Recent Developments in Practical Course Timetabling

Michael W. Carter[1] and Gilbert Laporte[2]

[1]Department of Mechanical and Industrial Engineering
University of Toronto
5 King's College Road
Toronto, Ontario M5S 3G8
carter@mie.utoronto.ca

[2]GERAD
École des Hautes Études Commerciales de Montréal
3000 chemin de la Côte-Sainte-Catherine
Montréal, Canada H3T 1V6
gilbert@crt.UMontreal.ca

Abstract. Course timetabling is a multi-dimensional NP-Complete problem that has generated hundreds of papers and thousands of students have attempted to solve it for their own school. In this paper, we describe the major components of the course timetabling problem. We discuss some of the primary types of algorithms that have been applied to these problems. We provide a series of tables listing papers in refereed journals that have either implemented a solution or tested their algorithm on real data. We made no attempt to provide a qualitative comparison. We restricted our presentation to a description of the types of technique used and the size of problem solved We have not included commercial software vendors

1 Introduction

Course timetabling is an important problem encountered in virtually every high school, college and university throughout the world. Over the past thirty years or so, several algorithms have been proposed for the solution of this problem. Our purpose is to report on some recent developments and trends in this area with an emphasis in work that has actually been implemented. More specifically, as in Carter and Laporte [10], we will concentrate on articles published since 1980 and on those that report on real implementations or on tests performed on real data. There are a number of timetabling systems in practice and in the literature that use the computer as a decision support tool for a primarily manual process. We have restricted our survey to include only those papers where the computer is actually used to attempt to solve the timetabling problem automatically.

This survey is organized as follows. In Section 2, we describe the various components of the course timetabling problem. This is followed in Section 3 by a description of the main algorithms used in this field. In Section 4, we summarize in table form the main algorithms that have either been implemented, or tested on real data. The conclusions follow on Section 5.

2 Problem Components

Course timetabling can be viewed as a multi-dimensional assignment problem in which students, teachers (or faculty members) are assigned to courses, course sections or classes; "events" (individual meetings between students and teachers) are assigned to classrooms and times. Problem definition and terminology varies from one institution to the next but the following concepts are common.

a) A *course* corresponds to a subject taught one or more times a week during part of a year. The same course may be taught in one or multiple sections, meaning that it can be repeated by different teachers during the week.

b) A *class* is a group of students taking an identical set of courses and typically remaining together throughout the week.

c) A *program* consists of a set of required courses and a set of elective courses to be taken by students wishing to obtain a given degree.

Some of these concepts do not apply to some institutions. In particular, high schools often work with predefined classes and have few programs so that the main problem in this context is to determine teachers for given subjects and classes, and to construct appropriate teaching schedules. The time frame is typically limited and teachers have heavy teaching loads. The scheduling problem is therefore very tight. In universities, individual students have more choice and faculty members may only teach a few hours a week. The main difficulty in this case is to ensure that students can feasibly select courses required by their program. We summarize in Table 1, the major differences between the high school and the university scheduling problems. Clearly, there are high schools with a variety of electives, and universities with structured problems; but, we present these classical definitions for extreme points on the continuum.

The course scheduling problem is normally decomposed into five different subproblems, not all of whom may be relevant to a particular situation:

Table 1. Characteristics of 'High School' vs 'University' Course Timetabling

Characteristic	High School	University
Scheduling	- by classes	- by student
Choice	- few choices - highly structured programs	- many electives - loosely structured programs
Teacher Availability	- tight (heavy teaching load)	- flexible (light teaching load)
Rooms	- few rooms - same size - central location	- many rooms - variety of sizes - decentralized
Student Load	- very tight (busy all day)	- fairly loose - days and evenings
Criteria	- no conflicts	- minimum conflicts

2.1 Course Timetabling

Assign courses or course sections to time periods. Several side constraints may be present such as respecting program structures as well as room and teacher availabilities.

In some instances, courses or course sections must be spread in particular ways throughout the week. For example, some institutions require that all sections of a same course be scheduled at the same times. A course may be broken into two or three periods and these must be scheduled on different days, etc. Some schedules include lunch periods, time to travel between different buildings, and constraints may be imposed on the number of teaching hours a student or teacher can have in a given day.

