School of Computer Science and Information Technology
University of Nottingham
Jubilee Campus
NOTTINGHAM NG8 1BB, UK

Computer Science Technical Report No. NOTTCS-TR-2006-4

A Survey of Search Methodologies and Automated Approaches for Examination Timetabling

Rong Qu, Edmund Burke, Barry McCollum
Liam T.G. Merlot and Sau Y. Lee

First released: July, 2006

© Copyright 2006 Rong Qu, Edmund Burke, Barry McCollum, Liam T.G. Merlot and Sau Y. Lee

In an attempt to ensure good-quality printouts of our technical reports, from the supplied PDF files, we process to PDF using Acrobat Distiller. We encourage our authors to use outline fonts coupled with embedding of the used subset of all fonts (in either Truetype or Type 1 formats) except for the standard Acrobat typeface families of Times, Helvetica (Arial), Courier and Symbol. In the case of papers prepared using TEX or LATEX we endeavour to use subsetted Type 1 fonts, supplied by Y&Y Inc., for the Computer Modern, Lucida Bright and Mathtime families, rather than the public-domain Computer Modern bitmapped fonts. Note that the Y&Y font subsets are embedded under a site license issued by Y&Y Inc. For further details of site licensing and purchase of these fonts visit http://www.yandy.com

A Survey of Search Methodologies and Automated Approaches for Examination Timetabling

R. Qu[1], E. K. Burke[1], B. McCollum[2], L.T.G. Merlot[3], and S.Y. Lee[4]

[1] Automated Scheduling, Optimisation and Planning (ASAP) Group
School of CSiT, University of Nottingham, Nottingham, NG8 1BB, U.K.
rxq@cs.nott.ac.uk, ekb@cs.nott.ac.uk

[2] School of Computer Science, Queens University, Belfast
University Road, N. Ireland, BT7 1NN
b.mccollum@qub.ac.uk

[3] Quantm Ltd, 4/333 Flinders Lane, Melbourne, Australia, 3000
liam.merlot@quantm.net

[4] Mechanical and Industrial Engineering, University of Toronto
5 King's College Road, Toronto, ON, Canada M5S 3G8
lee@mie.utoronto.ca

Abstract. Exam timetabling is one of the most important administrative activities that takes place in academic institutions. In this paper we present a critical discussion of the research on exam timetabling in the last decade or so. This last ten years has seen an increased level of attention on this important topic. There has been a range of significant contributions to the scientific literature both in terms of theoretical and practical aspects. The main aim of this survey is to highlight the new trends and key research achievements that have been carried out in the last decade. We also aim to outline a range of relevant important research issues and challenges that have been generated by this body of work.

We first define the problem and review previous survey papers. Algorithmic approaches are then classified and discussed. These include early techniques (e.g. graph heuristics) and state-of-the-art approaches including meta-heuristics, constraint based methods, multi-criteria techniques, hybridisations, and recent new trends concerning neighbourhood structures, which are motivated by raising the generality of the approaches. Summarising tables are presented to provide an overall view of these techniques. We discuss some issues on decomposition techniques, system tools and languages, models and complexity. We also present and discuss some important issues which have come to light concerning the public benchmark exam timetabling data. Different versions of problem datasets with the same name have been circulating in the scientific community in the last ten years which has generated a significant amount of confusion. We clarify the situation and present a re-naming of the widely studied datasets to avoid future confusion. We also highlight which research papers have dealt with which dataset. Finally, we draw upon our discussion of the literature to present a (non-exhaustive) range of potential future research directions and open issues in exam timetabling research.

1 Introduction

Timetabling problems arise in various forms of real-world problem solving circumstances including educational timetabling (e.g. [31]), nurse scheduling (e.g. [19]), sports timetabling (e.g. [76]) and transportation timetabling (e.g. [94]). They have represented a challenging and important problem area for researchers across both Operational Research and Artificial Intelligence since the 1960s. Recent years have seen an increased level of research activity in this area. This is evidenced (among other things) by the emergence of a series of international conferences on the Practice and Theory on Automated Timetabling (PATAT) ([17, 18, 25, 43, 44]), and the establishment of a EURO (European Association of Operational Research Societies) working group on automated timetabling (see http://www.asap.cs.nott.ac.uk/watt/index.html).

Burke, Kingston and de Werra [31] (2004) gave a definition of general timetabling, which covers many cases:

> A timetabling problem is a problem with four parameters: T, a finite set of times; R, a finite set of resources; M, a finite set of meetings; and C, a finite set of constraints. The problem is to assign times and resources to the meetings so as to satisfy the constraints as far as possible.

Among the wide variety of timetabling problems, educational timetabling is one of the mostly studied from a practical viewpoint. It is one of the most important and time-consuming tasks which occur periodically (i.e. annually, quarterly, etc) in all academic institutions. The quality of the timetabling has a great impact on a broad range of different parties including lecturers, students and administrators (see [123, 129]). Variants of educational timetabling include school timetabling (class-teacher scheduling), course and exam timetabling (e.g. [31, 131]), faculty timetabling and classroom assignment (e.g. [9]). It has been observed that course and exam timetabling are relatively close problems [131] but very significant differences do exist [100]. This survey will concentrate on exam timetabling.

An excellent survey of examination timetabling was published in 1986 [47] and an insightful *follow up* paper appeared in 1996 [49]. However, a significant number of research papers in the area have been published since 1996. This paper will concentrate upon the research that has appeared since the publication of [49]. The last decade has seen the establishment of a collection of benchmark exam timetabling problems [51] which have been used by many of the examination timetabling research that have appeared since 1996. Moreover, there has been some confusion in the literature caused by the existence of different benchmark problem datasets with the same names. This paper aims to eradicate such confusion by presenting a definitive re-naming of the sets and by clarifying the situation over which papers dealt with which problems.

1.1 Exam Timetabling Problems

Exam timetabling problems can be defined as assigning a set of exams $E = e_1$, e_2, ... e_e into a limited number of ordered timeslots (time periods) $T = t_1$, t_2, ... t_t, subject to a set of constraints. The complexities and challenge of timetabling problems arise from the fact that a large variety of constraints, some of which contradict each other, need to be satisfied in different institutions ([23, 50]). In timetabling literature, constraints are usually categorised into two types: hard constraints and soft constraints, which are explained below:

– *Hard Constraints* cannot be violated in any circumstances (mainly due to physical restrictions). For example, *conflicting* exams (i.e. those which involve common resources such as students) cannot be scheduled simultaneously. Another example can be seen by noting that the number of students taking an exam cannot exceed the seating capacity of the room to which the exam is assigned. A timetable which satisfies all of the hard constraints is usually termed *feasible*.
– *Soft Constraints* are desirable but are not absolutely critical. In practice it is usually impossible to find feasible solutions that satisfy all of the soft constraints. Soft constraints usually vary (and sometimes conflict with each other) from one institution to another in terms of both the types and their importance ([23]). The most common soft constraint in the exam timetabling literature is to spread conflicting exams as much as possible throughout the examination session so that students can have enough revision time between exams. An example of another soft constraint which may conflict with this is represented by the goal of scheduling all the large exams as early as possible to allow enough time for marking. The quality of timetables is usually measured by checking to what extend the soft constraints are violated in the solutions generated.

Due to the large variety of problems presented and investigated, it would be neither practical nor beneficial to present a comprehensive list of all the hard and soft constraints that occur in timetabling research. We list some of the key hard and soft constraints for exam timetabling in Table 1 and Table 2, respectively. We believe that these cover most of the constraints appeared in the literature. It can be observed that they can be roughly grouped as time related (No. 1. in Table 1 and Nos 1.-7. in Table 2) or resource related (No. 2. in Table 1 and No. 8.-11. in Table 2). Most of the survey papers reviewed in Section 1.2 present lists of constraints in exam and general timetabling. It appears that the hard constraints listed in Table 1 and the first soft constraint in Table 2 have been most covered by the research in the literature.

We will begin this critical review of the research area by overviewing a number of surveys that have appeared in the literature since the 1960s. Many of these papers cover educational timetabling in general and thus include discussions of examination timetabling in addition to other discussions.

Table 1. Primary Hard Constraints in Exam Timetabling Problems

	Primary Hard Constraints
1.	no exams with common resources (e.g. students) assigned simultaneously
2.	resources of exams need to be sufficient (i.e. size of exams need to be below the room capacity, enough rooms for all of the exams)

Table 2. Primary Soft Constraints in Exam Timetabling Problems

	Primary Soft Constraints
1.	spread conflicting exams as even as possible, or not in x consecutive timeslots or days
2.	groups of exams required to take place at the same time, on the same day or at one location
3.	exams to be consecutive
4.	schedule all exams, or largest exams, as early as possible
5.	ordering (precedence) of exams need to be satisfied
6.	limited number of students and/or exams in any timeslot
7.	time requirements (e.g. exams (not) to be in certain timeslots)
8.	conflicting exams on the same day to be located nearby
9.	exams may be split over similar locations
10.	only exams of the same length can be combined into the same room
11.	resource requirements (e.g. room facility)

1.2 Previous Surveys on Educational Timetabling

A number of previous surveys on timetabling and related issues have appeared in the last four decades. In addition, an online bibliography was prepared by **Kingston** [92] in 1995 and includes more than 1000 references.

An early survey by **Miles** [105] in 1975 provides a useful bibliography of early developments on computer aided timetabling. Another well-known early survey by **Schmidt and Strohlein** [133] in 1979, including more than 200 references, covers almost all the work on timetabling before 1979. **de Werra** in 1985 [65] introduced various mathematical (graph theoretical) models and briefly overviewed methods for class-teacher and course timetabling based on graph colouring and network flows methods. The author noted that exam timetabling and course scheduling were similar to each other although there were differences between them. In 1997 [66] the author introduced some requirements in timetabling into the restricted graph coloring models and reviewed some mathematical programming formulations.

Carter in 1986 [47] presented a review of the early research on practical applications of examination timetabling in several universities. He reviewed a variety of graph heuristics and pointed out that none of the algorithms/packages had been implemented in more than one institution. There was no standard data on which comparisons could be carried out. Also, measures of a problem's difficulty did not exist. In 1996, **Carter and Laporte** [49] updated the above survey to summarise the algorithmic approaches from 1986 to 1996. The criteria for discussion was that the method should be either tested on real data or implemented in real world applications. They categorised the methods into four types: cluster methods, sequential methods, generalised search (meta-heuristics) and constraint based techniques. They observed that the approaches implemented in practice were relatively simple variants of different methods and only addressed a subset of the constraints in the problems. The authors concluded by suggesting that timetabling researchers should report test results on benchmark problems to gain a better understanding of various approaches taken in exam timetabling. As we will see later in this paper, this is what has happened since 1996.

Burke et al constructed a questionnaire in 1996 on exam timetabling [23] and sent it to 95 British universities, of which 56 replied. The issues concerned included:

– The structure of the problems (i.e. size, complexity and constraints, etc),
– How the problems were solved, and
– The objective of the timetabling problem (i.e. what constitutes good solutions).

The resultant data was analysed to provide information on the constraints involved between exams, students, departments, timeslots and rooms. In addition to the 13 constraints originally listed in the questionnaire, another 19 constraints were provided by the universities, demonstrating that in reality there is a wide variety of constraints among different institutions. It was found that just 21% of the universities used some form of computational help. Where timetables were constructed manually, half of the institutions did not base their solution on the previous year's timetable, requiring a workload of many months. The paper suggested some appropriate properties of automated timetabling systems that could be utilised in practice. This paper provided some insight into the pertinent issues that impacted upon real world exam timetabling issues at the time. In 1997, **Burke et al** [27] presented a brief introduction to automated university exam timetabling research. The paper concentrated on techniques on university timetabling which were popular at the time.

Bardadym in 1996 [9] considered different issues in computer-aided management systems for timetabling. He discussed problems, requirements, data representations and mathematical models. Solution methods from the 1960s to the 1990s were also overviewed mainly on heuristics, meta-heuristics and algorithmic tools for integration in decision support systems. Meta-heuristics and interactive timetabling were seen as the new wave of computer-aided timetabling systems. He also discussed open issues for future timetabling research.

Wren [148] in 1996 illustrated a useful and interesting link between scheduling, timetabling and rostering by studying an example of the Traveling Salesman Problem. He concluded that the similarity between timetabling and staff rostering may lead to successful cross-fertilisation on different types of problems. Indeed, recent research (as we shall see later) has provided some evidence to support this.

Schaerf in his 1999 survey [131] looked at the formulations of school, course and exam timetabling and declared that it is difficult to make a distinction between the later two. Based on the definitions of variants of these problems, solution techniques particularly from artificial intelligence were classified and reviewed. Possible future directions were presented including specific techniques, standardization, approximability, the design of a powerful constraint language, and the combination and comparisons on different techniques.

Burke and Petrovic in 2002 [41] gave an overall review of recent research conducted on university (course and exam) timetabling that had been carried out in their group including hybrid evolutionary algorithms, multi-criteria approaches and case-based reasoning techniques. An outline of research on sequential, clustering, constraint based techniques and meta-heuristic methods was also provided. Future directions highlighted knowledge based systems and approaches which aim to raise the generality of timetabling system. A survey by **Petrovic and Burke** in 2004 [115] discussed a wider range of themes including the integration of meta-heuristics and multi-criteria approaches, case-based reasoning and adaptive approaches with methods aiming at a higher level of generality.

An article by **Burke, Kingston and de Werra** [31] (2004) discussed the application of graph coloring methods to timetabling. The authors consider class-teacher, course, exam and sports timetabling. This paper highlights the role that graph coloring methods have played in the timetabling literature over the last 40 years or so.

From the above brief discussion, we can see that there are a number of excellent surveys in the literature concerning different issues that have impacted upon exam timetabling research. We also provide in Appendix A a list of PhD theses that have appeared during the years on both course and exam timetabling, where extensive reviews have been carried out upon specific aspects of the educational timetabling. However there is no comprehensive review on exam timetabling in the last decade. This body of work includes a number of state-of-the-art approaches and has introduced a wide variety of diverse and successful methodologies. This paper aims to build on **Carter and Laporte**'s 1996 survey [49] to provide a modern discussion of the methods and techniques that have been developed for this important problem. With this in mind, we will not discuss in detail the work that appeared before 1996. We aim to keep our bibliography of examination timetabling papers and our classification tables (see later on) up to date on the following web page http://www.cs.nott.ac.uk/∼rxq/bibliography.htm. We would be very grateful if authors could contact us as new papers appear in order to regularly update this public resource. Although we have covered all the relevant papers of which we are aware, we may have inadvertently omitted relevant papers that have already appeared. If so, we apologise and would welcome the opportunity to add them to the information on our web site.

2 Exam Timetabling Approaches/Techniques

There has been a significant amount of exam timetabling research in the last ten years. In this section we classify the wide variety of research techniques that have appeared in this time. We note that many of the successful methodologies that have appeared in the literature represent hybridisations of a number of techniques. Thus the classification is not strict. Indeed, several of the methodologies could have appeared in two or more of the classifications. Where possible, we have classified by the main technique used. For each technique reviewed, a corresponding summarising table is presented in Appendix B.

2.1 Graph Based Sequential Techniques

Timetabling problems, without soft constraints, can be modelled as graph coloring problems (see [47, 79]). The paper by **Welsh and Powell** [142] in 1967 built the bridge between graph coloring and timetabling, which led to a significant amount of later research on graph heuristics in timetabling (e.g. [101, 102]). In exam timetabling problems, the exams can be represented by vertices in a graph, and the hard constraint between exams are represented by the edges between the vertices. The graph coloring problem of assigning colors to vertices, so that no adjacent vertices have the same color, then corresponds to the problem of assigning timeslots to exams. Different soft constraints (such as those listed in Table 2) need to be considered separately and the degree to which timetabling solution satisfy represents a measure of solution quality.

