6, PD1, PD2, L1, L2, L3
	Alvarez-Valdes et al. (2002)	Spanish high school	PR1, PR2, NC1, NC2
	Santos et al. (2004)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD4, L3
	Santos et al. (2005)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD4, L3
	Jacobsen et al. (2006)	German high school	PR1, PR2, NC1, NC2, NC3, RU2, RU7, RU8, W1, W2, PD1, PD3, PD4, P1, L1, L3
	Ohtsubo et al. (2006)	Junior high school	PR1, PR2, NC1, NC2
	Bello et al. (2008)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD2, PD3, PD4, W1, L3
	Minh et al. (2010)	Vietnam high school	PR1, PR2, NC1, NC2
Threshold accepting	Abboud et al. (1998)	Generated problems	PR1, PR2, NC1, NC2, RU2, PD3, PD4, P1

Table 4 (*Continued*)

Method	Studies	Problems	Constraints
Tiling algorithms	Kingston (2004)	Australian high school	PR1, PR2, NC1, NC2
	Kingston (2006)	Australian high school	PR1, PR2, NC1, NC2, NC3, RU5, W1, PD2, PD3, L3, L4
	Kingston (2008)	Australian high school	–
Walk Down Jump Up Algorithm	Wilke and Killer (2010b)	German high school	PR1, PR2, NC1, NC2, NC3, PD1
Hybrid approaches	Yoshikawa et al. (1994) • Arc consistency • Hill climbing	–	PR1, PR2, NC1, NC2, PD1, PD3, L5
	Yoshikawa et al. (1996) • Arc consistency • Hill climbing	–	PR1, PR2, NC1, NC2, RU2, RU3, W1, PD3
	Schaerf (1996) • Tabu search • Randomize non ascendant	Italian schools	PR1, PR2, NC1, NC2, RU1, W1, PD2, PD3, P2, L3, L4
	Alvarez-Valdes et al. (1996) • Tabu search • Floyd-Warshall algorithm	Spanish schools	PR1, PR2, NC1, NC2, RU2, RU7, RU8, PD1, PD2, PD3, L1, L4
	Monfroglio (1996) • Genetic algorithms • Constrained heuristic search	–	PR1, PR2, NC1, NC2
	Drexl and Salewski (1997) • Greedy randomized algorithms • Genetic algorithms	German high school	PR1, PR2, PR3, NC1, NC2, NC3, RU2, PD1, L1, L5
	Schaerf (1999b) • Tabu search • Randomize non ascendant	Italian high school	PR1, PR2, NC1, NC2, RU1, W1, PD2, PD3, P2, L3, L4
	Lohnertz (2002) • Tabu search • Graph coloring matching algorithm	German gymnasium school	PR1, PR2, NC1, NC2, RU2, L3
	Willemen (2002) • Tree search algorithm • Tabu search	Dutch school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD3, L3
	Kwan et al. (2003) • Constraint technology • Heuristics • Local search with a tabu list	–	PR1, PR2, NC1, NC2
	de Haan (2004) • Beam search • Branch and bound algorithm	Kottenpark: Netherlands secondary school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD1, PD2, PD3, P2, L1, L2, L4, L5

Table 4 (*Continued*)