We also distinguish course timetabling problems as *Master Timetables* or *Demand Driven* systems. The primary difference between these systems lies in the sequence in which the various subproblems are solved.

Master Timetabling

1. Determine number of sections
2. Assign times (and rooms) to sections
3. Students choose courses from published timetables
4. Students assigned to sections (student scheduling)
5. Drop/add "negotiation" phase

Demand-Driven

1. Students select courses from list
2. Determine number of sections
3. Assign teachers to sections
4. Assign times (and rooms) to sections
5. Student scheduling (assign students to sections)
6. Drop/add

2.2 Class-Teacher Timetabling

This problem arises mostly in high schools. Here, the scheduling unit is a class, not a student. It is normally assumed that the assignment of teachers to courses and classes has already been made. The problem is then to schedule class-teacher meetings without creating conflicts and while satisfying some side constraints on the spread and sequencing of courses. Feasibility is often a major difficulty there and it is common to reconsider previously made class-teacher assignments in order to reach a feasible solution. It is interesting to observe, however, that the class-teacher timetabling problem, without side constraints, can be solved in polynomial time by means of a network flow algorithm (de Werra [20]).

2.3 Student Scheduling

This problem occurs when courses are taught in multiple sections. Once students have selected their courses, they must be assigned to sections. The aim is to provide good quality (conflict free) schedules for students while balancing section sizes and respecting room capacities.

2.4 Teacher Assignment

The problem here is to assign teachers to courses while maximizing a preference function. Again, this problem without side constraints can be solved as a network flow problem (Dryer and Mulvey [22]).

2.5 Classroom Assignment

Events must be assigned to specific rooms to satisfy size, location and facility preferences and restrictions. This is normally performed after the individual events have been assigned to times. Ideally, times and rooms should be assigned simultaneously, but many schools separate the two processes for simplicity.

3 Algorithms

The algorithms used to solve the various components of the course timetabling problem can broadly be classified into global algorithms, constructive heuristics and improvement heuristics. In addition, several methods include user interaction with a computer system.

3.1 Global Algorithms

Small size problems can sometimes be solved directly by means of a standard integer linear programming (ILP) package. Side constraints can often be incorporated into the ILP formulation. Several decomposition techniques can be used for problems that are too large for optimizing. Examples are provided by Glassey and Mizrach [26], and Carter [7]. In the Gosselin and Truchon [27] method for the classroom assignment problem, an ILP is solved repeatedly with different cost coefficients to account for various preference values. In most ILP's, 0-1 variables correspond to assignment decisions. An interesting exception is the McClure and Wells [37] method for the teacher assignment problem in which each variable represents a full teacher schedule and the problem is formulated as a set partitioning problem with side constraints. The number of variables is restricted in order to reduce problem size.

In a similar vein, network flow formulations and algorithms are sometimes used to solve some simple assignment problems without side constraints. Examples are provided in the articles of Osterman and de Werra [42] and Chahal and de Werra [11] for class-teacher assignment, and of Dyer and Mulrey [22] and Dinkel, Mote and Venkataramanan [21] for teacher assignment.

Preferences can be handled by goal programming, as in the work of Miyaji, Ohno and Mine [38] for student scheduling and of Schniderjans and Kim [54] for teacher assignment. Alternatively, an assignment problem can be solved repeatedly with different weights. Graves, Schrage and Sankaran [28] describe an auction based interactive system for student scheduling that uses this idea.

3.2 Constructive Heuristics

The simplest heuristic for constrained assignment problems probably consists of making sequential assignments while preserving feasibility, until this is no longer possible. Then, backtracking may be used to undo some of the previous decisions and the assignment process is reapplied. This type of method may not, however, always converge toward a feasible solution and care must be taken to avoid cycling. This is essentially the method used by Carter, Laporte and Lee [9] for examination timetabling. In the field of course timetabling, it has been applied by Papoulias [44], de Gans [19] and by Cooper and Kingston [15] for class-teacher timetabling, by Sabin and Winter [51] for student scheduling, and by Selim [55] for teacher assignment.

Incomplete branch and bound has been used by Cooper and Kingston [15] for class-teacher timetabling, and by Laporte and Desroches [36] for student scheduling.