The basic graph coloring based timetabling heuristics are constructive methods that order the exams and assign them one by one by how difficult they are to be scheduled into the timeslots. There is a range of ordering strategies and their modified variants that appear in the timetabling literature [47]. We list, in Table 3, some of the widely employed ordering strategies. A random ordering method has also been employed in the literature to introduce randomness in hybrid approaches and provide comparisons.

Table 3. Widely Studied Ordering Strategies in Graph Heuristics in Exam Timetabling Problems

Heuristics	**Ordering Strategy**
Saturation Degree [12]	increasingly by the number of timeslots available for the exam in the timetable at the time
Largest Degree [13]	decreasingly by the number of conflicts the exams have with the other exams
Largest Weighted Degree [51]	the same as Largest Degree but weighted by the number of students involved
Largest Enrolment [147]	decreasingly by the number of enrolments for the exam
Random Ordering	randomly order the exams
Color Degree [51]	decreasingly by the number of conflicts the exam has with those scheduled at the time

Graph based heuristics as simple constructive methods played a very important role in the early days of timetabling research [47]. Although originally presented as techniques (albeit simple ones) in their own right, they are still being employed and adapted within the current research literature. Their great strength is that they can provide reasonably good results within a small computational time and are very easy to implement. They are often used to construct initial solutions, or to build good portions of solutions before improvement techniques are applied (see more details in the following sections). A recent article by **Burke et al** [31] overviewed graph coloring techniques.

Carter, Laporte and Lee [51] in 1996 studied the first five ordering strategies in Table 3 on real and randomly generated exam timetabling problems. Largest cliques, which are the largest sub-graphs where each of the vertices is adjacent to all of the others, were used to build initial solutions, based on which graph heuristics and backtracking techniques were employed to construct the solutions. The idea is that the cardinality of the largest clique determines the least number of timeslots required for the problem. The results indicated that none of the heuristics outperformed any of the rest over all of the problems tested. Another important contribution of this work is the introduction of a set of 13 exam timetabling problems, which became standard benchmarks in the field. They have been widely studied and used by different approaches during the years (see Table 6). We call this the University of Toronto data and discuss it further in Section 3.1. In 2001, **Carter and Johnson** [48] investigated the sub-graphs which are sufficiently dense (*almost cliques*) on 11 of the instances in the above data. They observed that in real exam timetabling problems there are usually many of the largest cliques and showed that employing the *almost cliques* can potentially extend and improve the above approach.

Burke, Newall and Weare [40] in 1998 studied the effect of introducing a random element into the employment of graph heuristics (Saturation Degree, Color Degree and Largest Degree in Table 3) by developing two variants of selection strategies: (1) tournament selection that randomly chooses one from a subset of the first exams in the ordered list; and (2) bias selection that takes the first exam from an ordered list of a subset of all of the exams. These simple techniques, when tested on three of the Toronto datasets, improved the pure graph heuristics with backtracking in terms of both the quality and diversity of the solutions.

Burke and Newall in 2004 [37] investigated a dynamic ordering strategy which ordered the exams adaptively during the problem solving in an iterative process. A heuristic modifier was designed to update the ordering of the exams according to the experience obtained with regard to the difficulty of assigning them in the previous process. It was observed that a fixed pre-defined heuristic (employed as a measure of difficulty) in a traditional sequential strategy (as shown in Table 3) does not always perform well over the full range of problems. The method described in this paper adapts the heuristic ordering itself *on the fly* during the problem solving for the particular problem being solved. Extensive experiments were carried out on 11 of the Toronto datasets, and another benchmark (which we call the Nottingham data, see Section 3.2). This approach was shown to be simple and effective (comparable or occasionally better than state-of-the-art approaches [51, 71]) and not dependent on the initial ordering of exams.

Fuzzy logic was employed by **Asmuni et al** [8] in 2004 to order the exams to be scheduled based on graph coloring heuristics on the Toronto datasets. The idea is that when ordering the exams by how difficult they are, fuzzy functions can be used to give an appropriate evaluation. It was seen that different fuzzy functions need to be used on different problems to obtain the best results.

Corr et al [59] developed an neural network from which a measure of the difficulty of assigning exams during the timetable construction can be obtained by recursively inputting the updated solution construction states. The objective is to adaptively assign the most difficult exams at the early stage of solution construction. The neural network was trained by storing the states of timetable construction (feature vectors) using three graph heuristics. The work has demonstrated the feasibility of employing neural network based methods as an adaptive and generally applicable technique on timetabling problems.

Due to the limitations of constructive methods, where early assignments may lead to situations that no feasible timeslots are available for exams left later on in the construction process, *backtracking* is usually employed (e.g. [51]) that unassigns the early conflicting exams to allocate the current ones. A *look ahead* technique was also studied in [35] to solve this problem (for more details see memetic algorithms in Section 2.4).

As mentioned earlier, techniques which hybridise graph heuristics with other methods are still appearing in the most modern exam timetabling research literature. The employment of graph heuristics within Hyper-heuristics is discussed in Section 2.6.

2.2 Constraint Based Techniques

Constraint logic programming [86] and constraint satisfaction techniques [11] have their origins in Artificial Intelligence research. Such methods have attracted the attention of researchers in timetabling due to the ease and flexibility with which they can be employed for timetabling problems. Exams are modelled as variables with finite domains. Values within the domains representing the timeslots and rooms for the variables are assigned sequentially to construct the solutions for the problems. Early research focused on finding feasible solutions (i.e. satisfying all hard constraints). **Brailsford, Potts and Smith** [11] in 1999 introduced various searching methods on constraint satisfaction problems and demonstrated that this technique can be applied to optimisation problems.

Constraint based techniques are usually computationally expensive due to the fact that the number of possible assignments increases exponentially with the number of variables. They, on their own, cannot usually provide high quality solutions compared with the state-of-the-art approaches [11] on complex optimisation problems. Backtracking is employed when no values can be assigned to variables later in the process. Different heuristics and techniques are usually integrated with such methods to reduce the time complexity for solving practical problems (see [49, 143]). For example, the *labelling strategy* indicates the order in which the variables are to be initiated and is usually where heuristics are introduced.

David [62] (1998) applied constraint satisfaction techniques to model an exam timetabling problem in a French school, the Ecole des Mines de Nantes. Time complexity was crucial thus partial solutions were firstly obtained, based on which particular local repairing strategies were employed successively to obtain complete solutions and make improvements. The approach was run several times with different initial assignments to reduce the chance of missing good solutions. It was employed successfully in the school and can usually generate solutions within one second.

Reis and Oliveira [120] (1999) developed an examination timetabling system based on ECLiPSe [7], which is a Prolog based system that serves as the platform for developing various extensions in constraint logic programming. A set of hard and soft constraints in the problem were built into a constraint satisfaction model, where set variables were employed and handled by the libraries in ECLiPSe. Its application on random and a large real exam timetabling problem data in the University of Fernando Pessoa in Porto demonstrated the efficiency of the model.

Merlot et al (2003) [104] employed constraint programming in a similar way to that of [10] using OPL [87], an optimisation programming language, to produce initial solutions. Then a Simulated Annealing and a hill climbing method (see Section 2.3 below) were used to improve the solutions. Variables (exams) were ordered by the sizes of their domains (available timeslots) and scheduled into the earliest timeslots one by one. The pure constraint programming obtained the best result for one of the Toronto datasets. The overall hybrid approach was tested on problems at the University of Melbourne, two variants of Toronto instances and the Nottingham data (see Section 3). This approach obtained the best results reported in the literature on several instances of the Toronto and Nottingham datasets at the time.

Duong and Lam [64] (2004) also employed constraint programming to generate initial solutions for a Simulated Annealing methodology for the exam timetabling problems at HoChiMinh City University of Technology. Backtracking and forward checking were employed to reduce the searching effort. The labelling strategy dynamically ordered the variables (exams) by a number of factors such as the size of the domain and the number of students.

Recent research on constraint based techniques represents overall methodologies which are hybridised with other techniques. The labelling strategy is usually integrated with different problem specific heuristics for variable ordering and is crucial to the success of the method. The development of some powerful constraint programming systems/languages (e.g. ECLiPSE [7], CHIP [135], OPL [87], Prolog [54]) significantly supported the construction of the complete exam timetabling systems in real world applications. However, only particular problems at different institutions have been tackled with this approach in the literature. No comparisons have been made between constraint based techniques and other state-of-the-art approaches (besides with a manual method) on the same problems, except **Merlot et al** [104] on the Toronto and Nottingham data. It is worth noting though that this method can produce the best results in the literature on some benchmark problems.

2.3 Local Search Based Techniques

Local search based techniques [117] (e.g. Tabu Search, Simulated Annealing and their variants) and Evolutionary Algorithms (see Section 2.4) are usually seen as belonging to meta-heuristics [28, 82]. Local search methods are a family of general techniques which solve problems by searching from an incumbent solution to its *neighbourhood*. Different neighbourhood structures and moving operators within the search space distinguish different local search techniques. The search is guided by a defined objective function, which is used to evaluate the quality of the generated timetables.

These techniques represent a large body of work in the last decade [49] and have been applied on a variety of timetabling problems, mainly because different constraints can be handled relatively easy. The performance and efficiency of these techniques are highly dependent upon the parameters and search space properties (e.g. connectivity, ruggness), thus a lot of domain knowledge is usually built-in to deal with specific problems. A large amount of variants and combinations have been investigated. We will first deal with Tabu Search.

2.3.1 Tabu Search

Tabu Search [80, 81] explores the search space by not re-visiting a list of recent moves (kept in a *tabu list*). They may, however, be selected if they generated the best solution obtained so far by using an aspiration strategy. Otherwise the search moves to other neighbourhoods even if the resulting solutions are worse than the incumbent solutions, which is able to escape from local optima. Parameters need to be fine-tuned in designing the approach and this is very much dependant on the problem in hand. Such parameters include the tabu list and the stopping criteria among others.

Di Gaspero and Schearf [71] (2001) carried out a valuable investigation on a family of Tabu Search based techniques whose neighbourhoods concerned those which contributed to the violations of hard or soft constraints. Exhaustive and biased selection strategies were also studied. The length of the tabu list is dynamic and the cost function is adaptively set during the search. This approach was tested on two sets of benchmark problems (Toronto and Nottingham) and was shown to perform similarly to graph heuristics, clique initialisation [51] and Memetic Algorithm [38], but worse than a multi-stage memetic approach [35]. The authors experimentally demonstrated that the adaptive cost function and the effective selection of neighbourhoods concerning the violations were key features of the approach. In 2002 **Di Gaspero** [70] improved the approach by employing multiple neighbourhoods based on a *token-ring* search which circularly employs *recolor* (change single exam) and *shake* (swapping groups of exams), followed by *kickers* (change sequence of single exams) to further improve the solutions obtained. The technique extended the idea of diversifying the search from local optima.

White and Xie [144] (2001) developed a four-stage Tabu Search called OTTABU, where the solutions were gradually improved by considering more constraints at each stage, for the exam timetabling problem at the University of Ottawa. Should the 1^{st} stage fail, feasible solutions can be generated by adding some extra timeslots in the 2^{nd} stage. Subsequently, the solutions were gradually improved by considering 2^{nd} and 3^{rd} order constraints in the problem at the 3^{rd} and 4^{th} stage, respectively. In addition to recency short term memory, a frequency long term memory was also used to record the frequency of the most active moves in the search history. The size of the long term memory was set by analysing the number of less important exams in the problem. In [145] (2004) this approach was extended where both of the tabu lists could be dynamically relaxed (emptied) after a certain length of search time with no improvement. This approach compared favourbly to those from [51] and [71] on the Toronto data. The authors experimentally showed that employing long term memory can significantly improve Tabu Search on real-world problems.

Paquete and Stutzle (2002) [113] developed a Tabu Search methodology for exam timetabling where ordered priorities were given for the constraints. The constraints were considered in two ways: (1) one constraint at a time from the highest priority, where ties were broken by considering the lower priority constraints; (2) all the constraints at a time, starting from the highest priority. The 2^{nd} strategy, whose results were comparable with those of [51], obtained better results while the 1^{st} strategy was more consistent. The length of the tabu list was adaptively set by considering the number of violations in the solutions. It was observed that the length of the tabu list needed to be increased with the size of the problems.

2.3.2 Simulated Annealing

Simulated Annealing [1, 2] is motivated by the natural annealing process [2]. The idea is to search a wider area of the search space at the beginning of the process by accepting worse moves with a higher probability, which is gradually decreased as the search continues. A *temperature* is used within a *cooling schedule* to control the probability of the acceptance of worse moves in the search. Many parameters need to be tuned in Simulated Annealing including the initial and final temperatures, and the cooling factor in the cooling schedule. These parameters affect the performance and success of this approach.

Thompson and Dowsland [140] (1998) carried out valuable work to develop a two-stage approach where feasible solutions from the 1^{st} stage were improved in the 2^{nd} stage by Simulated Annealing concerning soft constraints. As different objectives were dealt with in different stages in turn, the solutions from early stage may be poor and thus a backtracking technique was proposed. **Dowsland** [73] also observed that the way neighbourhood was defined, the importance and how difficult to achieve the objectives greatly affected how they were tackled in each stage. Based on their work in [139], the authors further investigated the Kempe chain neighbourhood, where chains of exams rather than individual exams were moved. This gave more flexibility to enable the movement of large difficult exams within the timetable. They concluded that the most important factors in Simulated Annealing were the cooling schedule and the way neighbourhoods were defined and sampled. The authors reported that the developed exam timetabling system has been used in Swansea University successfully since 1993.

Bullnheimer [14] (1998) discussed how a model for Quadratic Assignment Problems was adapted to formulate a small scale practical exam timetabling problem at the University of Magdeburg. The models enabled the university administrators to control how much the conflicting exams need to be spaced out. Simulated Annealing was employed where two sets of neighbourhood structures (moving the timeslots of exams and moving single exams) were studied. However, the details of the parameters in the algorithm were not given.

Merlot et al (2003) [104] employed a Simulated Annealing approach initialised by constraint programming techniques (see Section 2.2 on "Constraint Based Techniques") and followed by hill climbing to further improve the solution. A modified Kempe chain neighbourhood was employed. The best results at the time for several of the Toronto instances were achieved by this hybrid approach. Indeed, the method still has some of the best known results. The authors suggested that methods combining solution construction with local search will dominate the future of exam timetabling research.

Duong and Lam [64] (2004) employed Simulated Annealing on the initial solutions generated by constraint programming for the exam timetabling problem at HMCM University of Technology. A Kempe Chain neighbourhood was employed in the Simulated Annealing, whose cooling schedule was experimentally set using mechanisms and algorithms. The authors noted that when limited time is given, it is crucial to tune the components in Simulated Annealing to the specific problems to be solved.

Burke et al [16] (2004) studied a variant of Simulated Annealing, called the Great Deluge algorithm [75]. The search accepts worse moves as long as the decrease on the quality is below a certain *level*, which is originally set as the quality of the initial solution and gradually lowered by a decay factor. The decay factor and an estimate of desired quality represent the parameters in this approach. The authors noted that such parameters can be pre-defined by users, who are usually not experts on Simulated Annealing. The initial solutions, however, need to be feasible to calculate the decay factor so a Saturation Degree was run a number of times, from which the best solutions were employed as the starting points. This approach was superior to a Simulated Annealing developed by the authors. It was shown to be effective and generated some of the best results on the Toronto and Nottingham datasets when compared with other approaches ([51, 71]). Comprehensive experiments were also carried out to analyse the trade-off between the time and solution quality on problems of different size. The approach was further studied in [36] where it was initialised by the method presented in [37].