Method	Studies	Problems	Constraints
	Landman (2005) • Beam search • Branch and bound algorithm • Shifting algorithm • Re-coloring algorithm	Dutch school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD1, PD2, PD3, PD4, P2, L1, L4
	Post and Ruizenaar (2004) • Clustering algorithm • Branch and bound algorithm	Kottenpark: Netherlands secondary school	PR1, PR2, NC1, NC2, NC3, RU2, RU4, RU6, RU8, PD1, PD2, PD3, P2, L1, L2, L4, L5
	Souza (2004) • GRASP • Tabu search	Brazilian high school	PR1, PR2, NC1, NC2, RU2, W1, PD3, L3
	Avella et al. (2007) • Very large neighborhood search • Simulated annealing	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, RU2, W3, PD1, PD2, PD3, PD4, P2, L1
	de Haan et al. (2007a) • Clustering algorithm • Tabu search	Netherlands secondary school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, L3
	de Haan et al. (2007b) • Branch and bound algorithm • Dynamic priority rule • First-fit heuristic • Tabu search	Netherlands high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, L3
	Zuters (2007) • Genetic algorithms • Neural networks	–	PR1, PR2, NC1, NC2, PD1, PD3, L2
	Cedeira-Pena et al. (2008) • Random ascent method • Genetic algorithms	–	PR1, PR2, NC1, NC2
	Liu et al. (2009) • Simulated annealing • Tabu search		PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, P2
Comparative studies	Colorni et al. (1998) • Simulated annealing • Tabu search • Genetic algorithms	Italian high school	PR1, PR2, NC1, NC2, PD1, PD2
	Smith et al. (2003) • Hopfield neural network • Greedy search • Simulated annealing • Tabu search	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, NC3
	Wilke and Ostler (2008) • Tabu search • Simulated annealing • Genetic algorithms • Branch and bound algorithms	German high school	PR1, PR2, NC1, NC2, NC3, RU4

Table 4 (*Continued*)

Method	Studies	Problems	Constraints
	Wilke and Killer (2010b) • Genetic algorithms • Immune systems • Harmony search • Tabu search • Simulated annealing • Great deluge • Walk down jump up	German high school	PR1, PR2, NC1, NC2, NC3, RU4
Distributed methods	Slechta (2005)	–	PR1, PR2, NC1, NC2
	Kingston (2010)	–	PR1, PR2, NC1, NC2, NC3, RU5, W1, PD2, PD3, L3, L4

Appendix B: Commercial and freeware software for school timetable construction

Table 5 lists commercial and freeware packages for school timetable construction, the country that the package was developed in, and whether the software is specific to the domain of school timetabling or can be used to generate timetables for other educational institutions as well.

Table 5 Software packages for school timetable construction

Package	Country	Institutions
ABC Timetable (freeware)	Germany	Schools
Asc Automatic Timetabler	India	Schools
aSc Timetable Scheduler	Slovakia	Primary and secondary schools
EduSwift	India	Schools
eduTimer	India	Schools
edval	Australia	Australian high schools
FirstClass	Australia	Schools
gp-Unitis	Hong Kong and Switzerland	Universities, schools and colleges
Keith Johnson's Timetabler	UK	Schools
Mimosa	Finland	Schools
Modelino	Romania	Universities, schools and colleges
Paralax Planning Solutions	Netherlands	Schools
Prime Timetable	USA	Schools
School Scheduler	South Africa	Schools
School Timetable	Poland	Schools
Sign Up Genius	USA	Schools
Supertime 2000	India	Colleges and schools
Timechart	Australia	Colleges and schools
Time Design	South Africa	Schools
TimeFinder	Germany	High schools and universities
Timetabler	Australia	Schools
Timetabler Plus	Malaysia and Singapore	Universities, schools, polytechs and colleges
Untis Timetabling Program	Austria	Universities and schools
Visual Classroom Scheduler	Australia	Universities, schools and colleges
Visual Timetabling	France	Universities, schools and colleges
Wise Timetable	Slovenia	Universities, schools and colleges

References

Abbas, A. M., & Tsang, E. P. K. (2001). Constraint-based timetabling—a case study. In *Proceedings of ACS/IEEE international conference on computer systems and applications (AICCSA'01)* (p. 67). Los Alamitos: IEEE Comput. Soc.

Abboud, N., Sakawa, M., & Inuiguchi, M. (1998). School scheduling using threshold accepting. *Cybernetics and Systems*, *29*(6), 593–611.

Abramson, D. (1991). Constructing school timetables using simulated annealing: sequential and parallel algorithms. *Management Science*, *37*(1), 98–113.