More sophisticated enumerative algorithms are based on constraint logic programming (CLP) or similar methods. Here the problems are modeled using variables with finite domains. As variables are fixed, the domains of the free variables are restricted. This technique has been gaining popularity in recent years, as witnessed by the work of Azevedo and Barahona [5], Cheng, Kang, Leung and White [13], Ram and Scogings [46], Guéret, Jusien, Boizumault and Prins [29], Lajos [35], and Henz and Wurtz [30] for course timetabling, and of Fahrion and Dollansky [25] for teacher assignment.

3.3 Improvement Heuristics

Once a feasible solution has been obtained, it can be improved by means of a standard exchange scheme as in the work of Aubin and Ferland [4] and Sampson, Freeland and Weiss [52] for course timetabling. In classical improvement procedures, the search only proceeds from a solution to a better solution. However, with modern search methods such as simulated annealing (SA) and tabu search (TS), the search may move towards a worse solution. Both methods consider at iteration t a solution x_t and its neighbourhood $N(x_t)$. A neighbour of x_t is any solution derived from x_t by applying a given operation (such as switching two elements, etc.) In SA, a candidate solution x is randomly selected in $N(x_t)$. If x improves upon x_t , then x_{t+1} , is set equal to x. Otherwise, x_{t+1} is set equal to x with probability P_t and to x_t with probability $(1\text{-}P_t)$, where P_t is a decreasing function of t. In TS, some solutions are declared tabu during the search process. At iteration t, x_{t+1} is set equal to the best non-tabu solution of $N(x_t)$ and x_t is then declared tabu for a number of iterations.

Evolutionary or genetic algorithms (GA) constitute another popular search strategy. Here a population of solutions is maintained. At every iteration, two parent solutions are selected to produce two offspring and two elements of the population are

discarded. The idea is to produce better and better offspring by selecting some of the best parents from the population. Several variants of these very basic SA, TS and GA algorithms have been produced over the years for a wide variety of combinational optimization problems. Interested readers can consult the books of Reeves [48], of Aarts and Lenstra [3] and the bibliography compiled by Osman and Laporte [41].

In the field of course timetabling, SA has been applied by Abramson [1] for class-teacher timetabling and by Davis and Ritter [18] for student scheduling. TS algorithms have been implemented by Hertz [31] for course timetabling, by Schaerf [53] and by Costa [17] class-teacher timetabling. A large number of GA algorithms have proposed. These include the work of Corne and Ross [16], Paechter, Cumming and Luchian [43], Rich [49] and Erben and Keppler [24] for course timetabling, and of Colorni, Dorigo and Maniezzo [14] for class-teacher timetabling.

3.4 Interactive Systems

In addition to the algorithmic strategies just described, several researchers propose interactive procedures to produce families of solutions under different constraints or preference weights. Chahal and de Werra [11] propose such a system for class-teacher timetabling and Graves, Schrage and Sankaran [28] use an interactive auctioning algorithm for student scheduling.

4 Summary Tables

We have summarized our findings in five tables. Table 2 lists papers related to Course Timetabling, Table 3 describes Class-teacher Timetabling, Table 4, 5 and 6 list the results for Student Scheduling, Teacher Assignment and Classroom Assignment respectively. For each problem, we list the references chronologically. In the second column, we provide the name of the institution and the size (where available) in terms of number of students, courses, faculty, sections, rooms and classes. The third column provides a brief description of the problem that the authors solved in their version. Column 4 describes the algorithm that was used, and column 5 states whether or not the method was implemented or simply tested on real data.

Table 2. Course Timetabling

Reference	Institution (students, courses, faculty, sections, rooms)	Problem Description	Algorithm	Results
Aubin & Ferland [4]	A large Montreal High School and two University departments	- course timetabling and sectioning - minimize student conflicts subject to lecturer and room availability - quadratic assignment	- exchange heuristic (hill climbing)	Implemented
Hertz [31]	University of Geneva Faculty of Ecomonics (1729,288,143,-,67)	- minimum faculty and student conflicts	- Tabu Search (TATI)	Tested on real data
Kang, van Schoenberg & White [34] Cheng, Kang, Leung & White [13]	University of Ottawa (-,2147,-,-,167)	- students grouped in programs with required and elective courses - assign courses to times, rooms, teachers	- constraint logic programming	Tested on real data
Azevedo & Barahona [5]	New University of Lisbon Computer Science Department (-,-,-,-,10)	- courses to times subject to room and teacher constraints	- expert system - constraint logic programming	Tested on real data
Ross, Corne & Fang [50] Corne & Ross [16]	University of Edinburgh, Department of Artificial Intelligence (-, 82, -,-,-)	- minimize lecturer conflict - minimize expected student conflict (lecturers in same theme) - room assignments	-evolutionary algorithm	Tested on real data
Paechter Cumming & Luchian [43] (Rankin [47])	Napier University Computer Science (-,238,52,-,7)	- assign classes to timeslot subject to room, teacher and student group availability	-evolutionary algorithm (genetic)	Implemented