2.3.3 Other Local Search Based Techniques

Recently, along with the study of different ways of escaping from local optima in local search based techniques, some researchers turned to investigating the effect of designing different neighbourhoods and have obtained some success on timetabling problems. This demonstrated that not only the way of search, but also the structure of the neighbourhood had significant impact on the searching algorithms. For example, *Kempe chain* neighbourhood structures as mentioned above were investigated by a number of researchers in exam timetabling (see [52, 61, 104]). The idea is that chains of conflicting exams are swapped between timeslots. Reasons for why this neighbourhood structure worked well were analysed [140]. Other approaches concerned multiple neighbourhood structures [70]. Compared with standard moves on single exams, this brought more flexibility in the navigation of search spaces for different problems.

Abdullah et al [3] in 2006 developed a large neighbourhood search based on the methodology of improvement graph construction originally developed by **Ahuja and Orlin** [6] for different optimization problems. To generate large neighbourhoods, instead of just considering traditional pair-wise exchange based operators, a tree-based neighbourhood structure was designed to carry out cyclic exchanges among all of the timeslots. The approach has provided the best results on a number of Toronto dataset problems at the time of publication. However, a large amount of computational time was needed. The approach was further developed in [4] where the improvement moves were kept in a tabu list. Capacitated exam timetabling problems (Toronto *c* in section 3.2) were considered in this paper.

Another technique concerning different neighbourhoods is Variable Neighbourhood Search (e.g. [83, 106]). This approach systematically varies a number of neighbourhood structures. The aim is to escape from local optima by switching from the search space defined by one neighbourhood to another. However, not much work has been done in exam timetabling using this approach. **Burke et al** [22] investigated variants of Variable Neighbourhood Search and obtained the best results in the literature across some of the problems in the Toronto datasets. The results were further improved by using a standard Genetic Algorithms to intelligently select subset of neighbourhoods. The later approach has strong link to the work in hyper-heuristics and indicated promising directions on developing general approaches on neighborhoods rather than directly on solutions. In hyper-heuristics, Variable Neighbourhood Search was also employed where graph heuristics rather than neighbourhoods were searched. We present more details in Section 2.6 on "Hyper-heuristics".

In addition to designing different neighbourhood structures within local search based techniques, some researchers have also looked into how iterative techniques can help in solving complex problems. In Iterative Local Search [99], the search restarts after a certain criteria is met. The motivation is to explore more areas of search space within a short time. It was first applied on the graph coloring problem [114] in 2002.

Caramia, DellOlmo and Italiano [45] (2001) developed a fine-tuned local search method where a *greedy scheduler* assigned exams into the least possible number of timeslots and a *penalty decreaser* improved the timetable without increasing the number of timeslots. When no improvement can be made the number of timeslots were increased gradually by a *penalty trader*. The process was repeated employing a permutation technique to reassign the priorities of exams. A properly tuned *checkpointing* scheme was also used to release the memory of the search. This approach still hold the best results reported in the literature on several instances of the Toronto datasets.

Casey and Thompson [52] (2003) investigated a Greedy Randomised Adaptive Search Procedures (GRASP) approach [122], which is a relatively new technique in timetabling. In GRASP, a local search algorithm is started iteratively after local optima are reached based on the initial solutions generated by a greedy approach. In [52], the initial solution in each iteration was generated by a modified Saturation Degree, where one exam from the first n (experimentally set as 2-6) exams ordered was assigned into the timetable. Backtracking was employed in conjunction with a tabu list to forbid indefinite cycles. A limited form of Simulated Annealing with high starting temperature and fast cooling was used in the improvement phase. Kempe chain moves were employed on exams that contributed the cost. The approaches applied on the Toronto datasets reported competitive results on some of the instances at the time.

2.3.4 Summary on the Discussion of Local Search Based Techniques

During the last decade local search based techniques have been very heavily studied and have obtained a marked level of success on timetabling. All of the work discussed above was either tested on benchmark data or implemented in real applications. Different ways of accepting the moves (i.e. moving strategies, acceptance strategy and selection strategies) were studied so as to escape from local optima by defining a variety of approaches in meta-heuristics for exam timetabling. However one significant drawback of these approaches is the effort required to tune the parameters for specific problems to get high quality solutions.

2.4 Population Based Algorithms

2.4.1 Evolutionary Algorithms

Evolutionary algorithms represent a family of population based algorithms including Genetic Algorithms, Memetic Algorithms and Ant Algorithms. Genetic algorithms have been the most studied in terms of exam timetabling research. In particular, hybridisations of genetic algorithms with local search methods (sometimes called memetic algorithms) have led to some success in the field.

Genetic Algorithms simulate the evolutionary process in nature by manipulating and evolving populations of solutions within the search space (see [82, 119, 130]). Solutions are coded as *chromosomes* and are evolved by a reproduction process using crossover and mutation operators, with the aim of getting better and better solutions through a series of generations. A set of parameters and operators in Genetic Algorithms need to be defined and set properly, making the approach (usually) more complicated than that of local search based searching methods. The searching strategy in Genetic Algorithms is fundamentally different from the local search based approaches discussed above in the sense that several solutions are dealt with at once (a population of solutions) rather than just one solution being improved through a series of iterations.

Corne, Ross and Fang [58] in 1994 provided a brief survey on using Genetic Algorithms in general educational timetabling and addressed some issues and future prospects. One important contribution of the work concerns the use of direct representation in Genetic Algorithms, which was shown to be incapable of dealing with certain problem structures in some specially generated graph coloring problems. In 2003 **Ross, Hart and Corne** [127] updated the above review in 1994 on representations and algorithms used in Evolutionary Algorithms on various kinds of timetabling problems. They concluded that researchers were tackling their own problems in different institutions and there was still the need for more comparisons of approaches on a wide range of problems.

Ross, Corne and Terashima-Marin [125] (1996) showed that transition regions exist in solvable timetabling problems by experimenting upon specially generated graph coloring problems of different connectivity and homogeneity. The authors indicated that the study can assist the understanding of how different algorithms perform on complex timetabling problems. In 1998 **Ross, Hart and Corne** [126] provided further evidence for the weakness of the use of direct coding in Genetic Algorithms. They observed the failure of a number of (evolutionary and non-evolutionary) approaches on solving special classes of graph coloring problems. They suggested that Genetic Algorithms should search for algorithms rather than actual solutions. Indeed hyper-heuristics (where a set of low level heuristics is searched by a high level algorithm - see more details in Section 2.6) do exactly this.

Terashima-Marin, Ross and Valenzuela-Rendon [136] in 1999 designed a clique-based crossover operator on timetabling problems that was transferred into graph coloring problems. Different recombination strategies were tested in the reproduction processes to maintain the cliques in parents into offsprings. They pointed out the same problem with direct representation in Genetic Algorithms as discussed above with [126]. They suggested alternatives for future work. The same authors also studied the penalty function on both random and real timetabling problems employing Hardness Theory, which predicts where the hardest instances are within timetabling problems [137]. However, they observed that adding this measure is not helpful in guiding Genetic Algorithms towards promising areas of search space. Based on the above work, they investigated the non-direct coding in Genetic Algorithms [138], where solution construction strategies and heuristics rather than the actual solutions were coded (e.g. configurations of constraint satisfaction methods, ordering of nodes being assigned and heuristics dealing with constraints, etc). Promising results obtained by this approach on the Toronto datasets indicated the potential of non-direct representations in Genetic Algorithms.

Erben [77] (2001) developed a grouping Genetic Algorithm where appropriate encoding and fitness functions were studied. Genes were grouped for each color in graph coloring problems (which model the exam timetabling problems with only hard constraints). Specially designed crossover and mutation operators for the group encoding were employed. Although the results in terms of solution quality did not compete with the best, the approach requires less computational time than some of the methods in the literature.

Sheibani [134], in 2002, built a special mathematical model and developed a standard Genetic Algorithm for solving exam timetabling problems in training centers with the objective of maximising the intervals between the exams. An activity-on-arrow network was employed to estimate the closeness between exams, which was used in the fitness function in Genetic Algorithm.

Wong, Cote and Gely [146] (2002) discussed some issues concerning their implementing a Genetic Algorithm on solving an exam timetabling problem at Cole de Technologie Suprieure, which was modelled as a Constraint Satisfaction problem. Tournament selection was used to select parents and repairing strategies were incorporated with mutation to produce better candidates. In 2005, **Cote, Wong, and Sabourin** [61] investigated a bi-objective evolutionary algorithm with the objectives of minimising timetable length and spacing out conflicting exams. Two local search operators (instead of recombination operators) were employed to deal with hard and soft constraints. They were a classic Tabu Search and a simplified Variable Neighbourhood Descent with Kempe chain and single move neighbourhoods. A ranking procedure was based on Pareto strength to carry out evaluation of the individuals in the population. The approach obtained competitive results on a number of benchmark problems against some of the methods in the literature (e.g. [51, 38, 104]). The paper also provided a review on all the state-of-the-art approaches on the Toronto datasets at the time.

2.4.2 Memetic Algorithms

Memetic algorithms [108] are an extension of Genetic Algorithms whose basic idea is that individuals in a population are improved during their lifetime within a generation. This is often implemented by employing local search methods (in the form of hill climbing or repairing strategies, etc) on individual members of a population between generations. **Burke and Landa Silva** [32] discussed a number of issues concerning the design of Memetic Algorithms for scheduling and timetabling problems. Recent research ideas and future directions on this topic were presented.

The ability to explore by employing a population based method and exploit for local search enables such methods to deal effectively with large complex problems. However, there is usually a price to pay in terms of the computational time required. Also the right balance between *exploration* and *exploitation* needs to be established [32]. There exists a number of in-depth studies on Memetic Algorithms concerning structures of the search space and different ways of hybridisations over a range of combinatorial optimisation problems (e.g. see [111, 112]).

Burke, Newall and Weare [38] (1996) developed a Memetic Algorithm which employs light and heavy mutation operators to reassign single exams and sets of exams, respectively, with the aim of escaping from local optima. Neither of these mutations on their own provided substantial improvement on the solution quality. Hill climbing was used to improve the individuals and it was shown that this improved the quality of timetables although a larger amount of computational time was required. Another contribution of this paper was the introduction of more benchmark exam timetabling problems (named as Nottingham data and described in Section 3.2). These have been widely used by a number of researchers in later work (see Table 8 and 13). The same authors also investigated the effects of diversity in initial populations in Memetic Algorithms [39] (1998). To generate a good diversity in the initial population, randomness was introduced by using different selection strategies in graph heuristics (see [40]), and three diversity measures were also developed to study the trade-off between the quality and diversity. It was shown that the study of diversity in initialisation offered great potential benefits for Memetic Algorithms. In 1999, **Burke and Newall** [35] studied decomposing the exam timetabling problems by each time assigning a subset of n exams which are the most difficult ones measured by graph heuristics (i.e. Color Degree, Largest Degree, Saturation Degree - see Table 3). Backtracking and look-ahead techniques were employed to avoid the problem of early assignments creating infeasibility later on. The exams assigned in previous stages are fixed and the sub-problem at the current stage is solved using the Memetic Algorithm investigated (e.g. see [38, 107, 130]. The algorithm dramatically reduced the time complexity required and produced high quality solutions when dealing with larger timetabling problems. The decomposition technique was actually independent of the memetic timetabling algorithm which was used on each of the decomposed subsets.

2.4.3 Ant Algorithms

Ant Algorithms [68] belong to the family of population based techniques. They simulate the way ants search for the shortest route to food by laying pheromone on the way. The shortest trails generate stronger levels of pheromone over a period of time. In the algorithm, each ant is used to construct a solution and the information during the search is maintained as pheromone, which is used to help generating solutions in the next stage. In exam timetabling, Ant Algorithms represent relatively recently explored techniques. In this context, they have not been particularly widely studied. Relevant work does, however, exist on graph coloring problems [60], where the frequency of the colors assigned for the vertices in the solution construction were employed as the pheromone.

Naji Azimi [109] in 2004 implemented an Ant Colony System and compared it with Simulated Annealing, Tabu Search and a Genetic Algorithm under a unified framework for solving systematically designed exam timetabling problems. Initial solutions for the Ant Colony System were generated heuristically and improved by local search afterwards. The results analyzed over the running time indicated that the Ant Colony approach performed the best (although not on all of the problems) and Tabu Search had the highest level of improvement upon the initial solution randomly generated. Three variants of hybridisation on Tabu Search and the Ant Colony method were then studied in [110]. It was observed that the hybrid approaches work better than each single algorithm, and the sequential Ant Colony System followed by Tabu Search obtained the best results. However only randomly generated data was tested.

Dowsland and Thompson in 2005 [74] developed Ant Algorithms based on the graph coloring model studied in [60] for solving exam timetabling problems without soft constraints (i.e to find the lowest number of timeslots). Extensive experiments were carried out to measure the performance of the algorithm with different configurations. These include the initialisation methods (i.e. recursive Largest Degree and Saturation Degree), trail calculations, three variants of fitness functions and different parameter settings. The results obtained were competitive to the others on the same dataset. It was also observed that the initialisation methods had significant influence on the solution quality. Extensions of the algorithm to incorporate other constraints (i.e. time windows, seating capacities and second-order conflicts) were also discussed.

2.4.4 Summary of the Discussion of Population-Based Techniques

Evolutionary methods (particularly evolutionary hybrids) have been very effective in providing high quality solutions to exam timetabling problem. Recent research on Evolutionary Algorithms discussed has the issues of encoding to deal with the problem structures which direct coding is not capable of dealing with. This opened up a new research direction in Evolutionary Algorithms and has led to some of the initial work in Hyper-heuristics (see Section 2.6). Multi-criteria techniques also form an important research direction in the area of Evolutionary Algorithms for exam timetabling problems. More details are discussed in Section 2.5 on "Multi-criteria Techniques".

Ant Algorithms have been applied in exam timetabling with some initial observations (see [74, 109]). As relatively new techniques, they represent some potential and should attract more attention in the exam timetabling domain.

2.5 Multi-Criteria Techniques

In the majority of algorithms/approaches on timetabling, weighted costs of violations of different constraints are summed and used to indicate the quality of the solutions. However, in real world circumstances the constraints are often considered from different points of view by different parties involved in the timetabling process [49]. The simple sum of costs on different constraints cannot always take care of the situation in such cases. Multi-criteria techniques have been studied recently in timetabling with the aim of handling different constraints easily by considering a vector of constraints instead of a single weighted sum. In multi-criteria techniques, each criterion can be considered to correspond to a constraint, which has a certain level of importance and is dealt with individually. In some approaches, multiple stages have been employed to deal with different objectives. **Landa, Burke and Petrovic** [95] provide a review of a large number of scheduling and timetabling applications which employ multi-criteria techniques.

Burke, Bykov and Petrovic [15] (2001) developed a two-stage multi-criteria approach dealing with nine criteria in exam timetabling problems (e.g. room capacity, proximity of exams, time and order of exams, etc). In the 1^{st} stage, Saturation Degree was used to generate a set of feasible solutions, where each criterion was dealt with individually. The 2^{nd} stage then heuristically improved these solutions simultaneously. A multi-criteria method called Compromise Programming [150] was used where the quality of the solutions was evaluated by the distance between them to an ideal point representing optimal solutions concerning all criteria. This technique was further studied in [116] by **Petrovic and Bykov**. They based their multi-criteria approach on the Great Deluge algorithm [75]. A reference point provided by users was used to draw a trajectory in the criteria space. The criteria weights can be dynamically changed to guide the search, starting from random points, towards the reference point. It aims at the ideal point in the criteria space. However, the initial weights needed to be set were dependant on the problems. Also the search was not guaranteed to converge. Published results from [38] were used as the reference points of the approach and the final results were better on some of the benchmark problems tested. These approaches provided the flexibility for timetablers to obtain desired solutions by managing the weights of different constraints.