Abramson, D., & Abela, J. (1991). A parallel genetic algorithm for the solving the school timetabling problem. In *Proceedings of the fifteenth Australian conference: division of information technology, C.S.I.R.O.* (pp. 1–11).

Abramson, D., Krishnamoorthy, M., & Dang, H. (1999). Simulated annealing cooling schedules for the school timetabling problem. *Asia-Pacific Journal of Operational Research*, *16*(1), 1–22.

Alvarez-Valdes, R., Martin, G., & Tamarit, J. M. (1996). Constructing good solutions for the Spanish school timetabling problem. *Journal of the Operational Research Society*, *47*(10), 1203–1215.

Alvarez-Valdes, R., Parreno, F., & Tamarit, J. M. (2002). A tabu search algorithm for assigning teachers to courses. *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research*, *10*(2), 239–259.

Appleby, J. S., Blake, D. V., & Newman, E. A. (1961). Techniques for producing school timetables on computer and their application to other scheduling problems. *The Computer Journal*, *3*(4), 237–245.

Aust, R. J. (1976). An improvement algorithm for school timetabling. *Computer Journal*, *19*(4), 339–343.

Avella, P., D'Auria, B., Salerno, S., & Vasil'ev, I. (2007). A computational study of local search algorithms for Italian high school timetabling. *Journal of Heuristics*, *13*(6), 543–556.

Beasley, J. (1990). OR library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, *41*(11), 1069–1072.

Beasley, J. F. (2010). OR library. http://people.brunel.ac.uk/mastjjb/jeb/orlib/tableinfo.html. Last accessed 1 February 2010.

Bedoya, C. F., & Santos, M. (2003). *A non-standard genetic algorithm approach to solve constrained school timetabling problems*. Eurocast, 26–37.

Beligiannis, G. N., Moschopoulos, C. N., Kaperonis, G. P., & Likothanassis, S. D. (2008). Applying evolutionary computation to the school timetabling problem: the Greek case. *Computers and Operations Research*, *35*, 1265–1280.

Beligiannis, G. N., Moschopoulos, C. N., & Likothanassis, S. D. (2009). A genetic algorithm approach to school timetabling. *Journal of the Operational Research Society*, *60*(1), 23–42.

Bello, G. S., Rangel, M. C., & Boeres, M. C. S. (2008). An approach for the class/teacher timetabling problem. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT2008)*. http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Boeres-WA2b.pdf. Last accessed 05/02/10.

Birbas, T., Daskalaki, S., & Housos, E. (1997). Timetabling for Greek high schools. *Journal of the Operational Research Society*, *48*(2), 1191–1200.

Birbas, T., Daskalaki, S., & Housos, E. (2009). School timetabling for quality student and teacher schedules. *Journal of Scheduling*, *12*(2), 177–197.

Boland, N., Hughes, B. D., Merlot, L. T. G., & Stuckey, P. J. (2008). New integer linear programming for course timetabling. *Computers and Operations Research*, *35*, 2209–2233.

Bufe, M., Fischer, T., Gubbels, H., Hacker, C., Hasprich, O., Scheibel, C., Weicker, K., Weicker, N., Wenig, M., & Wolfangel, C. (2001). Automated solution of highly constrained school timetabling problem. In *Proceedings of the EvoWorkshops on the applications of evolutionary computing* (pp. 431–440).

Burke, E., Hart, E., Kendall, G., Newall, J., Ross, P., & Schulenburg, S. (2003). Hyper-heuristics: an emerging direction in modern research. In *Handbook of metaheuristics* (pp. 457–474). Dordrecht: Kluwer Academic. Chap. 16.

Burke, E. K., de Wera, D., & Kingston, J. F. (2004). Applications of timetabling. In *Handbook of graph theory* (pp. 445–474). London: Chapman & Hall, Chap. 5.6.

Burke, E. K., McCollum, B., Meisels, A., Petrovic, S., & Qu, R. (2007). A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research (EJOR)*, *176*, 177–192.