Table 2. Course Timetabling *Continued*

Reference	Institution (students, courses, faculty, sections, rooms)	Problem Description	Algorithm	Results
Sampson, Freeland & Weiss [52]	University of Virginia, Darden Graduate School of Business (230,64,-,89,-)	- determine times for course sections - assign to term - minimize student conflicts - satisfy teacher preferences - sectioning	local search: assign students one-at-a-time	Implemented
Erben & Keppler [24]	Unspecified University	- schedule courses and assign them to rooms subject to several side constraints	- genetic algorithm	Tested on data from a real context
Guéret, Jussien, Boizumault & Prins [29]	Institute of Applied Mathematics, France (160,91,42,-,8)	- schedule courses and assign them to rooms, subject to side constraints on rooms, teachers availabilities, student schedules, course interactions, etc.	- constraint logic programming	Tested in the 1993-1994 data
Henz & Würtz [30]	Catholic College for Social Work, Saarbrücken, Germany (-,91,34,-,7)	- schedule courses and assign them to rooms subject to thirteen classes of side constraints	- constraint logic programming	Tested on real data
Lajos [35]	University of Leeds (- ,1000,-,2500,-)	- schedule courses and assign them to rooms, subject to seven classes of side constraints	- constraint logic programming	Leeds is looking at a follow-up project
Ram & Scogings [46]	University of Natal, South Africa (-,450,-,-,80)	- schedule courses and assign them to rooms, subject to side constraints on rooms	- multiple constraint reasoning	3 variations tested on real data
Rich [49]	A University (-,101,56,-,55)	- schedule courses and assign them to rooms	- genetic algorithm	Tested on real data
Elmohamed, Coddington & Fox [23]	Syracuse University (13653, 3839, -, 3839, 509)	- schedule courses and assign them to rooms subject to several side constraints	- simulated annealing	Tested on real data
White & Zhang [56]	Some U. of Ottawa depts (-, 262, -, -, 46)	- schedule courses and assign them to rooms subject to several side constraints	- Constraint logic and tabu search	Tested on real data

Table 3. Class-Teacher Timetabling

Reference	Institution (classes, courses, faculty, rooms)	Problem Description	Algorithm	Results
Papoulias [44]	Five British Schools (-,-,-,-)	- schedule class-teacher meetings with side constraints	- assignment technique with backtracking	Tested on real data
de Gams [19]	21 H.S. in the Netherlands (-,-,-,-)	- schedule class-teacher meetings with side constraints	- random assignments with some backtracking	Tested on real data
Ostermann & de Werra [42]	H.S.'s in Canton de Vand, Switzerland (40,1350,95,-)	- schedule class-teacher meetings	-network flow algorithm to create a partial schedule; completed by hand.	Implemented in several schools
Chahal & de Werra [11]	An Adult Training School in Geneva (-,-,-,-)	- schedule class-teacher meetings	- network flow to create partial schedule plus facilities to produce alternative schedules under different constraints	Implemented
Abramson [1]	An Australian H.S. (15 to 101, 100 to 757,15 to 37, 15 to 24)	- schedule class-teacher meetings with side constraints	- simulated annealing	Tested on four data sets
Colorni, Dorigo & Maniezzo [14]	A H.S. in Milan	- schedule class-teacher meetings with side constraints	- genetic algorithm	Tested on one data set. No details
Cooper & Kingston [15]	A H.S. in Sidney, Australia	- schedule class-teacher meetings with side constraints	- a variety of techniques are described including incomplete branch and bound, and assignment techniques with backtracking	Tested on real data set. No details
Costa [17]	A H.S. in Porrentrug, Switz. (32,780,65,12 types)	- schedule class-teacher meetings with side constraints	- tabu search	Tested on real data
Schaerf [53]	Guinto O. Flacco H.S., Potenza, Italy (38,27 to 29 lectures per class, -,-)	- schedule class-teacher meetings with some side constraints	- tabu search	Implemented
Chan, Lau & Sheung [12]	Several primary &H.S.'s (-, 1000, -, -)	- schedule class, teacher and rooms	- constraint satisfaction	Implemented
Nepal, Melville, Ally [39]	ML Sultan Technicon Faculty of Commerce (35,35,31,21)	- schedule class, teacher and rooms	- brute force heuristics	Tested on real data