2.6 Hyper-heuristics

The dependence upon parameter tuning or the way of embedding domain knowledge (i.e. the hard coding of hard and soft constraints) impacts upon meta-heuristic development for examination timetabling. Some of the most effective techniques on the benchmark data in the literature are meta-heuristics. However, most of these methods represent a tailor made approach for one particular problem (in this case, exam timetabling). Such methods usually work poorly or are not capable of dealing with other problems. Indeed, it can be the case that such methods do not work consistently across exam timetabling problem instances. Often, parameter tuning can play a significant role. The effort of tuning parameters to fit new problems can be thought of as being as difficult as that of developing new approaches. This well-known issue has led a number of researchers to develop new technologies aimed at operating at a higher level of generality.

Hyper-heuristics are motivated by such observations and are attracting an increased level of research attention. The term can be seen as representing *heuristics that choose heuristics*, i.e. a search space of heuristics is the focus of attention rather than a search space of solutions (as is the case with most implementations of meta-heuristics [26, 124]). The aim is to develop more *general* approaches rather than to beat the fine-tuned and problem specific approaches which often require much effort on the tuning of parameters and are usually only appropriate for specific problems.

As mentioned above in Section 2.4.1, **Ross, Hart and Corne** [126] suggested that a Genetic Algorithm might be successfully employed in searching for a good algorithms rather than specific solutions. In [138], **Terashima-Marin, Ross and Valenzuela-Rendon** investigated using Evolutionary Algorithms to search for solution construction strategies.

Ahmadi et al [5] in 2003 developed a Variable Neighbourhood Search to find good combinations of parameterised heuristics for different exam timetabling problems. Permutations of the low level heuristics (i.e. seven exam selection, two timeslot selection and three room selection heuristics) and their associated parameters (weights) were employed to construct solutions.

Ross, Marin-Blazquez and Hart in 2004 [128] developed a general steady state Genetic Algorithm to search within a simplified search space of problem-state descriptions, which were corresponding to the events and timeslots picking heuristics to construct solutions. Since the search of the Genetic Algorithm was on heuristics rather than actual solutions, three different fitness functions were tested. The descriptions of the problem state (corresponding to heuristics) were experimentally studied with respect to these fitness functions. Promising results for both the benchmark course and exam timetabling problems demonstrated valuable potential research directions of this approach for a range of problems.

Kendall and Hussin in 2004 [90, 91] investigated a Tabu Search based hyper-heuristic based on the work in [29] where both moving strategies and constructive graph heuristics were employed as low level heuristics. The algorithms were tested on exam timetabling problems in the MARA University of Technology [89] and it was shown that it produced better results compared with solutions that were generated manually.

Burke et al [20, 42] investigated employing Case-Based Reasoning (see [96]), a knowledge based technique, as a heuristic selector for solving both course and exam timetabling problems. In [42] (2006), knowledge discovery techniques were employed to discover the most relevant features used in evaluating the similarity between problem solving situations. The objective was to choose the best heuristics from the most similar previous problem solving situation to construct good solutions for the problem in hand. The issue of defining the similarity between exam timetabling problems has also been studied in [21] in terms of choosing the best problem solving method. In [20] (2005), different ways of hybridising the low level graph heuristics (with and without CBR) were compared for solving the Toronto datasets. It was shown that employing knowledge based techniques rather than randomly/systematically hybridising heuristics in a hyper-heuristic framework presented good results. **Yang and Petrovic** [149] employed Case-Based Reasoning to choose graph heuristics to construct initial solutions for the Great Deluge algorithm and obatined the best results reported in the literature for several Toronto instances at the time. Attribute graphs studied in [33] were employed to model the constraints in the problems so that previous problems with similar constraints were retrieved to solve the problems in hand by reusing the most appropriate graph heuristics.

Burke et al [34] (2006) investigated using Tabu Search to search permutations of graph heuristics to construct solutions for timetabling problems. A different number of low level graph heuristics were studied in this graph based hyper-heuristic to adaptively assign the most difficult exams at different stages of solution construction. It was observed that the greater the number of intelligent low level heuristics, the better the performance may be. However, the size of the search space will grow significantly, thus the computational time may be an issue. The results on both the course and exam timetabling problems were competitive with the best state-of-the-art approaches reported in the literature and demonstrated the simplicity and efficiency of this general approach. **Qu and Burke** [118] further investigated the effect of employing different high level search algorithms (i.e. Steepest Descent, Tabu Search, Iterated Local Search and Variable Neighbourhood Search) in the unified graph based hyper-heuristic framework. Experimental results demonstrated that the method of search by different high level heuristics within the search space of graph heuristics was not crucial. The characteristics of the neighbourhood structures and search space were analysed. It was shown that the exploration over the large solution space enabled the approach to obtain good results on both the exam and course timetabling problems.

Various strategies and methodologies have been employed as the high level selection methods in a hyper-heuristic framework to choose appropriate low level heuristics. These low level heuristics might be either construction or improvement heuristics. Such methods are laying the foundations of methodologies to automatically design and adapt timetabling heuristics. This has led to some work on analysing the search space of the heuristics (rather than solutions) with the goal of fundamentally understanding the search processes which underpin this new perspective on timetabling research [118].

2.7 Other Issues

2.7.1 Decomposition

The idea of decomposition is that large problems are broken into small sub-problems, for which optimal or high quality solutions can be obtained by relatively simple techniques as the search spaces of the sub-problems are significantly smaller than the original problem [46]. Although it has had some success [35], decomposition in timetabling has not attracted as much attention as might be expected because of two drawbacks. Firstly, early assignments may lead to later infeasibility, which was also a problem encountered in clustering and constructive methods in the early days of timetabling research. Secondly, globally high quality solutions may be missed as certain soft constraints cannot be evaluated when the problems are decomposed. The clustering methods studied in early timetabling research [49] can be seen as decomposition approaches in the sense that the exams are decomposed into conflict-free or low-conflict groups. Another way of decomposing the problems is by finding the largest clique in the graphs.

This was studied by **Carter, Laporte and Chinneck** [50] (1994) and employed in their later work [51] (1996). **Carter and Johnson** [48] (2001) improved the approach by assigning the exams in all of the almost-cliques as all of them are potentially the most difficult exams.

Burke and Newall [35] in 1999 investigated a decomposition approach by using sequential heuristics to assign the first set of n exams. This use of graph based heuristics and the employment of a look ahead method represented the mechanism which aims to avoid making early assignments which lead to later infeasibilities. The decomposed sub-problems were then solved by a Memetic Algoritm developed in [38]. The algorithms presented good results in terms of both the computational time and solution quality compared with a multi-start greedy method on the Toronto and Nottingham data. At the time of publication, this paper had some of the best results on the capacitated benchmark problems (Toronto c in Section 3.2).

Lin [98] (2002) developed a multi-agent algorithm where problems were divided into sub-problems and solved by each agent locally. A broker was used to solve the remaining schedules including those were de-allocated from local schedules. The global solutions were obtained by aggregating all the schedules generated by agents and the broker. Both the Toronto data and randomly generated exam timetabling problems were tested and compared with that of [126]. The approach worked well on sparsely scheduled problems but less well on dense problems.

2.7.2 Timetabling Systems

During the years, a number of timetabling systems for both course and exam timetabling have appeared in the literature. However many of them (especially before 1996) were specially developed for, and implemented at, particular institutions [49]. Not only were the design of algorithms but also practical issues were considered in the development of the systems.

Hansen and Vidal [85] (1995) presented a nationwide exam timetabling system which was reported to have been in use since 1992 to solve the problems of centralised planning of both oral and written examinations for 248 high schools in Denmark. The complex problem with a variety of objectives was described and solved by a four-phase process dealing with different objectives using different techniques and heuristics. Some issues including the preparation of data and the scheduling of exams were discussed. Some experiences during the development of the system and other practical issues including the maintenance of centralised information and communications were also discussed [84].

Colijn and Layfield (1995) [55] applied a multi-stage approach for the exam timetabling problem in the University of Calgary. In the 1^{st} stage timeslots' of exams and individual exams were moved to reduce students sitting two exams in a row. In the 2^{nd} stage, students taking three and four exams in a row were considered using the above similar process. The authors also considered the cases where timetables have to be modified in unforeseen circumstances [56] in the

2^{nd} stage of the approach, which was a highly interactive process within a visual interface where exams can be moved, added or removed from the timetables.

Lim et al [97] (2000) developed a timetabling system, which was a 3-tier client/server application, for both the course and exam timetabling problems at the National University of Singapore. The problems and the overall manual process were described. In the exam portion of the system, exams were weighted by three measures and assigned into the timeslots one by one using constraint propagation by an arc consistency algorithm. The timetables were generated by the system in a much shorter time and compared favourably with the manually generated ones from the previously used manual system. **Ho, Lim and Oon** [88] (2001) further developed the system by using a Tabu Search, which employed four moving operators to improve the solutions obtained. The Push Forward Insertion heuristic, which has been employed in vehicle routing problems, was used to help spreading the exams across timetables. Real problems were used to test the approach.

Dimopoulou and Miliotis [69] (2001) developed a timetabling system to deal with both the course and exam timetabling problems at Athens University of Economics and Business. Firstly, an Integer Programming method was developed based on MPCODE and XPRESS-MP packages. This approach was employed to assign groups of courses to groups of timeslots for the course timetabling problem. Based on the course timetables the initial exam timetables were built and were adjusted repeatedly by a heuristic approach which dealt with a number of constraints. This provided good and feasible solutions with minimum effort.

2.7.3 Timetabling Languages and Tools

Over the years, timetabling researchers have employed some general packages (such as ECLiPse for constraint logic programming [7]) to build timetabling systems. However, some packages and languages which are specialised on timetabling have also appeared to support representations and comparisons in timetabling research.

Burke, Kingston and Pepper [30] (1997) presented general requirements (generality, completeness and practicability) for building a standard data format for general timetabling problems based on set theory and logic. Examples were given to show how common constraints were modelled by using this data format. The objective is to provide an open way of making comparisons on results and exchanging data in timetabling research.

Tsang, Mills and Williams [141] (1999) developed a language to specify constraint satisfaction problems so that constraint programming systems can be easily implemented for exam timetabling problems. The aim was to build a high level system which abstracted the details as much as possible so that end users, without knowledge of both the constraint programming and host languages, can focus on the problem specific information. The language was briefly described and a real exam timetabling problem was used as the example to build the constraint satisfaction system.

Reis and Oliera [121] (2000) proposed a language, called UniLang, which used a list of synonyms to naturally represent data, various constraints, quality measure and solutions for general university timetabling problems. Eight classes of sub-problems in timetabling were defined and lots of examples were presented to interpret the language proposed. The language was converted into constraint logic programming in ECLiPse [7] for different problems.

Schearf and Di Gaspero [132] (2001) introduced a software tool called EASYLOCAL++ for the implementation of a family of local search algorithms (Hill Climbing, Simulated Annealing and Tabu Search) on general timetabling problems. This represented an object-oriented framework which consisted of a hierarchy of abstract classes to take care of different aspects of local search. The main characteristics of the tool were reusability and generality, which were interpreted using examples from school timetabling, course timetabling and exam timetabling problems. It was employed to develop a family of Tabu Search methods in [71].

De Causmaecker et al [63] (2002) discussed how the Semantic Web can be used in timetabling. They studied, layer by layer, how this technology can be applied to interpret problem specific knowledge in timetabling using XML. An upper level timetabling ontology was presented to demonstrate the ability to support the fast development of applications on different timetabling problems, whose constraints and resources can be easily identified.

Chand [53] (2004) proposed a constraint based general model where constraints are grouped as domain, spread and CountResource, based upon which timetabling data and constraints can be transferred into a relational database. Examples of exam timetabling data by **Burke, Elliman and Weare** [24] and course timetabling data by **Goltz and Matzke** [72] were presented using the model proposed. The author declared that the format can be extended to include other constraints and can be applied to different languages. The author also provided a brief review of the relevant work on modelling timetabling data.

2.7.4 Models and Complexity Issues

Over the last ten years, important issues concerning the models and the complexity of timetabling problems have been discussed in the literature. However, so far, there are still no universally accepted complete models. Not much deployment work has been carried out on this topic. The main area of research on complexity issues has been on school timetabling.

Cooper and Kingston [57] (1996) represented timetabling problems using a timetabling specification language called TTL. They proved that timetabling problems are NP-Complete in five independent ways which actually occur in practice. Prospects were discussed to overcome the special cases in real problems.

de Werra, Asratian and Durand [67] (2002) studied the complexity of some variants of class-teacher timetabling problems. A simple model of the problem was first given, followed by the extension where the classes were partitioned into groups. The authors showed that when there is a teacher giving lectures to three groups of classes, besides giving lectures to individual groups, the problem is NP-complete. A polynomial procedure to find a timetable of certain number of timeslots based on network flows was given for the problems where there were at most two groups of classes.

3 Benchmark Exam Timetabling Data

The high level of research interest in examination timetabling has led to the establishment of a variety of different benchmark problems which have been widely studied. The established benchmarks, with variants of standard defined measures, provided a way for meaningful scientific comparisons and the exchange of research achievements. However, there has been some confusion in the literature due to the circulation of two different versions of eight of these benchmark problems (from the University of Toronto datasets). One of the goals of this paper is to eradicate this confusion by establishing new names for each of the different versions. This, of course, means that we actually have 21 problems that have been studied in the literature (rather than 13). Another aim of this section of the paper is to summarise which of the methods that have appeared in the literature are the best on these benchmarks. This is particularly important given the confusion mentioned above.

3.1 University of Toronto Benchmark Data

Carter, Laptore and Lee [51] in 1996 introduced a set of 13 real-world exam timetabling problems from three Canadian highs schools, five Canadian universities, one American university, one British university and one university in Saudi Arabia. Over the years they were widely tested in exam timetabling research by different state-of-the-art approaches and have been seen as a benchmark in the field. As mentioned above, there has been an issue concerning the circulation of different sets under the same name. This is discussed at length below.

In the problem, to indicate the density of the conflicting exams in each of the instances, a *Conflict Matrix C* was defined where each element $c_{ij} = 1$ if exam i conflict with exam j (have common students), or $c_{ij} = 0$ otherwise. The *Conflict Density* represents the ratio between the number of elements of value "1" to the total number of elements in the conflict matrix.

Two variants of objectives were defined:

– to minimise the number of timeslots needed for the problem (graph coloring) (named as Toronto a in Table 5); and
– to minimise the average cost per student (named Toronto b in Table 5).

For the 1st objective, the aim is to find the feasible timetables of the shortest length. For the 2nd objective, an evaluation function was defined to calculate the cost of the timetables generated. For students sitting two exams s timeslots apart, the cost was assigned using proximity values w_s i.e. w_1=16, w_2=8, w_3=4 w_4=2 and w_5=1. The aim is to space out the conflicting exams within a limited number of timeslots. The authors also introduced seven real world applications with side constraints (i.e. maximum room capacity per timeslot, pre-assigned exams, maximum number of exams per timeslot, no x exams in y timeslots, etc). This objective was modified later and tested by a number of approaches (see below).

During the years, however, two versions of the data were circulated and were tested by different approaches. To distinguish the data tested and to build a standard benchmark for future use in timetabling, we have carefully examined the data that has appeared in two different forms under the same name for eight of these benchmark problems. We list the characteristics of these two versions of data in Table 4. We have post-fixed "I" and "II" respectively to the circulated datasets to distinguish between them. The post-fix "I" has been used for the problem instance which we believe has appeared most often in the literature. For the problem instances of post-fix "II", some confusion occurred as three of the instances (car91 I, car92 II and pur93 II) have conflicts on the number of enrolments (i.e. a different number of enrolments defined in two data files for each instance). Later on in this section we attempt to cast light on the question of which technique has been applied to which version of these instances in the literature.