Calderia, J. P., & Ross, A. C. (1997). School timetabling using genetic search. In *Proceedings of the international conference on the practice and theory of automated timetabling (PATAT'97)* (pp. 115–122).

Carrassco, M. P., & Pato, M. V. (2004). A comparison of discrete and continuous neural network approaches to solve the class/teacher timetabling problem. *European Journal of Operational Research*, *153*, 65–79.

Carter, M. W., & Tovey, C. A. (1992). When is the classroom assignment problem hard? *Operations Research*, *40*(S1), 28–29.

Carter, M. W., Laporte, G., & Lee, S. Y. (1996). Examination timetabling: algorithmic strategies and applications. *The Journal of the Operational Research Society*, *47*(3), 373–383.

Carter, M. W., & Laporte, G. (1997). Recent developments in the practical course timetabling. In E. Burke & M. Carter (Eds.), *Lecture notes in computer science: Vol. 1408. Practice and theory of automated timetabling II* (pp. 3–19). Berlin: Springer.

Cedeira-Pena, A., Carpente, L., Farina, A., & Seco, D. (2008). New approaches for the school timetabling problem. In *Proceedings of the 7th Mexican conference on artificial intelligence (MICAI 2008)* (pp. 261–267).

Ciscon, L. A., De Oliveira, H. C. B., Andrade, M. C. A., Alvarenga, G. B., & Esmin, A. A. A. (2006). The school timetabling problem: a focus on elimination of open periods and isolated classes. In *Proceedings of the sixth international conference on hybrid intelligent systems and fourth conference on neuro-computing and evolving intelligence, HIS-NCEI, 2006* (p. 70). New York: IEEE Press.

Colorni, A., Dorigo, M., & Maniezzo, V. (1990). Genetic algorithms: a new approach to the timetable problem. *NATO ASI Series*, *F82*, 235–239.

Colorni, A., Dorigo, M., & Maniezzo, V. (1998). Metaheuristics for high school timetabling. *Computational Optimization and Applications*, *9*, 275–298.

Cooper, T. B., & Kingston, J. H. (1993). The solution of real instances of the timetabling problem. *The Computer Journal*, *36*(7), 645–653.

Cooper, T. B., & Kingston, J. H. (1995). A program for constructing high school timetables. In *Proceedings of the first international conference on the practice and theory of automated timetabling*.

Cooper, T. B., & Kingston, J. H. (1996). The complexity of timetable construction problems. In *Selected papers from the first international conference on the practice and theory of automated timetabling* (pp. 283–295).

Cumming, A., & Paetcher, B. (2000). Post-publication timetabling. In *Proceedings of the 3rd international conference on the practice and theory of automated timetabling (PATAT 2000)* (pp. 107–108).

de Gans, O. B. (1981). A computer timetabling system for secondary schools in the Netherlands. *European Journal of Operational Research*, *7*(2), 175–182.

de Haan (2004). *Timetabling in dutch secondary schools*. Masters Thesis, University of Twente.

de Haan, P., Landman, R., Post, G., & Ruizenaar, H. (2007a). A case study for timetabling in Dutch high schools. *Practice and Theory of Automated Timetabling*, *IV*, 267–279.

de Haan, P., Landman, R., Post, G., & Ruizenaar, H. (2007b). A four-phase approach to a timetabling problem for secondary schools. In E. K. Burke & H. Rudova (Eds.), *Lecture notes in computer science: Vol. 3867. The practice and theory of automated timetabling VI* (pp. 267–279). Berlin: Springer.

de Werra, D. (1971). Construction of school timetables by flow methods. *Information Systems and Operational Research*, *9*(1), 12–22.

Di Stephano, C., & Tettamanzi, A. G. B. (2001). An evolutionary algorithm for solving the school timetabling problem. In *Proceedings of the EvoWorkshops 2001* (pp. 659–672). Berlin: Springer.