Table 4. Student Scheduling

Reference	Institution (students, courses, sections)	Problem Description	Algorithm	Results
Laporte & Desroches [36]	École Polytechnique de Montréal (EPM) (2799,326,518)	Timetable is given. Assign students to course sections. Maximize student satisfaction. Balance section enrollments. Respect room capacities.	Incomplete branch and bound followed by two improvement phases.	Used by EPM starting in 1984
Sabin & Winter [51]	Memorial University, Newfoundland (-,-,-)	Timetable is given. Assign students to course sections. Maximize student satisfaction. Balance sections enrollments. Respect room capacities.	Construct potential student timetables by giving weights to course sections. Schedule students in a greedy fashion starting with those with the most awkward timetables. Feasibility always maintained in practice.	Implemented at Memorial University
Davis & Ritter [18]	Harvard University (118,-,-)	Assign students to sections while respecting preferences.	Simulated annealing.	Tested on real data
Miyaji, Ohno & Mine [38]	Okayama University, Japan (40 to 83, 13,-)	Assign students to sections.	Goal programming.	Tested on 41 real examples
Graves, Schrage & Sankaran [28]	University of Chicago Graduate School of Business (GSB) (>2000,-,475)	Allocate course sections to students while respecting student preferences. Respect section capacities. Allows for preference-based changes of registration during the term of registration.	Interactive system. Students use bidding points to bid on desired course schedules. Drop, add and swap operations are possible.	Implemented at the GSB since 1981

Table 5: Teacher Assignment

Reference	Institution (courses, faculty)	Problem Description	Algorithm	Results
Dyer & Mulvey [22]	UCLA, Graduate School of Management (-,-)	Assign faculty members to courses. Respect availabilities and preferences.	Network optimization model and algorithm.	Implemented since 1974
Bloomfield & McSharry [6]	Oregon State University School of Business (-)	Assign teachers to classes or section. Assign days, times, rooms to sections. Criteria: teacher preferences and seniority.	Exchange heuristics.	Implemented
Selim [55]	Unspecified	Assign faculty members to courses.	Assignment technique with backtracking.	Tested on real data
McClure & Wells [37]	Department of Decision Sciences, Miami University, Ohio (-,18)	Assign faculty members to courses. Respect faculty preferences.	Integer linear programming with variables representing full schedules. The number of potential schedules is curtailed.	Tested on real data
Schniederjans & Kim [54]	An unnamed department at the University of Nebraska	Assign faculty members to courses. Respect faculty preferences.	Goal programming.	Tested in real data
Dinkel, Mote & Venkataramanan [21]	Texas A&M; College of Business Administration (7,000 ,300)	Assigning teachers to courses, to classrooms. Criteria: faculty preference.	Network flow (initial) Plus integer programming.	Implemented (1983-85)
Fahrion & Dollansky [25]	Department of Economics, University of Heidelberg, Germany (119,43)	Assign faculty members to courses.	Constraint logic programming.	Tested on two real data bases

Table 6. Classroom Assignment

Reference	Institution (students, courses, faculty, sections, rooms)	Problem Description	Algorithm	Results
Glassey & Mizrach [26]	University of California at Berkeley (4000,250)	Assign classes to classrooms.	6-1 linear program solved by a decomposition heuristic.	Implemented since 1985
Gosselin & Truchon [27]	Université Laval, Canada (229 to 267,57)	Assign classes to classrooms. Preferences can be expressed for some assignments.	Integer linear program solved repeatedly with different preference values. Rooms or classes can be aggregated to reduce problem size.	Tested on real data
Carter [7]	University of Waterloo (1400,120)	Assign classes to classrooms.	Integer programming with Lagrangian relaxation.	Implemented since 1986

5. Conclusion

We were somewhat surprised to discover that there are very few course timetabling papers that actually report that the methods have been implemented and used at an institution. Specifically, we found only three implementation for course timetabling, four for class teacher, three for student scheduling, three for teacher assignment and two for classroom assignment.