Table 4. Characteristics of Two Versions of the Toronto Benchmark Datasets

Problem Instance	**Exams**	**Students**	**Enrolments**	**Conflict Density**	**Timeslots**
car91 I	682	16925	56877	0.13	35
car91 II	682	16925	56242/56877	0.13	35
car92 I	543	18419	55522	0.14	32
car92 II	543	18419	55189/55522	0.14	32
ear83 I	190	1125	8109	0.27	24
ear83 II	189	1108	8014	0.27	24
hec92 I	81	2823	10632	0.42	18
hec92 II	80	2823	10625	0.42	18
kfu93	461	5349	25113	0.06	20
lse91	381	2726	10918	0.06	18
pur93 I	2419	30032	120681	0.03	42
pur93 II	2419	30032	120686/120681	0.03	42
rye92	486	11483	45051	0.07	23
sta83 I	139	611	5751	0.14	13
sta83 II	138	549	5689	0.14	13
tre92	261	4360	14901	0.06	23
uta92 I	622	21266	58979	0.13	35
uta92 II	638	21329	59144	0.13	35
ute92	184	2749	11793	0.08	10
yor83 I	181	941	6034	0.29	21
yor83 II	180	919	6012	0.29	21

Table 5. Variants of the Toronto Benchmark Datasets

	Variants	**Objectives**
Toronto *a*	graph coloring	to minimise the number of timeslots needed
Toronto *b*	un-capacitated with cost	to space out conflicting exams within limited (fixed number of) timeslots
Toronto *c*	capacitated with cost	to minimise students sitting two exams in a row on the same day
Toronto *d*	capacitated with modified cost	same as above, and to minimise students sitting two exams overnight
Toronto *e*	estimated capacity and timeslots	to minimise students sitting two adjacent exams the same day

To avoid any further confusion, the definitive versions of these datasets are available at http://www.cs.nott.ac.uk/∼rxq/data.htm (together with all the other datasets discussed in this paper).

Burke, Newall and Weare [38] in 1996 modified the objective of the six real world problems introduced in [51] by considering the maximum room capacity per timeslot, and adjacent exams on the same day. In 1998 [39], timeslots in the problems were distinguished by setting three timeslots a day from Monday to Friday and one timeslot on Saturday. The objective is to minimise the students sitting two consecutive exams on the same day and overnight. These two variants are named Toronto *c* and *d* in Table 5. **Terashima-Marin et al** in 1999 [138] modified the dataset by assigning each problem instance an estimated number of timeslots and each timeslot an estimated maximum seats/capacity. This variant is named Toronto *e* in Table 5.

The approaches developed and tested on different variants of the Toronto datasets during the years are listed in Table 6 (ordered by the year in which the work was published). The values in "()" following the variants of the data in Table 6 give the number of problem instances tested by the corresponding approaches. Most of the work did not specify the exact characteristics of the data tested, and in many of the papers it is impossible to determine which version (I or II) of the data was tested (for the eight problematical instances). We have attempted, in Table 10 (by contacting authors), to clarify which versions of the datasets were used in each paper. If the entries are written in italics, we are not absolutely sure that the information with respect to this issue is correct. Otherwise we have had the situation confirmed by the authors concerned.

3.2 University of Nottingham Benchmark Data

Burke, Newall and Weare [38] in 1996 also introduced the 1994 exam timetabling data at the University of Nottingham as a benchmark. It was used later by a number of researchers on testing different approaches. Table 7 presents the characteristics of the dataset. We know that 23 is the least possible number of timeslots due to the limitations on the room capacity. The objective is to

Table 6. Approaches on the Toronto Benchmark Datasets

Reference	**Approach/Technique**	**Problem**
Carter et al [51] 1996	Graph heuristics with clique initialisation and backtracking	a(13), b(13)
Carter&Johnson [48] 1996	Almost cliques with sufficient density as the initialisation for graph heuristics	a(13)
Burke et al [38] 1996	Memetic Algorithm with hill climbing and light and heavy mutation	c(5)
Burke et al [39] 1998	Different initialisation strategies in Memetic Algorithms measured by diversity	d(3)
Burke et al [40] 1998	Non-determinisms introduced by selection strategies in graph heuristics	d(3)
Burke&Newall [35] 1999	Multi-stage Evolutionary Algorithm based on Memetic Algorithm	d(3)
Terashima -Marin et al [138] 1999	Genetic Algorithm with in-direct coding of the constructive strategies and heuristics	e(12)
Caramia et al [45] 2001	Iterated algorithm with novel improving factors	a(13), b(13), c(5)

Di Gaspero [71] &Schaerf 2001
Di Gaspero [70] 2002
	Adaptive tabu list and cost function in Tabu Search
	Multiple neighbourhood Tabu Search
		b(11), c(5), d(3)
		b(7),d(3)

White&Xie [144] [145] 2001&2004	Tabu Search with long term memory
	Relaxation on long and short term tabu lists
		b(2)
		b(7)

Paquete& [113] Stutzle 2002	Tabu Search with Lex-tie and Lex-seq strategies in the objective function	b(8)
Merlot et al [104] 2003	Constraint programming as initialisation for Simulated Annealing and hill climbing	a(12), b(12), c(5), d(2)
Casey& [52] Thompson 2003	GRASP with modified Saturation Degree initialisation and Simulated Annealing improvement	b(10)
Burke&Newall [36] 2003	Great Deluge with adaptive ordering as the initialisation	b(11)
Burke&Newall [37] 2004	Graph heuristics with adaptive heuristic modifier to dynamically order the exams	b(11), d(3)
Burke&Bykov et al [16] 2004	Time-predefined Great Deluge and Simulated Annealing	b(13), d(2)
Asmuni et al [8] 2004	Fuzzy rules with Largest Degree, Saturation Degree and Largest Enrolment	b(12)
Ross et al [128] 2004	Genetic Algorithm evolving constructive strategies and heuristics	e(12)
Burke et al [20] 2005	Hybridising graph heuristics in hyper-heuristic by CBR and systematic strategies	b(4)
Cote et al [61] 2005	Bio-objective Evolutionary Algorithm with local search operators in the recombination process	b(12)
Kendall&Hussin [90] 2005	Tabu Search based hyper-heuristic	b(8)
Yang&Petrovic [149] 2005	Similarity measure using fuzzy set on selecting hybridisations of Great Deluge and graph heuristics	b(12)

Table 6. (cont.) Approaches on the Toronto Benchmark Datasets

Reference	**Approach/Technique**	**Problem**
Abdullah et al [3] 2006	Large neighbourhood search with tree-based neighbourhood structure	b(12), c(5)
[4] 2006	Tabu Search based large neighborhood search	c(5)
Burke et al [34] 2006	Graph based hyper-heuristic using Tabu Search	b(11)
Qu&Burke [118] 2006	Graph based hyper-heuristic framework with different high level search algorithms	b(11)
Burke et al [22] 2006	Genetic Algorithms on selecting subset of neighborhoods in Variable Neighborhood Search	b(11)
Burke et al [42] 2006	Case based heuristic selection for the solution construction	b(11)

minimise the students sitting two consecutive exams on the same day. The data can be downloaded from http://www.cs.nott.ac.uk/∼rxq/data.htm.

In [39] (1998) the above problems were further constrained by modifying the objective function to consider also the consecutive exams overnight. In Table 7 we highlight these variants as Nottingham *a* and *b*. Table 8 presents the approaches applied on these datasets and the University of Melbourne datasets (see section 3.3 below) in the literature.

Table 7. Characterisitcs of the Nottingham Benchmark Datasets

	Nottingham *a*	Nottingham *b*
Exams	800	800
Students	7896	7896
Timeslots	23, 26	23
Enrolments	34265	34265
Conflicts	10034	10034
Capacity	1550	1550
Density	0.03	0.03
Objective	minimise adjacent exams on the same day	minimise adjacent exams on the same day and overnight

3.3 University of Melbourne Benchmark Data

Merlot et al [104] introduced exam timetabling datasets from the University of Melbourne at the PATAT conference in 2002. Two datasets were introduced. For these datasets, there were two timeslots on each day for each of the five workdays, and the capacity for each session varied. The availability of sessions for some of the exams was restricted. In one problem instance this prevented all feasible solutions so an alternate data set was created which allowed feasible solutions. These

Table 8. Approaches on the Nottingham and Melbourne Benchmark Datasets

Reference	**Approach/Technique**	**Problem**
Burke&Newall [37] 2004	Adaptive ordering based on graph heuristics using heuristic modifier	Nottingham *b*
Merlot et al [104] 2003	Constraint logic programming as initialisation for Simulated Annealing and hill climbing	Nottingham *a*,*b* Melbourne *I II*
Di Gaspero& Schearf [71] 2001	Tabu Search using exhaust and biased selection by the costs of exams	Nottingham *a*,*b*
Burke et al [16] 2004	Great Deluge with a number of runs of Saturation Degree	Nottingham *b*
Caramia et al [45] 2001	Iterative approach where the number of timeslots was gradually increased after greedy improvement	Nottingham *a*
Ahmadi et al [5] 2003	VNS to search permutations of heuristics and their weights	Nottingham *a*
Cote et al [61] 2005	Evolutionary algorithms with bio-objective constraint satisfaction	Nottingham *a* Melbourne *I*
Burke et al [38] 1996	Memetic algorithms with light & heavy mutations graph heuristic initialisation	Nottingham *a*
Burke&Newall [35] 1999	Multi-stage evolutionary algorithm initialised by graph heuristics with backtracking	Nottingham *b*
Burke et al [15] 2001	Multi-criteria approach dealing with 9 criteria based on initial solutions by Saturation Degree	Nottingham *b*

datasets can also be downloaded from http://www.cs.nott.ac.uk/∼rxq/data.htm. The Melbourne datasets are summarised in Table 9.

Table 9. Characterisitcs of the Melbourne Benchmark Datasets

	Exams	Timeslots	Students	Enrolls	Objective
I	521	28	20656	62248	minimise adjacent exams on the same day or overnight
II	562	31	19816	60637	same as above

3.4 Results on the Benchmark Problems

As mentioned above, there has been a large number of papers published which have worked with the datasets discussed above. In addition, the difficulties surrounding the publication of the Toronto datasets has led to some confusion over which methods were tackling which problems. Tables 10-12 attempt to clarify this. They list all of the methods which have addressed the Toronto problems and they attempt to illustrate which methods used which problems. This has been a difficult task and the authors would welcome additional information which we will use to keep an updated version of the table at http://www.cs.nott.ac.uk/∼rxq/data.htm. We would like to add more methods as they appear.

Table 10. Results in the literature on the two versions of the Toronto Dataset *b* (see Table 5). Values in italic represent that we are unsure about the accuracy in terms of the versions of the datasets used. Values in bold represent the best results reported. "-" represent the corresponding problem is not tested or a feasible solution cannot be obtained.

data set version I/II	Carter et al. (1996) [51] I	Caramia et al. (2001) [45] I	Di Gaspero & Schaerf (2001) [71] I	Di Gaspero (2002) [70] I	Paquete & Stutzle (2002) [113] I	Burke & Newall (2003) [36] I	Casey & Thompson (2003) [52] II	Merlot et al. (2003) [104] I
car91	*7.1*	*6.6*	*6.2*	*5.7*	-	4.65	*5.4*	5.1
car92	*6.2*	*6.0*	*5.2*	-	-	4.1	*4.4*	4.3
ear83	*36.4*	***29.3***	*45.7*	*39.4*	*38.9*	37.05	*34.8*	35.1
hec92	*10.8*	***9.2***	*12.4*	*10.9*	*11.2*	11.54	*10.8*	10.6
kfu93	*14.0*	*13.8*	*18.0*	-	*16.5*	13.9	*14.1*	13.5
lse91	*10.5*	***9.6***	*15.5*	*12.6*	*13.2*	10.82	*14.7*	10.5
rye92	*7.3*	***6.8***	-	-	-	-	-	-
sta83	*161.5*	*158.2*	*160.8*	*157.4*	*168.3*	168.73	***134.9***	157.3
tre92	*9.6*	*9.4*	*10.0*	-	*9.3*	8.35	*8.7*	8.4
uta92	*3.5*	*3.5*	*4.2*	*4.1*	-	3.2	-	3.5
ute92	*25.8*	***24.4***	*27.8*	-	*29.0*	25.83	*25.4*	25.1
yor83	*41.7*	***36.2***	*41.0*	*39.7*	*38.9*	37.28	*37.5*	37.4

Table 10. (cont.) Results in the literature on the variants of the Toronto Dataset *b*. *The value presented here is different from that in [149], as a different objective function was used in [149].

data set version I/II	Burke & Newall (2004) [37] I	Burke et al. (2004) [16] I	Asmuni el al (2005) [8] I	Cote et al (2005) [61] I	Kendall & Hissan (2005) [90] I	Yang & Petrovic (2005) [149] I	Abdullah et al (2006) [3] I	Burke et al (2006) [34] I	Burke et al (2006) [22] I
car91	5.0	4.8	5.29	*5.4*	5.37	**4.5**	5.2	5.36	4.6
car92	4.3	4.2	4.56	*4.2*	4.67	**3.93**	4.4	4.53	4.0
ear83	36.2	35.4	37.02	*34.2*	40.18	33.7	34.9	37.92	32.8
hec92	11.6	10.8	11.78	*10.4*	11.86	10.83	10.3	12.25	10.0
kfu93	15.0	13.7	15.81	*14.3*	15.84	13.82	13.5	15.2	**13.0**
lse91	11.0	10.4	12.09	*11.3*	-	10.35	10.2	11.33	10.0
rye92	-	8.9	10.35	8.8	-	8.53	8.7	-	-
sta83	161.9	159.1	160.42	*157.0*	157.38	158.35*	159.2	158.19	159.9
tre92	8.4	8.3	8.67	*8.6*	8.39	7.92	8.4	8.92	**7.9**
uta92	3.4	3.4	3.57	*3.5*	-	**3.14**	3.6	3.88	3.2
ute92	27.4	25.7	27.78	*25.3*	27.6	25.39	26.0	28.01	24.8
yor83	40.8	36.7	40.66	*36.4*	-	36.35	**36.2**	41.37	37.28

Table 11. Results in the literature on the Toronto Dataset I, variants *a* (left column) and *e* (right column).

data set	Carter et al. [51] (1996)	Caramia et al. [45] (2001)	Merlot et al. [104] (2003)	Terashima-Marin et al [138] (1999)	Ross et al [128] (2004)
car91	**28**	**28**	30	**130**	283
car92	**28**	**28**	31	**285**	542
ear83	**22**	**22**	24	**723**	958
hec92	**17**	**17**	18	**154**	224
kfu93	**19**	**19**	21	**223**	226
lse91	**17**	**17**	18	**221**	263
rye92	**21**	**21**	22	**671**	832
pur93	**35**	36	-	-	-
sta83	**13**	**13**	**13**	**821**	1058
tre92	**20**	**20**	21	**586**	604
uta92	32	**30**	32	**594**	855
ute92	**10**	**10**	11	**902**	967
yor83	**19**	**19**	23	**708**	758

Table 12. Results in the literature on the Toronto Dataset I, variants *c* (upper part) and *d* (lower part).

dataset	car92	car91	kfu93	tre92	uta92	pur93
Burke et al (1996) [38]	81	331	974	3	772	-
Caramia et al (2001) [45]	74	268	912	2	680	-
Di Gaspero&Schearf (2001) [71]	88	424	512	4	554	-
Merlot et al (2003) [104]	**31**	**158**	247	**0**	334	-
Abdullah et al (2006) [3]	37	278	548	**0**	**300**	-
Abdullah et al (2006) [4]	47	525	**206**	4	310	-
Burke et al (1998) [40]	2218	-	3256	-	**2440**	-
Burke&Newall (1999) [35]	1665	-	1388	-	-	**63824**
Di Gaspero&Schearf (2001) [71]	3048	-	1733	-	-	123935
Merlot et al (2003) [104]	1744	-	**1082**	-	-	-
Burke&Newall (2004) [37]	1775	-	1422	-	-	97237
Burke et al (2004) [16]	**1506**	-	1321	-	-	-

Table 13 presents the results from different approaches applied on the Nottingham datasets a and b (see Table 7) and Melbourne datasets I and II (see Table 7) in the literature.