Drexl, A., & Salewski, F. (1997). Distribution requirements and compactness constraints in school timetabling. *European Journal of Operational Research*, *102*(1), 193–214.

Eikelder, H. M. M., & Willemen, R. J. (2001). Some complexity aspects of secondary school timetabling problems. In *Lecture notes in computer science: Vol. 2079/2001. Practice and theory of automated timetabling III* (pp. 18–27).

Even, S., Itai, A., & Shamir, A. (1976). On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, *5*(4), 691–703.

Fernandes, C., Caldeira, J. P., Melicio, F., & Rosa, A. (1999a). High school weekly timetabling by evolutionary algorithms. In *Proceedings of the ACM symposium on applied computing* (pp. 344–350).

Fernandes, C., Caldeira, J. P., Melicio, F., & Rosa, A. (1999b). Evolutionary algorithm for school timetabling. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela & R. D. Smith (Eds.), *Proceedings of the genetic and evolutionary computation conference* (p. 1777).

Filho, G. R., & Lorena, L. A. N. (2001). A constructive evolutionary approach to school timetabling. In *Lecture notes in computer science: Vol. 2037. Proceedings of the EvoWorkshops on applications of evolutionary computing* (pp. 130–139). Berlin: Springer.

Grobner, M., Wilke, P., & Buttcher, S. (2003). A standard framework for timetabling problems. In E. K. Burke & P. De Causmaecker (Eds.), *Lecture notes in computer science: Vol. 2740. Proceedings of the international conference on the practice and theory of automated timetabling (PATAT 2002)* (pp. 24–38). Berlin: Springer.

Hertz, A., & Robert, V. (1998). Constructing a course schedule by solving a series of assignment type problems. *European Journal of Operational Research*, *108*, 585–603.

Jacobsen, F., Bortfeldt, A., & Gehring, H. (2006). Timetabling at German secondary schools: tabu search versus constraint programming. In E. K. Burke & H. Rudova (Eds.), *Proceedings of the international conference on the practice and theory of automated timetabling (PATAT 2006)* (pp. 439–442). ISBN 80-210-3726-1.

Junginger, W. (1986). Timetabling in Germany—a survey. *Interfaces*, *16*(4), 66–74.

Kingston, J. H. (2000). Modelling timetable problems with STTL. In E. Burke & W. Erben (Eds.), *PATAT'00: selected papers from the third international conference on the practice and theory of automated timetabling* (pp. 309–321). Berlin: Springer.

Kingston, J. H. (2004). A tiling algorithm for high school timetabling. In *Lecture notes in computer science: Vol. 3616/2005*. *Practice and theory of automated timetabling V* (pp. 208–225). Berlin: Springer.

Kingston, J. H. (2006). The KTS high school timetabling systems. In E. K. Burke & H. Rudova (Eds.), *Proceedings of the international conference on the practice and theory of automated timetabling (PATAT 2006)* (pp. 181–195). ISBN 80-210-3726-1.

Kingston, J. H. (2007). Hierarchical timetable construction. In E. Burke & H. Rudova (Eds.), *Lecture notes in computer science: Vol. 3867*. *PATAT VI* (pp. 294–307).

Kingston, J. H. (2008). Resource assignment in high school timetabling. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT2008)*. http://w1.cirrelt.ca/~patat2008/PATAT7PROCEEDINGS/Papers/Kingston-WA2b.pdf. Last accessed 05/02/10.

Kingston, J. H. (2010). Solving the general high school timetabling problem. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)* (pp. 517–518).

Kwan, A. C. M., Chung, K. C. K., Yip, K. K. K., & Tam, V. (2003). An automated school timetabling system using hybrid intelligent techniques. In *Lecture notes in computer science* (Vol. 2871, pp. 124–134).

Kwok, L., Kong, S., & Kam, Y. (1997). Timetabling in Hong Kong secondary schools. *Computer Education*, *28*(3), 173–183.

Landman, R. (2005). *Creating good-quality timetables for Dutch high schools*. Masters Thesis, University of Twente, the Netherlands.