There were several implementations during the 1960's and 1970's using fairly simple exchange heuristics followed by a relatively quiet period in terms of research activity.

During the past ten years, there has been considerable activity and we expect to see a number of new implementations in the near future.

Acknowledgements

This work was partly supported by the Canadian Natural Sciences and Engineering Research Council under grants OGP0001359 and OGP0039782. This support is gratefully acknowledged.

References

1 Abramson, D., "Constructing School Timetables Using Simulated Annealing: Sequential and Parallel Algorithms", Man. Sci. 37, No. 1, pp. 98-113., 1991.
2 Adamidis, P. and Arapakis, P., "Weekly Lecture Timetabling with Genetic Algorithms", PATAT '97.
3 Aarts, E. and Lenstra, J.K., Local Search in Combinatorial Optimization, Wiley, Chichester, 1997.
4 Aubin, J. and Ferland, J.A., "A Large Scale Timetabling Problem", Computers & Operations Research 16, No. 1, pp. 67-77, 1989.
5 Azevedo, F. and Barahona, P., "Timetabling in Constraint Logic Programming", Proceedings of World Congress on Expert Systems'94.
6 Bloomfield, S.D. and McSharry, M.M., "Preferential Course Scheduling", Interfaces 9, No. 4, pp. 24-31, August 1979.
7 Carter, M.W., "A Lagrangian Relaxation Approach to the Classroom Assignment Problem", INFOR 27, No. 2, pp. 230-246, 1989.
8 Carter, M.W. and Tovey, C.A., "When Is the Classroom Assignment Problem Hard?", Oper. Res. 40, Supp. No. 1, pp. S28-S39, 1992.
9 Carter, M.W., Laporte, G. and Lee, S.Y., "Examination Timetabling: Algorithmic Strategies and Applications", J. of Operational Research Society 47, No. 3, 373-383, March 1996.

10 Carter, M.W. and Laporte, G., "Recent Developments in Practical Examination Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.

11 Chahal, N. and de Werra, D., "An Interactive System for Constructing Timetables on a PC", EJOR 40, pp. 32-37, 1989.

12 Chan, H.W., Lau, C.K., and Sheung, J., "Practical School Timetabling: A Hybrid Approach Using Solution Synthesis and Iterative Repair", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science, Burke & Carter, eds., 1998.

13 Cheng,C., Kang,L., Leung,N. and White, G.M., "Investigations of a Constraint Logic Programming Approach to University Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.

14 Colorni A., Dorigo M. and Maniezzo V., "Genetic Algorithms - A New Approach to the Timetable Problem", Lecture Notes in Computer Science - NATO ASI Series, Vol. F 82, Combinatorial Optimization, (Akgul et al eds), Springer-Verlag, pp 235-239. 1990. (1992?)

15 Cooper, T.B. and Kingston, J.H., "The Solution of Real Instances of the Timetabling Problem", The Computer Journal, vol 36, no 7, pp 645-653, 1993.

16 Corne, D., and Ross, P., "Peckish Initialization Strategies for Evolutionary Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.

17 Costa, D., "A Tabu Search Algorithm for Computing an Operational Timetable", European J. of Operational Res. 76, pp. 98-110, 1994.

18 Davis, L. and Ritter, L., "Schedule Optimization with Probabilistic Search", Proceedings of the 3rd IEEE Conference on Artificial Intelligence Applications, Orlando, Florida, USA, pp. 231-236, IEEE, 1987.

19 de Gans, O.B., "A Computer Timetabling System for Secondary Schools in The Netherlands", EJOR 7, 175-182, 1981.

20 de Werra, D., "Construction of School Timetables by Flow Methods", INFOR 9, No.1, 1971, 12-22.

21 Dinkel, J.J., Mote, J. and Venkataramanan, M.A., "An Efficient Decision Support System for Academic Course Scheduling", Operations Research 37, No. 6, pp. 853-864, 1989.

22 Dyer, J.S. and Mulvey, J.M., "Computerized Scheduling and Planning", New Directions for Institutional Research 13, pp. 67-86, 1977.