Table 13. Results on the Nottingham Dataset a and b and Melbourne Dataset I and II. Values in bold represent the best results reported. "-" represent the corresponding problem is not tested or a feasible solution cannot be obtained.

Nottingham	Burke et al [38] (1996)	Di Gaspero&Schearf [71] (2001)	Caramia et al [45] (2001)	Merlot et al [104] (2003)	Abdullah et al [4] (2006)
a (26 slots)	53	11	44	**2**	18
a (23 slots)	269	123	-	**88**	-
	Burke& Newall [35] (1999)	Di Gaspero &Schearf [71] (2001)	Merlot et al [104] (2003)	Burke& Newall [37] (2004)	Burke et al [16] (2004)
b	519	751	401	545	**384**
Melbounre	Merlot et al [104] (2003)				
I	1072				
II	1115				

Table 10-13 also illustrate which of the methods are most effective in terms of solution quality. The very best results are presented in bold. We have not listed computational times for these reasons. Firstly, many of these papers do not report the relevant times. Secondly, comparisons across very different platforms are impossible. Thirdly, examination timetabling is a problem which is almost always tackled weeks or months before the timetable will be used. As such, it is definitely not a time critical problem and there are many real world scenarios where it would be perfectly reasonable to leave an algorithm running overnight or even over a weekend.

4 Conclusions and Future Directions

Timetabling research started with simple sequential techniques in the 1960s. Constraint based techniques appeared later and still play a significant role in timetabling today. Recent research in exam timetabling is dominated by meta-heuristics and their integrations/hybridisations with a variety of techniques, including many of the early techniques. Local search based techniques, multi-criteria techniques and approaches which aim to be more general than the state of the art have also presented interesting outcomes.

Recent innovations have utilised different mechanisms in exam timetabling and they cover a variety of new techniques including Variable Neighbourhood

Search, Iterative Local Search, GRASP, and hyper-heuristics with the aim of developing more powerful, efficient, effective and more general approaches.

In the following section we will draw upon the above discussion to highlight a number of conclusions and to present some ideas for future research which are generated by these conclusions. It is worth noting that Burke et al [19] outline some future research directions in nurse rostering. There is of course, some synergy with issues discussed in [19] and these are alluded to here.

Meta-heuristics have attracted the most attention in exam timetabling research.

In addition to a comprehensive treatment of Tabu Search, Simulated Annealing, Genetic Algorithms and various hybrids, some new exam timetabling techniques have been presented. For example, GRASP and Iterated Local Search build on the similar idea of exploring wider areas of the search space by using a multi-start greedy search technique to reduce the risk of being stuck in local optima. Variable Neighbourhood Search escapes from local optima by switching between the search spaces defined by different neighbourhood structures. Large neighbourhood search fulfils this by extending the flexibility of moves within the search space. In summary, these techniques extend the idea of helping the search to escape from local optima in a variety of ways and have obtained promising progress on a wide range of exam timetabling problems. The development of these new techniques has opened up a wide variety of new research directions such as exploring alternative neighborhood structures, new multi-start techniques, hybridisation issues, alternative operators and many others. One of the key research goals is to provide an appropriate balance between *exploration* and *exploitation* in search algorithms.

Extensive study is also required in how to determine appropriate parameter settings for meta-heuristic methods. The determination of suitable initialisation methods and in-depth analysis of the effects of initialisation on a range of meta-heuristics is another important exam timetabling research topic. Theoretical issues (such as phase transition) and multi-criteria techniques represent other important directions in meta-heuristic research. Evolutionary methods and other population based techniques represent a significant proportion of the meta-heuristic literature on exam timetabling. There are many research directions generated by considering the hybridisation of meta-heuristic methods particularly between population based methods and other approaches. A study of coding issues represents a new and promising direction in both evolutionary algorithms and hyper-heuristic research.

More General and adaptive techniques have been also studied.

Hyper-heuristics are concerned with searching for appropriate heuristics rather than concentrating on the problem specific details of actual solutions, which have been the focus of traditional search algorithms. This opens up a new direction of research and represents much potential in both practical applications and theoretical study. Adaptive techniques have also recently emerged where information collected during the problem solving is used to guide the search. Some work has been carried out on knowledge based techniques where the experience from previous problem solving drives the search. Further investigation of knowledge based techniques has the promise to underpin the development of fundamentally more general approaches. The goal is to deal automatically with different problems in a dynamic way so that extra effort is not needed to fine-tune the approach. An analysis of heuristics/techniques concerning the nature of the search space could be beneficial. It is generally accepted that little is known about the nature of search spaces, especially for complex real-world problems such as exam timetabling.

Hybridisations of different techniques have been investigated in exam timetabling.

Although different authors have favored different approaches, it has been observed that hybrid approaches are usually superior to *pure* algorithms. For example, all of the recent work on constraint based techniques represent hybridisation with other techniques (see Section 2.2). However, in most of the cases, methodologies are hybridised in a sequential way rather than being efficiently integrated. More work needs to be done to not just simply combine but rather more meaningfully integrate different methodologies efficiently. For example, in memetic algorithms, local search is used co-operatively after each generation. In a hyper-heuristic, one approach taken is that, low level heuristics are searched and combined adaptively during the problem solving. Further in-depth analysis and investigation can underpin the design and development of more powerful techniques.

Benchmark exam timetabling problems have been formed and thoroughly tested by a number of approaches.

Recent state-of-the-art approaches in exam timetabling research have carried out comparisons on the benchmark problems (see Section 3) that have appeared over the last ten years. This has led to fundamental developments in exam timetabling research. However, these problems still represent simplified versions of the problem. In the wider context of scheduling research, there has been much recent debate about the "gap between theory and practice". The same is true for exam timetabling research. A major research direction is represented by exploring the wide range of research issues that are opened up by considering the high levels of complexity that are generated by real world problems [100]. In addition, there still is no widely accepted universal data format and standard timetabling languages. The establishment of quality measures by standard techniques on both solution quality (objective functions) and computational time for exam timetabling problems also requires much work and is crucial in conjunction with the formation of benchmarks. The requirements for the development of automatic tools to support timetabling staff to save significant development time still exists. To encourage such development, we are building up an archive where benchmark exam timetabling problems are collected, together with a categorised updated timetabling bibliography (after 1995). We welcome contributions to this exam timetabling archive. It is held at http://www.cs.nott.ac.uk/∼rxq/bibliography.htm.

In summary, it is possible to draw a number of conclusions from an in-depth survey of the examination timetabling literature in the last ten years. Firstly, there has been a significant number of research successes in that time. Secondly, the current state of the art provides a strong platform from a range of important research directions. Thirdly, future research requires a particular emphasis on the complexity of real world issues and this requires the establishment of more benchmarks that are drawn from real world problems. Fourthly, raising the level of generality of decision support systems (including for exam timetabling) represents an emerging theme. Finally, it is worth noting that successful papers in exam timetabling have been authored by researchers from a range of disciplinary backgrounds and particularly at the interface of Operational Research and Artificial Intelligence. Such interdisciplinary collaboration is crucial to scientific progress in the area. It is clear from this analysis of the literature that the future of exam timetabling research is inter-disciplinary.

References

1. E.H.L. Aarts, J. Korst and W. Michiels (2005). Simulated annealing. In: E.K. Burke and G. Kendall (eds.) (2005). *Introductory Tutorials in Optimisation, Decision Support and Search Methodology.* ISBN: 0387234608, Springer. Chapter 7, 187-211.
2. E.H.L. Aarts and J. Korst. (1989). *Simulated Annealing and Boltzmann Machines.* New York: John Wiley & Sons.
3. S. Abdullah, S. Ahmadi, E.K. Burke and M. Dror. (2006). Investigating Ahuja-Orlins large neighbourhood search for examination timetabling. Accepted by *OR Spectrum*, 2006.
4. S. Abdullah, S. Ahmadi, E.K. Burke B., Dror M. and McCollum B. (2006). A tabu based large neighbourhood search methodology for the capacitated examination timetabling problem. Accepted by *Journal of Operational Research*, 2006.
5. S. Ahmadi, R. Barrone, P. Cheng, P. Cowling and B. McCollum. (2003). Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem. In: Proceedings of Multidisciplinary International Scheduling: Theory and Applications (MISTA 2003), Nottingham, August 13-16, 2003, 155-171. ISBN: 0-9545821-2-8.
6. R.K. Ahuja, J.B. Orlin and D. Sharma. (2001). Multi-exchange neighbourhood search algorithm for capacitated minimum spanning tree problem. *Mathematical Programming*, **91**: 71-97.
7. F. Ajili and M.W. Wallace. (2003). Hybrid problem solving in ECLiPSe. In: M. Milano (ed). *Constraint and Integer Programming: Toward a Unified Methodology*, Chapter 6, 169 -201, Kluwer Academic Publishers.
8. H. Asmuni, E.K. Burke, J. Garibaldi and B. McCollum: (2004). Fuzzy multiple ordering criteria for examination timetabling. In: E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. 334-353.

9. V.A. Bardadym. (1996). Computer-aided school and university timetabling: The new wave. In: E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 22-45.
10. P. Boizumault, Y. Delon and L. Peridy. (1996). Constraint logic programming for examination timetabling. *Journal of Logic Programming*, **26**(2): 217-233.
11. S.C. Brailsford, C.N. Potts and B.M. Smith. (1999). Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research*, **119**: 557-581.
12. D. Brelaz. (1979) New methods to color the vertices of a graph. *Communication of the ACM*, **22**(4): 251-256.
13. S. Broder. (1964). Final examination scheduling. *Communications of the ACM*, **7**: 494-498.
14. B. Bullnheimer. (1998) An examination scheduling model to maximize students study time. In: E.K. Burke and M.W. Carter (eds). (1998). Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference. Springer Lecture Notes in Computer Science, vol. 1408. 78-91.
15. E.K. Burke, Y. Bykov and S. Petrovic. (2001). A multicriteria approach to examination timetabling. In: E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. 118-131.
16. E.K. Burke, Y. Bykov, J.P. Newall, and S. Petrovic. (2004). A time-predefined local search approach to exam timetabling problems. *IIE Transactions*, **36**(6): 509-528.
17. E.K. Burke and M.W. Carter (eds). (1998). Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference. Springer Lecture Notes in Computer Science, vol. 1408. ISBN 3-540-64979-4.
18. E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. ISBN 3-540-40699-9.
19. E.K. Burke, P. De Causmaecker, G. Vanden Berghe and H. Van Landeghem. (2004). The state of the art of nurse rostering. *Journal of Scheduling*, **7**(6): 441-499.
20. E.K. Burke, M. Dror, S. Petrovic and R. Qu. (2005). Hybrid graph heuristics in hyper-heuristics applied to exam timetabling problems. In: B.L. Golden, S. Raghavan and E.A. Wasil (eds.): *The Next Wave in Computing, Optimization, and Decision Technologies*. 79-91. Springer, Maryland, Jan 2005.
21. E.K. Burke, A.J. Eckersley, B. McCollum, S. Petrovic and R. Qu. (2004). Analysing similarity in examination timetabling. In: E.K. Burke and M. Trick (eds). (2004). Proceedings of The 5th International Conference on the Practice and Theory of Automated Timetabling. 18th-20th Aug, Pittsburgh, PA USA. 89-106.
22. E.K. Burke, A.J. Eckersley, B. McCollum, S. Petrovic and R. Qu. (2006). Hybrid variable neighbourhood approaches to university exam timetabling. Technical Report NOTTCS-TR-2006-2, School of CSiT, University of Nottingham.
23. E.K. Burke, D.G. Elliman, P.H. Ford and R.F. Weare. (1996). Examination timetabling in British universities: A survey. In: E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 76-90.
24. E.K. Burke, D.G. Elliman and R.F. Weare. (1994). A genetic algorithm for university timetabling. Proceedings of the AISB Workshop on Evolutionary Computing. 11th-13th Apr, 1994. University of Leeds, UK.

25. E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. ISBN 3-540-42421-0.
26. E.K. Burke, E. Hart, G. Kendall, J. Newall, P. Ross and S. Schulenburg. (2003). Hyper-Heuristics: An Emerging Direction in Modern Search Technology. In: Handbook of Meta-Heuristics, (eds. F. Glover and G. Kochenberger), Ch. 16, 457-474, Kluwer.
27. E.K. Burke, K. Jackson, J.H. Kingston and R. Weare. (1997). Automated university timetabling: The state of the art. *The Computer Journal*, **40**(9): 565-571.
28. E.K. Burke and G. Kendall (eds.) (2005). *Introductory Tutorials in Optimisation, Decision Support and Search Methodology.* ISBN: 0387234608, Springer.
29. E.K. Burke, G. Kendall and E. Soubeiga. (2003). A tabu-search hyper-heuristic for timetabling and rostering. *Journal of Heuristics*, **9**: 451-470.
30. E.K. Burke, J. Kingston and P.A. Pepper. (1997) A standard data format for timetabling instances. In: E.K. Burke and M.W. Carter (eds). (1998). Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference. Springer Lecture Notes in Computer Science, vol. 1408. 215-224.
31. E.K. Burke, J.H. Kingston and D. de Werra. (2004). Applications to timetabling. In: J. Gross and J. Yellen (eds.) *The Handbook of Graph Theory*, Chapman Hall/CRC Press, 2004, 445-474.
32. E.K. Burke and J.D. Landa Silva. (2004). The design of memetic algorithms for scheduling and timetabling problems. In: W.E. Hart, N. Krasnogor and J.E. Smith (editors) (2004). Recent Advances in Memetic Algorithms and Related Search Technologies. Studies in Fuzziness and Soft Computing 166, Springer, Berlin, Heidelberg, NewYork. 289-312.
33. E.K. Burke, B. MacCarthy, S. Petrovic and R. Qu. (2000). Structured cases in CBR - Re-using and adapting cases for timetabling problems. *Knowledge-Based Systems*, **13**(2-3): 159-165.
34. E.K. Burke, B. McCollum, A. Meisels, S. Petrovic and R. Qu. (2006). A graph based hyper-heuristic for exam timetabling problems. *European Journal of Operational Research*, in press, Available online November 2005.
35. E.K. Burke and J.P. Newall. (1999) A multi-stage evolutionary algorithm for the timetable problem. *IEEE Transactions on Evolutionary Computation*, **3**(1): 63-74.
36. E.K. Burke and J.P. Newall. (2003). Enhancing timetable solutions with local search methods. In: E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. 195-206.
37. E.K. Burke and J.P. Newall. (2004). Solving examination timetabling problems through adaptation of heuristic orderings. *Annals of Operational Research*, **129**: 107-134.
38. E.K. Burke, J.P. Newall and R.F. Weare. (1996). A memetic algorithm for university exam timetabling. In: E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 241-250.
39. E.K. Burke, J.P. Newall and R.F. Weare. (1998). Initialization strategies and diversity in evolutionary timetabling. *Evolutionary computation*, **6**(1): 81-103.
40. E.K. Burke, J.P. Newall and R.F. Weare. (1998). A simple heuristically guided search for the timetable problem. In: Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems (EIS98), 574-579.
41. E.K. Burke and S. Petrovic. (2002). Recent research directions in automated timetabling. *European Journal of Operational Research*, **140**(2): 266-280.