Lara, C., Flores, J. J., & Calderon, F. (2008). Solving a school timetabling problem using a bee algorithm. In *Lecture notes in computer science* (Vol. 5317, pp. 664–674).

Lawrie, N. L. (1969). An integer linear programming model of a school timetabling problem. *The Computer Journal*, *12*(4), 307–316.

Lion, J. (1966). Matrix reduction using the Hungarian method for the generation of school timetables. *Communications of the ACM*, *9*(5), 349–354.

Lion, J. (1967). The Ontario school timetabling problem. *The Computer Journal*, *10*(1), 14–21.

Liu, Y., Zhang, D., & Leung, S. C. H. (2009). A simulated annealing algorithm with a new neighbourhood structure for the timetabling problem. In *GEC'09: proceeding of the first ACM/SIGEVO summit on genetic and evolutionary computation* (pp. 381–386).

Lohnertz, M. (2002). A timetabling system for the German gymnasium. In *Proceedings of the fourth international conference on the practice and theory of automated timetabling*.

Marte, M. (2002). *Models and algorithms for school timetabling—a constraint-programming approach*. Phd Thesis, University of Munchen.

Marte, M. (2006). Towards constraint-based grammar school timetabling. http://www3.deis.unibo.it/Events/Deis/Workshops/PapersCPAIOR99/21final.ps. Last accessed 12 February 2010.

McCollum, B., McMullan, P., Paechter, B., Lewis, R., Schaerf, A., Di Gaspero, L., Parkes, A. J., Qu, R., & Burke, E. K. (2009). Setting the research agenda in automated timetabling: the second international timetabling competition. *INFORMS Journal on Computing*. doi:10.1287/ijoc.1090.0320.

Melicio, F., Calderia, J. P., & Rosa, A. (2006). THOR: a tool for school timetabling. In E. K. Burke & H. Rudova (Eds.), *Proceedings of the 6th international conference on the practice and teaching of automated timetabling (PATAT 2006)* (pp. 532–535).

Meisels, A., Ell-sana, J., & Gudes, E. (1997). Decomposing and solving timetabling constraint networks. *Computational Intelligence*, *13*(4), 486–505.

Minh, K. N. T. T., Thanh, N. D. T., Trang, K. T., & Hue, N. T. T. (2010). Using tabu search for solving a high school timetabling problem. *Studies in Computational Intelligence*, *283*, 305–313.

Mohammadi, M. S., & Lucas, C. (2008). Cooperative co-evolution for school timetabling problem. In *Proceedings of the 7th international conference on cybernetic intelligent systems (CIS 2008)* (pp. 1–7).

Monfroglio, A. (1996). Timetabling through constrained heuristic search and genetic algorithms. *Software—Practice and Experience*, *26*(3), 251–279.

Moschopoulos, C. N., Alexakos, C. E., Dosi, C., Beligiannis, G. N., & Likothanassis, S. D. (2009). A user-friendly evolutionary tool for high-school timetabling. *Studies in Computational Intelligence*, *166*, 149–162.

Moura, A. V., & Scaraficci, R. A. (2010). A GRASP strategy for a more constrained school timetabling problem. *International Journal of Operational Research*, *7*(2), 152–170.

Nurmi, K., & Kyngas, J. (2007). A framework for school timetabling problem. In *Proceedings of the 3rd multidisciplinary international scheduling conference: theory and application*. http://www.mistaconference.org/2007/papers/A%20Framework%20.for%20School%20Timetabling%20Problem.pdf. Last accessed 15 February 2012.

Nurmi, K., & Kyngas, J. (2008). A conversion scheme for turning a curriculum-based timetabling problem into a school timetabling problem. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT 2008)*, Montreal. http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Nurmi-TC1.pdf. Last accessed 12 February 2010.

Ohtsubo, M., Kurashige, K., & Kameyama, Y. (2006). Approach to the timetabling problems for junior high schools. *Journal of Japan Industrial Management Association*, *57*(3), 231–242.

Ostler, J., & Wilke, P. (2010). The Erlangen advanced timetabling system (EATTS) unified XML file format for the specification of timetabling systems. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)* (pp. 447–464).

Papoutsis, K., Valouxis, C., & Housos, E. (2003). A column generation approach for the timetabling problem of Greek high schools. *Journal of Operational Research Society*, *54*(3), 230–238.

Pillay, N., & Banzhaf, W. (2009). A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. *European Journal of Operational Research (EJOR)*, *197*, 482–491.

Pillay, N. (2010). A study into hyper-heuristics for the school timetabling problem. In *SAICSIT 2010: fountains of computing research* (pp. 258–264). ACM for Computing Machinery.

Post, G. F., & Ruizenaar, H. W. A. (2004). *Clusterschemes in dutch secondary schools* (Memorandum No. 1707). University of Twente, the Netherlands.

Post, G., Ahmadi, S., Daskalaki, S., Kingston, J. H., Kyngas, J., Nurmi, C., Ranson, D., & Ruizenaar, H. (2008). An XML format for benchmarks in high school timetabling. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT 2008)*, Montreal. http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Post-WD2a.pdf. Last accessed 12 February 2010.

Post, G., Ahmadi, S., & Geertsema, F. (2010a). Cyclic transfers in school timetabling. *OR Spectrum*. doi:10.1007/s00291-010-0227-y.

Post, G., Kingston, J. H., Ahmadi, S., Daskalaki, S., Gogos, C., Kyngas, G., Nurmi, C., Santos, H., Rorije, B., & Schaerf, A. (2010b). An XML format for benchmarks in high school timetabling. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)* (pp. 347–352).

Raghavjee, R., & Pillay, N. (2008). An application of genetic algorithms to the school timetabling problem. In C. Cilliers, L. Barnard & R. A. Botha (Eds.), *Proceedings of SAICSIT 2008* (pp. 193–199). New York: ACM.

Raghavjee, R., & Pillay, N. (2009). Evolving solutions to the school timetabling problem. In *Proceedings of the world congress on nature and biologically inspired computing (NaBIC'09)* (pp. 1524–1527). New York: IEEE Press.

Raghavjee, R., & Pillay, N. (2010a). An informed genetic algorithm for the high school timetabling problem. In *Proceedings of SAICSIT 2010: fountains of computing research* (pp. 408–412). New York: ACM.

Raghavjee, R., & Pillay, N. (2010b). Using genetic algorithms to solve the South African school timetabling problem. In *Proceedings of the world congress on nature and biologically inspired computing (NaBIC'10)*. New York: IEEE Press.

Reis, L. P., & Oliveira, E. (2000). A language for specifying complete timetabling problems. In *PATAT'00: selected papers from the 3rd international conference on the practice and theory of automated timetabling* (pp. 322–341).

Ribic, S., & Konjicija, S. (2010). A two phase integer programming approach to solving the school timetabling problem. In *Proceedings of the international conference on information technology interfaces (ITI)* (No. 5546473, pp. 651–656).

Santos, H. G., Ochi, L. S., & Souza, M. J. F. (2004). An efficient tabu search heuristic for the school timetabling problem. In *Lecture notes in computer science* (Vol. 3059, pp. 468–481).

Santos, H. G., Ochi, L. S., & Souza, M. J. F. (2005). A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. *Journal of Experimental Algorithms*, *10*, 2.9. doi:10.1145/1064546.1180621.

Santos, H. G., Uchoa, E., Ochi, L. S., & Maculan, N. (2008). Strong bounds with cut and column generation for class-teacher timetabling. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT 2008)*, Montreal. http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Santos-WD2b.pdf. Last accessed 12 February 2010.

Schaerf, A. (1996). *Tabu search techniques for large high-school timetabling problems* (Technical Report CS-R9611 1996). Computer Science/Department of Interactive Systems, Centrum Voor Wiskunder en Informatica (CWI). ISSN 0169-118X.