42. E.K. Burke, S. Petrovic and R. Qu. (2006). Case-based heuristic selection for timetabling problems. *Journal of Scheduling*, **9**: 99-113, 2006.
43. E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. ISBN 3-540-61794-9.
44. E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. ISBN 3-540-30705-2.
45. M. Caramia, P. DellOlmo and G.F. Italiano. (2001). New algorithms for examination timetabling. In: S. Naher and D. Wagner (eds): Algorithm Engineering 4th International Workshop, Proceedings WAE 2000. Springer Lecture Notes in Computer Science, vol. 1982, 230-241.
46. M.W. Carter. (1983). A decomposition algorithm for practical timetabling problems. Technical Paper 83-06, Department of Industrial Engineering, University of Toronto.
47. M.W. Carter. (1986). A survey of practical applications of examination timetabling algorithms. *Operations Research*, **34**(2): 193-202.
48. M.W. Carter and D.G. Johnson. (2001). Extended clique initialisation in examination timetabling. *Journal of Operational Research Society*, **52**: 538-544.
49. M.W. Carter and G. Laporte. (1996). Recent developments in practical examination timetabling. In: E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 3-21.
50. M.W. Carter, G. Laporte and J.W. Chinneck. (1994). A general examination scheduling system. *Interfaces*, **24**: 109-120.
51. M.W. Carter, G. Laporte and S.Y. Lee. (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of Operational Research Society*, **47**(3): 373-383.
52. S. Casey and J. Thompson. (2003). GRASPing the examination scheduling problem. In: E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. 232-244.
53. A. Chand. (2004). A constraint based genetic model for representing complete University timetabling data. In: E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. 125-150.
54. W.F. Clocksin and C.S. Mellish. (2000). *Programming in PROLOG.* fourth edition, Springer-Verlag.
55. A.W. Colijn and C. Layfield. (1995). Conflict reduction in examination schedules. In: E.K. Burke and P. Ross (eds). (1995). Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling. 30th Aug - 1st Sep 1995. Napier University, Edinburgh, UK. 297-307.
56. A.W. Colijn and C. Layfield. (1995). Interactive improvement of examination schedules. In: E.K. Burke and P. Ross (eds). (1995). Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling. 30th Aug - 1st Sep 1995. Napier University, Edinburgh, UK. 112-121.
57. Cooper T.B. and Kingston J.H. (1996). The complexity of timetable construction problems. In: Burke E.K. and Ross P. (eds.) The Practice and Theory of Automated Timetabling: Selected Papers the First International Conference, Springer Lecture Notes in Computer Science, vol. 1153, 283-295.

58. D. Corne, P. Ross and H. Fang. (1994). Evolutionary timetabling: Practice, prospects and work in progress. In: P. Prosser (ed). Proceedings of UK Planning and Scheduling SIG Workshop.
59. P.H.Corr, B.McCollum, M.A.J.McGreevy, P.McMullan. (2006). A New Neural Network Based Construction Heuristic for the Examination Timetabling Problem. Accepted by The International Conference on Parallel Problem Solving From Nature (PPSN) 2006, Reykjavik, Iceland, September 2006.
60. D. Costa and A. Hertz. (1997). Ant can colour graphs. *Journal of Operational Research Society*, **48**: 295-305.
61. P. Cote, T. Wong and R. Sabouri. (2005). Application of a hybrid multi-objective evolutionary algorithm to the uncapacitated exam proximity problem. In: E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. 151-168.
62. P. David. A constraint-based approach for examination timetabling using local repair techniques. In: E.K. Burke and M.W. Carter (eds). (1998). Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference. Springer Lecture Notes in Computer Science, vol. 1408. 169-186.
63. P. De Causmaecker, Y. Lu, P. Demeester and G. Vander Berghe. (2002). Using web standards for timetabling. In: E.K. Burke and P. De Causmaecker (eds). (2002). Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling. 21st-23rd August 2002. KaHo St.-Lieven, Gent, Belgium. 238-257.
64. T.A. Duong, K.H. Lam (2004). Combining constraint programming and simulated annealing on university exam timetabling. In: Proceedings of the 2nd International Conference in Computer Sciences, Research, Innovation & Vision for the Future (RIVF2004), Hanoi, Vietnam, February 2-5, 2004, 205-210.
65. D. de Werra. (1985). An introduction to timetabling. *European Journal of Operational Research*, **19**: 151-162.
66. D. de Werra. (1997). Restricted coloring models for timetabling. *Discrete Mathematics*, 165/166: 161-170.
67. D. de Werra, A.S. Asratian and S. Durand. (2002). Complexity of some special types of timetabling problems. *Journal of Scheduling*, **5**: 171-183.
68. M. Dorigo and C. Blum. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*. 344(2-3): 243-278.
69. M. Dimopoulou and P. Miliotis. (2001). Implementation of a university course and examination timetabling system. *European Journal of Operational Research*, **130**: 202-213.
70. L. Di Gaspero. (2002). Recolour, shake and kick: A recipe for the examination timetabling problem. In: E.K. Burke and P. De Causmaecker (eds). (2002). Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling. 21st-23rd August 2002. KaHo St.-Lieven, Gent, Belgium. 404-407.
71. L. Di Gaspero and A. Schaerf. (2001). Tabu search techniques for examination timetabling. In: E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. 104-117.
72. H.J. Goltz and D. Matzke. (1999). University timetabling using constraint logic programming. In: Practical Aspects of Declarative Languages, Springer Lecture Notes in Computer Science, vol. 1551, 320-334.
73. K.A. Dowsland. (1996). Simulated annealing solutions for multi-objective Scheduling and Timetabling. In: V.J.R. Smith, I.H. Osman, C.R. Reeves and G.D. Smith (eds.), Modern Heuristic Search Methods, John Wiley, 155-166.

74. K.A. Dowsland and J. Thompson. (2005). Ant colony optimization for the examination scheduling problem. *Journal of Operational Research Society*, **56**: 426-438.
75. G. Dueck. (1993). New optimization heuristics: the great deluge and the recordto-record travel. *Journal of Computational Physics*, **104**: 86-92.
76. K. Easton, G. Nemhauser and M. Trick. (2004) Sports scheduling. In: J. Leung (ed.) *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, Chapter 52. CRC Press.
77. W. Erben. (2001). A grouping genetic algorithm for graph colouring and exam timetabling. In: E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. 132-156.
78. Fogel D. (1994). An Introduction to Simulated Evolutionary Optimization. *IEEE Transactions on Neural Networks.* 5(1): 3-14.
79. M.R. Garey and D.S. Johson. (1979). *Computers and intractability - a guide to NP-completeness*. San Francisco: W.H. Freeman and Company.
80. M. Gendreau and J.Y. Potvin. (2005). Tabu Search. In: E.K. Burke and G. Kendall (eds.) (2005). Introductory Tutorials in Optimisation, Decision Support and Search Methodology. ISBN: 0387234608, Springer. Chapter 6, 165-186.
81. F. Glover and M. Laguna. (1993). Tabu search. In: C.R. Reeves (ed.). *Modern Heuristic Techniques for Combinatorial Problems*. Oxford: Scientific Publications.
82. F. Glover and K. Kochenberher. (2003). *Handbook of Meta-heuristics*, Kluwer.
83. P. Hansen and N. Mladenovic. (2001). Variable neighbourhood search: Principles and applications. *European Journal of Operational Research*, **130**: 449-467.
84. M.P. Hansen and V. Lauersen and R.V.V. Vidal. (1995) Nationwide scheduling of examinations: lessons from experience. In: E.K. Burke and P. Ross (eds). (1995). Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling. 30th Aug - 1st Sep 1995. Napier University, Edinburgh, UK. 468-473.
85. M.P. Hansen and R.V.V. Vidal. (1995). Planning of high school examinations in Denmark. *European Journal of Operational Research*, **87**: 519-534.
86. P.V. Hentenryck (1989). Constraint satisfaction in logic programming. Logic Programming Series, MIT Press, Cambridge, MA, 1989.
87. P.V. Hentenryck (1999). *The OPL Optimization Programming Language.* The MIT Press.
88. W.K. Ho, A. Lim and W.C. Oon. (2001). Maximizing paper spread in examination timetabling using a vehicle routing method. In: Proceedings of 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI01), 359-366.
89. N. Hussin. Tabu search based hyper-heuristic approaches for examination timetabling. PhD Thesis, Department of Computer Science, University of Nottingham, UK, November 2005.
90. G. Kendall and N.M. Hussin. (2004). A tabu search hyper-heuristic approach to the examination timetabling problem at the MARA university of technology. In: E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. 199-218.
91. G. Kendall and N.M. Hussin. (2004). An Investigation of a tabu search based hyper-heuristic for examination timetabling. In: Kendall G., Burke E. and Petrovic S. (eds), Selected papers from Multidisciplinary Scheduling; Theory and Applications, 309-328, 2005.
92. J.H. Kingston. (1995). Bibliography on practice and theory of automated timetabling. http://liinwww.ira.uka.de/bibliography/Misc/timetabling.html

93. J. Kolodner. (1993). *Case-based reasoning.* Morgan Kaufmann Publishers, Inc. San Mateo.
94. R. Kwan. (2004). Bus and train driver scheduling. In: J. Leung (ed.), *Handbook of Scheduling: Algorithms, Models, and Performance.* Analysis CRC Press. Chapter 51.
95. J.D. Landa Silva, E.K. Burke and S. Petrovic. (2004). An introduction to multi-objective meta-heuristics for scheduling and timetabling. In: X. Gandibleux, M. Sevaux, K. Sorensen and V. Tkindt (eds): Multiple Objective Meta-heuristics. Springer Lecture Notes in economics and mathematical systems, vol. 535. 91-129.
96. D.B. Leake. (1996). *Case Based Reasoning: Experiences, Lessons, and Future Directions.* AAAI Press/MIT Press.
97. A. Lim, A.J. Chin, H.W. Kit and W.C. Oon. (2000). A campus-wide university examination timetabling application. In: Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence, 1020-1025.
98. S.L.M. Lin. (2002). A broker algorithm for timetabling problem. In: E.K. Burke and P. De Causmaecker (eds). (2002). Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling. 21st-23rd August 2002. KaHo St.-Lieven, Gent, Belgium. 372-386.
99. H.R. Lourenco, O. Martin and T. Stutzle. (2003). Iterated local search. In: F. Glover and K. Kochenberher. (2003). *Handbook of Meta-heuristics*, Kluwer. Chapter 11, 321-354. Academic Publishers, Boston, MA, USA.
100. B.G.C. McCollum. (2005). Bridging the Gap Between Research and Practice: University Timetabling in the Real World. KEYNOTE in the Proceedings of the 47th Annual Operational Society Conference (OR47), September 2005, Chester, UK.
101. N.K. Mehta. (1981). The application of a graph coloring method to an examination scheduling problem. *Interfaces*, **11**: 57-64.
102. N.K. Mehta. (1982). A computer-based examination management system. *Journal of Educational Technology Systems.* 11: 185-198.
103. A. Meisels and E. Kaplansky. (2002). Scheduling agents - distributed employee timetabling. In: E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. 167-180.
104. L.T.G. Merlot, N. Boland, B.D. Hughes and P.J. Stuckey. (2003). A hybrid algorithm for the examination timetabling problem. In: E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. 207-231.
105. R. Miles. (1975). Computer timetabling: A bibliography. *British Journal of Educational Technology.* **6**(3): 16-20.
106. N. Mladenovic and P. Hansen. (1997). Variable neighbourhood search. *Computers and Operations Research*, **24**(11): 1097-1100.
107. P. Moscato and C. Cotta. (2003) A Gentle Introduction to Memetic Algorithms. In: F. Glover and K. Kochenberher. (2003). *Handbook of Meta-heuristics*, Kluwer. Chapter 5, 105-144.
108. P. Moscato and M.G. Norman. (1992). A “memetic” approach for the travelling salesman problem - implementation of a computational ecology for combinatorial optimisation on message passing systems. In: Proceedings of the International Conference on Parallel Computing and Transputer Applications. IOS Press (Amsterdam), 177-186.

109. Z. Naji Azimi. (2004). Comparison of metaheuristic algorithms for examination timetabling problem. *Applied Mathematics and Computation*, **16**(1-2): 337-354.
110. Z. Naji Azimi. (2005). Hybrid heuristics for examination timetabling problem. *Applied Mathematics and Computation*. 163(2): 705-733.
111. Krasnogor, N. and Smith, J.E., (2006). A tutorial for competent memetic algorithms: model, taxonomy and design Issues. *IEEE Transactions on Evolutionary Computation*, in press.
112. I.H. Osman and G. Laporte. (1996). Metaheuristics: A bibliography. *Annals of Operational Research*. 63: 513-628.
113. L. Paquete and T. Stutzle. (2002). Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem. In: E.K. Burke and P. De Causmaecker (eds). (2002). Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling. 21st-23rd August 2002. KaHo St.-Lieven, Gent, Belgium. 413-420.
114. L. Paquete and T. Stutzle. (2002). An experimental investigation of iterated local search for coloring graphs. In: S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, and G. Raidl (eds): Applications of Evolutionary Computing - EvoWorkshops 2002. Springer Lecture Notes in Computer Science, vol. 2279, 122-131.
115. S. Petrovic and E.K. Burke. (2004). Chapter 45: University timetabling. In: J. Leung (ed): *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, CRC Press, April 2004.
116. S. Petrovic and Y. Bykov. (2003). A multiobjective optimisation technique for exam timetabling based on trajectories. In: E.K. Burke and P. De Causmaecker (eds). (2003). Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference. Springer Lecture Notes in Computer Science, vol. 2740. 179-192.
117. M. Pirlot. (1996). General local search methods. *European Journal of Operational Research*, **92**: 493-511.
118. R. Qu and E.K. Burke. (2006). Hybridisations within a Graph Based Hyper-heuristic Framework for University Timetabling Problems. Technical Report NOTTCS-TR-2006-1, School of CSiT, University of Nottingham.
119. C.R. Reeves. (ed.): *Modern Heuristic Techniques for Combinatorial Problems.* Oxford: Scientific Publications.
120. L.P. Reis and E. Oliveira. (1999). Constraint logic programming using set variables for solving timetabling problems. 12th International Conference on Applications of Prolog.
121. L.P. Reis and E. Oliveira. (2000). A language for specifying complete timetabling problems. In: E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. 322-341.
122. M.G.C. Resende and C.C. Ribeiro. (2003). Greedy randomized adaptive search procedures. In: F. Glover and K. Kochenberher. (2003). *Handbook of Metaheuristics*, Kluwer. 219-249.
123. B.P. Romero. (1982). Examination scheduling in a large engineering school: a computer assisted participative procedure. *Interfaces*, **12**: 17-23.
124. P. Ross. (2005). Hyper-heuristics. In: E.K. Burke and G. Kendall (eds.), Search Methodologies: Introductory Tutorials in Optimisation and Decision Support Techniques. Ch. 17, 529-556. Springer.
125. P. Ross, D. Corne, H. Terashima-Marin. (1996). The phase transition niche for evolutionary algorithms in timetabling. In: E.K. Burke and P. Ross (eds). (1996).

Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 309-324.