Schaerf, A. (1999a). A survey of automated timetabling. *Artificial Intelligence Review*, *13*(2), 87–127.

Schaerf, A. (1999b). Local search methods for high school timetabling problems. *IEEE Transactions on Systems, Man and Cybernetics*, *29*(4), 377–386.

Slechta, P. (2005). Decomposition and parallelization of multi-resource timetabling problem. In E. Burke & M. Trick (Eds.), *Lecture notes in computer science* (Vol. 3616, pp. 177–189).

Smith, K. A., Abramson, D., & Duke, D. (2003). Hopfield neural networks for timetabling: formulations, methods, and comparative results. *Computers and Industrial Engineering*, *44*, 283–305.

Srndic, N., Dervisevic, M., Pandzo, E., & Konjicija, S. (2009). The application of a parallel genetic algorithm to timetabling of elementary school classes: a coarse grained approach. In *Proceedings of ICAT 2009–2009 22nd international symposium on information, communication and automation technologies* (pp. 1–5). New York: IEEE Press.

Souza, M. J. F. (2004). A GRASP-tabu search algorithm for solving school timetabling problems. *Metaheuristics: Computer Decision-Making*, 659–672.

Valouxis, C., & Housos, E. (2003). Constraint programming approach for school timetabling. *Computers and Operations Research*, *30*, 1555–1572.

Wilke, P., Grobner, M., & Oster, N. (2002). A hybrid genetic algorithm for school timetabling. In *Lecture notes in computer science: Vol. 2557/2002. AI 2002: advances in artificial intelligence* (pp. 455–464). Berlin: Springer.

Wilke, P., & Ostler, J. (2008). Solving the school timetabling problem using tabu search, simulated annealing, genetic and branch & bound algorithms. In *Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT 2008)*, Montreal. http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Wilke-WD2c.pdf. Last accessed 12 February 2010.

Wilke, P., & Killer, H. (2010a). A comparison of algorithms for solving school and course timetabling problems using the Erlangen advanced timetabling system. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)* (pp. 427–439).

Wilke, P., & Killer, H. (2010b). Walk down jump up algorithm—a new hybrid algorithm for timetabling problems. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)* (pp. 440–446).

Willemen, R. J. (2002). *School timetabling construction: algorithms and complexity*. Phd Thesis, Technische Universiteit Eindhoven, The Netherlands.

Wood, J., & Whitaker, D. (1998). Student central school timetabling. *Journal of the Operational Research Society*, *49*(11), 1146–1152.

Wright, M. (1996). School timetabling using heuristic search. *Journal of the Operational Research Society*, *47*, 347–357.

Yigit, T. (2007). Constraint-based school timetabling using hybrid genetic algorithms. In *Lecture notes in computer science* (Vol. 4733, pp. 848–855).

Yongkai, L., Defu, Z., & Leung, S. C. H. (2009). A simulated annealing algorithm with a new neighborhood structure for the timetabling problem. In *Proceedings of the 2009 world summit on genetic and evolutionary computation (GEC'09)* (pp. 381–386).

Yoshikawa, M., Kaneko, K., Nomura, Y., & Watanabe, M. (1994). A constraint-based approach to high-school timetabling problems: a case study. In *AAAI'94: proceedings of the twelfth national conference on artificial intelligence* (pp. 1111–1116). Menlo Park: Am. Assoc. of Artificial Intelligence.

Yoshikawa, M., Kaneko, K., Yamanouchi, T., & Watanabe, M. (1996). A constraint-based high school scheduling system. In *IEEE expert: intelligent systems and their applications* (pp. 63–72). IEEE Educational Activities Department.

Zuters, J. (2007). Neural networks to enrich fitness function in a GA-based school timetabling model. In *Proceedings of WSEAS transactions on information science and application* (Vol. 4(2), pp. 346–353).