126. P. Ross, E. Hart and D. Corne. (1998). Some observations about GA-based exam timetabling. In: E.K. Burke and M.W. Carter (eds). (1998). Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference. Springer Lecture Notes in Computer Science, vol. 1408. 115-129.
127. P. Ross, E. Hart and D. Corne. (2003). Genetic algorithms and timetabling. In: A. Ghosh and S. Tsutsui (eds.): Advances in Evolutionary Computing: Theory and Applications. Springer-Verlag New York, USA. 755-771.
128. P. Ross, J.G. Marin-Blazquez and E. Hart. (2004). Hyper-heuristics applied to class and exam timetabling problems. In: Proceedings of the 2004 Congress on Evolutionary Computation (CEC2004), 1691-1698.
129. G.C.W. Sabin and G.K. Winter. (1986). The impact of automated timetabling on universities - A case study. *Journal of Operational Research Society*, **37**: 689-693.
130. K. Sastry, D. Goldberg and G. Kendall. (2006). Genetic algorithms. In: E.K. Burke and G. Kendall (eds.) (2005). *Introductory Tutorials in Optimisation, Decision Support and Search Methodology.* ISBN: 0387234608, Springer. Chapter 4, 97-125.
131. A. Schaerf. (1999). A survey of automated timetabling. *Artificial Intelligence Review*, **13**(2): 87-127.
132. A. Schaerf and L. Di Gaspero. (2001). Local search techniques for educational timetabling problems. In: Lenart L., Zadnik Stirn L. and Drobne S. (eds.) Proceedings of the 6 th International Symposium on Operations Research in Slovenia. Slovenia, September 26-28, 2001. 13-23.
133. E.A. Schmidt, T. Strohlein. (1979) Timetable construction - An annotated bibliography. *The Computer Journal*, **23**: 307-316.
134. K. Sheibani. (2002). An evolutionary approach for the examination timetabling problems. In: E.K. Burke and P. De Causmaecker (eds). (2002). Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling. 21st-23rd August 2002. KaHo St.-Lieven, Gent, Belgium. 387-396.
135. H. Simonis. (1995). The CHIP system and its applications. In: First International Conference on Principles and Practice of Constraint Programming, Springer Lecture Notes in Computer Science, vol. 976, 643-646.
136. H. Terashima-Marin, P. Ross and M. Valenzuela-Rendon. (1999). Clique-based crossover for solving the timetabling problem with GAs. In: Schoenauer M. et al (eds.): Proceedings of CEC 99 Conference. Washington: IEEE Press. 1200-1206.
137. H. Terashima-Marin, P. Ross and M. Valenzuela-Rendon. (1999). Application of the hardness theory when solving the timetabling problem with GAs. Proceedings of the Congress on Evolutionary Computation 1999. Washington, D.C. 604-611.
138. H. Terashima-Marin, P. Ross and M. Valenzuela-Rendon. (1999). Evolution of constraint satisfaction strategies in examination timetabling. In: Proceedings of the Genetic and Evolutionary Conference, Orlando, Florida, 635-642.
139. J. Thompson and K. Dowsland. (1996). Variants of simulated annealing for the examination timetabling problem. *Annals of Operational Research*, **63**: 105-128.
140. J. Thompson and K. Dowsland. (1998). A robust simulated annealing based examination timetabling system. *Computer & Operations Research*, **25**, 637-648.
141. E. Tsang, P. Mills and R. Williams. (1999). A computer aided constraint programming system. In: The 1st International Conference on The Practical Application of Constraint Technologies and Logic Programming (PACLP), 81-93.

142. D.J.A. Welsh and M.B. Powell. (1967). The upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*. 11: 41-47.
143. G.M. White. (2000). Constrained satisfaction, not so constrained satisfaction and the timetabling problem. In: E.K. Burke and W. Erben (eds): Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling. 16th-18th August 2000. University of Applied Sciences, Constance, Germany. 32-47.
144. G.M. White and B.S. Xie. (2001). Examination timetables and tabu search with longer-term memory. In: E.K. Burke and W. Erben (eds). (2001). Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference. Springer Lecture Notes in Computer Science, vol. 2079. 85-103.
145. G.M. White, B.S. Xie, and S. Zonjic. (2004). Using tabu search with longer-term memory and relaxation to create examination timetables. *European Journal of Operational Research*, **153**(16): 80-91.
146. T. Wong, P. Cote and P. Gely. (2002). Final exam timetabling: a practical approach. IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 2002). Volume: 2, 726- 731.
147. D.C. Wood. (1968). A system for computing university examination timetables. *The Computer Journal*, **11**(1): 41-47.
148. A. Wren. (1996). Scheduling, timetabling and rostering - A special relationship? In: E.K. Burke and P. Ross (eds). (1996). Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science, vol. 1153. 46-75.
149. Y. Yang and S. Petrovic. (2004). A Novel similarity measure for heuristic selection in examination timetabling. In: E.K. Burke and M. Trick (eds). (2005). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. 377-396.
150. M. Zeleny. (1974). A concept of compromise solutions and method of displaced ideal. *Computers & Operations Researh*. **1**(4), 479-496.

Appendix A. PhD Theses on University Timetabling

- J. Pendlebury. A computer-aided timetabling system for use in FE/HE. PhD Thesis. Department of Operational Research, University of Lancaster, UK. July 1989.
- H. Fang. Genetic Algorithms in Timetabling and Scheduling. PhD Thesis, Division of Informatics, University of Edinburgh, 1994.
- R.F. Weare. Automated Examination Timetabling. PhD Thesis, Department of Computer Science, University of Nottingham, UK, June 1995.
- H. Terashima-Marin. Combinations of GAs and CSP Strategies for Solving the Examination Timetabling Problem. PhD Thesis, Instituto Tecnologico y de Estudios Superiores de Monterrey, August 1998.
- J. P. Newell. Hybrid Methods for Automated Timetabling. PhD Thesis, Department of Computer Science, University of Nottingham, UK, May 1999.
- H. Rudova. Constraint Satisfaction with Preferences. Ph.D. Thesis, Faculty of Informatics, Masaryk University, 2001.

– R. Qu. Case Based Reasoning for Course Timetabling Problems. PhD Thesis, Department of Computer Science, University of Nottingham, UK, August 2002.
– L. Di Gaspero. Local Search Techniques for Scheduling Problems: Algorithms and Software Tools. PhD Thesis, Computer Science, Universit degli Studi di Udine, October 2002.
– R.J. Willemen. School timetable construction Algorithms and complexity. Technische Universiteit Eindhoven, 2002.
– Y. Bykov. Time-Predefined and Trajectory-Based Search: Single and Multiobjective Approaches to Exam Timetabling. PhD Thesis, Department of Computer Science, University of Nottingham, UK, November 2003.
– Y. Yang. Solving Examination Timetabling Problems by Case Based Reasoning. PhD Thesis, Department of Computer Science, University of Nottingham, UK, September 2004.
– L. Merlot. Techniques for Academic Timetabling. PhD Thesis, Department of Mathematics and Statistics and Department of Computer Science and Software Engineering, University of Melbourne, Australia, August 2005.
– N. Hussin. Tabu Search Based Hyper-heuristic Approaches for Examination Timetabling. PhD Thesis, Department of Computer Science, University of Nottingham, UK, November 2005.
– S. Abdullah. Heuristic Approaches for University Timetabling Problems. PhD Thesis, Department of Computer Science, University of Nottingham, UK, July 2006.

Appendix B. Summarising Tables Catergorised by Techniques in Exam Timetabling

In this appendix, the research methods in the exam timetabling literature are categorised in a series of tables by the approaches and techniques they employed. The work from the same authors was grouped in the table to represent the continuous development of a line of investigation. The work in each table (grouped by the same authors) is ordered by the year of publication to represent the development related techniques over the years. In the tables, the term “practical” indicates that the corresponding work was tested/implemented on real world problems. “-” indicates that the corresponding properties were not presented in the paper. “Toronto”, “Nottingham” and “Melbourne” refer to the benchmark problems described in Section 3.

Table 14. Graph Heuristics in the Exam Timetabling

Reference	**Techniques**	**Problems**
Carter et al [51] 1996	largest cliques as the initialisation for graph heuristics with backtracking	Toronto, random
Burke et al [40] 1998	biased and tournament selection that introduce non-determinism to graph heuristics	Toronto
Carter& [48] Johnson 2001	*almost cliques* that are sufficiently dense as the initialisation for graph heuristics	Toronto
Burke& Newall [37] 2004	adaptive heuristic modifier based on Largest Degree, Color Degree and Saturation Degree that dynamically order the exams	Toronto, Nottingham

Table 15. Constraint Based Techniques

Reference	**Techniques**	**Problem**	**Notes**
David [62] 1998	iterative approach, repairing strategies on partial solutions generated by constraint satisfaction model	practical	
Reis&Oliera [120] 1999	constraint satisfaction model, set variables	random, practical	ECLiPSe package
Merlot et al [104] 2003	constraint logic programming as initialisation for Simulated Annealing and hill climbing, labeling: by the size of domain	Toronto, Nottingham Melbourne	OPL language
Duong& Lam [64] 2004	constraint programming with Simulated Annealing, backtracking & forward checking, labeling: by the size of domain, number of students, etc	practical	

Table 16. Tabu Search

Reference	**Moving Strategies**	**Tabu List**	**Problem**	**Notes**
Di Gaspero &Schearf [71] 2001 [70] 2002	exhaust and biased selection on exams causing costs token ring search with 3 neighbourhood structures	adaptive adaptive	Toronto, Nottingham Toronto	greedy initialisation dynamic cost function, employing the EASYLOCAL++ tool
White&Xie [144, 145] 2001&2004	moves on single exams	long and short term memory	practical, Toronto	4-stage Tabu Search, largest enrollment initialisation
Paquete& Stutzle [113] 2002	moves on single exams causing costs	adaptive	Toronto	ordered priorities for constraints

Table 17. Simulated Annealing

Reference	**Moving Strategies**	**Initialisation**	**Problem**	**Notes**
Thompson& Dowsland [140] 1998	Kempe chain	random	practical and derived problems	two-phase approach
Bullnheimer [14] 1998	slot&exam moves	-	practical	adapted model of QAP
Merlot et al [104] 2003	Kempe chain	constraint programming	Toronto, Melbourne, Nottingham	geometric cooling schedule
Duong&Lam [64] 2004	Kempe chain	constraint programming	practical	components set experimentally
Burke et al [16] 2004	moves on single exams	a number of runs of Saturation Degree	Toronto, Nottingham	time-predefined Great Deluge

Table 18. Local Search Based Techniques (beside Tabu Search and Simulated Annealing)

Reference	**Techniques**	**Problem**
Caramia et al [45] 2001	iterative process, the number of timeslots was gradually increased after greedy improvement	Toronto, Nottingham
Ahmadi et al [5] 2003	search on permutations of heuristics and their weights by Variable Neighbourhood Search	extended Nottingham
Casey& Thompson [52] 2003	GRASP: Saturation Degree as initialisation, backtracking, improvement by modified Simulated Annealing with Kempe chain neighbourhood	Toronto
Abdullah et al [3] 2004	Large neighbourhood search with cyclic exchanges of exams among timeslots	Toronto
Qu&Burke [118] 2005	Variable Neighbourhood Search as the high level search upon graph heuristics	Toronto

Table 19. Evolutionary Algorithms

Reference	Operators	Problem	Notes
Ross et al [125] 1996	standard	special graph coloring problems	phase transition, compared with stochastic hill climbing
[126] 1998	standard	special graph coloring problems	issues of direct coding in Genetic Algorithms
Terashima-Marin et al [136] 1999	clique-based crossover	special classes of graph coloring problems	issues of direct coding in Genetic Algorithms
[137] 1999	standard two-point crossover	as above	hardness theory
[138] 1999		Toronto	indirect coding of heuristics rather than actual solutions
Erben [77] 2001	specialised operators	special graph coloring problems	specialised fitness function
Sheibani [134] 2002	standard	derived problems	special mathematical model to estimate costs
Wong et al [146] 2002	mutation with heuristic repairing strategies	practical	modelled as constraint satisfaction problem
Cote et al [61] 2005	local search operators	Toronto, Melbourne, Nottingham	bio-objective constraint satisfaction problems
Burke et al [38] 1996	light and heavy mutations	Toronto, Nottingham	Memetic Algorithm with hill climbing, graph heuristics as the initialisation
[39] 1998	as above	Toronto	initial populations with different diversities in Memetic Algorithm
Burke& Newall [35] 1999	as above	Toronto, Nottingham	multi-stage Evolutionary Algorithm initialised by graph heuristics with backtracking

Table 20. Ant Algorithms

Reference	Initialisation	Problems	Notes
Naji Azimi [109] 2004	heuristic method	derived Toronto	local improvement, compared with Simulated Annealing, Tabu Search and Genetic Algorithms
[110] 2005	heuristic method	as above	hybridisations with Tabu Search
Dowsland& Thompson [74] 2005	recursive Largest Degree and Saturation Degree	Toronto	modified fitness function based on [60], trail calculation, parameter settings

Table 21. Multi-Criteria Techniques

Reference	Techniques	Problems	Notes
Burke et al [15] 2001	hill climbing and heavy mutation on initial solutions by Saturation Degree	Nottingham	deal with 9 criteria
Petrovic &Bykov [116] 2003	search by Great Deluge towards the ideal point guided by the predefined trajectory in the criteria space	Nottingham Toronto	random initial solutions, dynamic weights

Table 22. Hyper-heuristics

	High level heuristic	**Low level heuristic**	**Problem**	**Notes**
Terashima -Marin [138] 1999	Evolutionary Algorithms	solution construction strategies	Toronto	study of in-direct codings
Ahmadi et al [5] 2003	Variable Neighbourhood Search	constructive heuristics &weights	extended Nottingham	perturbations of low level heuristics and their weights in heuristic space
Petrovic &Yang [149] 2005	Case based reasoning	graph heuristics	Toronto	reuse graph heuristics as initialisation methods
Asmuni et al [8] 2004	fuzzy techniques	3 graph heuristics	Toronto	different fuzzy functions
Kendall& Hussin [90] 2004	Tabu Search	constructive, moving strategies	practical	hybridise Tabu Search with hill climbing and Great Deluge
Ross et al [138] 2004	steady state Genetic Algorithm	exams and timeslots picking heuristics	Toronto, course timetabling problems	simplified problem-state descriptions mapped to constructive heuristics, fitness functions
Burke et al [42] 2005	Case based reasoning	graph heuristics	Toronto, course timetabling	heuristics mapped and reused to problem solving situations
[20] 2005	Case based reasoning	as above	Toronto	hybridise graph heuristics by different methodologies
Burke et al [34] 2006	Tabu Search	graph heuristics	Toronto	different number of low level graph heursitics, multi-stage approach
Qu&Burke [118] 2005	Iterated Local Search, Variable Neighbourhood Search, etc	as above	Toronto	different neighbourhood and high level heuristics in the graph based hyper-heuristic

Table 23. Decomposition Approaches

Reference	**Techniques**	**Problem**	**Notes**
Burke&Newall [35] 1999	sequential methods to partition the problems	Toronto Nottingham	sub-problems solved by Memetic Algorithms
Lin [98] 2002	multi-agent algorithm	Toronto random	aggregate schedules from agents and a broker

Table 24. Timetabling Systems

Reference	**Techniques**	**Problem**	**Notes**
Hansen& Vidal [84] [85] 1995	4-phase process with different objectives	Danish high schools	centralised planning on oral and written exams, GIA-system
Colijn& Layfield [55] [56] 1995	multi-stage process for different objectives interactive interface for unforseen circumstances	practical practical	matrices used to record the violations
Lim et al [97] 2000 Ho et al [88] 2002	constructive heuristic with 3 measures, arc consistency algorithm Tabu Search with Push Forward Insertion heuristic	National Uni of Singapore as above	exam and course timetabling, 3-tier client/server application, UTTS system maximise exam paper spreading
Dimopoulou &Miliotis [69] 2001	Integer Programming, grouped courses assigned to grouped timeslots	Athens Uni of Economic & Business	initial exam timetables based on course timetables and improved by heuristics

Table 25. Timetabling Languages and Tools

Reference	**Languages/Tools**	**Notes**
Burke et al [30] 1997	similar to the Z specification language	standard data format for easy exchange of results and comparisons in timetabling research
Tsang et al [141] 1999	EaCL	easy specification of constraint satisfaction problems to support building Constraint Satisfaction systems
Reis&Oliera [121] 2000	UniLang using a list of synonyms	timetabling problem specification in the ECLiPse package
Schearf&Di Gaspero [132] 2001	EASYLOCAL++	object-oriented software tool to support the development of local search algorithms
De Causmaecker et al [63] 2002	Semantic Web, XML	machine accessible way of easy identification for timetabling problems