23 Elmohamed, S., Coddington, P., and Fox, G., "A Comparison of Annealing Techniques for Academic Course Scheduling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science, Burke & Carter, eds., 1998.

24 Erben,W. and Keppler, J., "A Genetic Algorithm Solving a Weekly Course Timetabling Problem", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.

25 Fahrion, R. and Dollansky, G., "Construction of University Faculty Timetables Using Logic Programming", Discrete Applied Mathematics 35, No. 3, pp. 221-236, 1992.

26 Glassey, C.R. and Mizrach, M., "A Decision Support System for Assigning Classes to Rooms", INTERFACES 16, No. 5, pp. 92-100, 1986.

27 Gosselin, K. and Truchon, M., "Allocation of Classrooms by Linear Programming", J. Operational Research Soc. 37, No. 6, 561-569, 1986.

28 Graves, R.L., Schrage, L. and Sankaran, J., "An Auction Method for Course Registration", INTERFACES 23, No. 5, pp 81-92, September-October 1993.

29 Gueret,G., Jussien,N., Boizumault,P. and Prins, C., "Building University Timetables Using Constraint Logic Programming", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.
30 Henz,M. and Wurtz,J., "Using Oz for College Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.
31 Hertz, A., "Tabu Search for Large Scale Timetabling Problems", EJOR 54, pp. 39-47, 1991.
32 Jaffar, J. and Maher, M.J., "Constraint Logic Programming: A Survey", Journal of Logic Programming 19/20, pp. 503-581, 1994.
33 Kang, L. and White, G.M., "A Logic Approach to the Resolution of Constraints in Timetabling", European Journal of Operational Research 61, pp. 306-317, 1992.
34 Kang, L., Von Schoenberg, G.H. amd White, G.M., 1Complete University Timetabling Using Logic1, Computers and Education, 17 No. 2, pp. 145-153, 1991.
35 Lajos, G., "Complete University Modular Timetabling Using Constraint Logic Programming", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.
36 Laporte, G. and Desroches, S., "The Problem of Assigning Students to Course Sections in a Large Engineering School", Comp. & Ops. Res. 13, No. 4, pp. 387-394, 1986.
37 McClure, R.H. and Wells, C.E., "A Mathematical Programming Model for Faculty Course Assignments", Decision Sciences 15, No. 3, pp. 409-420, 1984.
38 Miyaji, I., Ohno, K. and Mine, H., 1Solution Method for Partitioning Students into Groups1, European Journal of Operational Research, 33, No. 1, pp. 82-90.616, 1981.
39 Nepal, T., Melville, S.W., and Ally, M.I., "A Brute Force and Heuristics Approach to Tertiary Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science, Burke & Carter, eds., 1998.
40 Ng, W.-Y., "TESS: An Interactive Support System for School Timetabling", Information Technology in Educational Management for the Schools of the Future, Fung et al (ed.), Chapman and Hall, pp. 131-137, 1997.
41 Osman, I.H. and Laporte, G., "Metaheuristics: A Bibliography. G. Laporte and I.H. Osman (eds), Metaheuristics in Combinatorial Optimization", Annals of Operations Research 63, pp. 513-623, Baltzer, Amsterdam, 1996.
42 Ostermann, R. and de Werra, D., "Some Experiments with a Timetabling System", OR Spektrum 3, pp. 199-204, 1982.
43 Paechter, B., Cumming, A. and Luchian, H., "The Use of Local Search Suggestion Lists for Improving the Solution of Timetable Problems with Evolutionary Algorithms", In Proceedings of the AISB Workshop on Evolutionary Computing, Sheffield, England, April 3-7, 1995.
44 Papoulias, D.B., "The Assignment-to-days problem in a School Time-table, a Heuristic Approach", EJOR 4, pp. 31-41, 1980.
45 Poutain, D., "Constraint Logic Programming", Byte 20, pp. 159-160, 1995.
46 Ram,V. and Scogings, C., "Automated Time Table Generation Using Multiple Context Reasoning with Truth Maintenance", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.
47 Rankin, R.C., "Automatic Timetabling in Practice", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.
48 Reeves, C.R., 1Modern Heuristic Techniques for Combinatorial Problems1, Blackwell, Oxford, 1993.

49 Rich, D.C., "A Smart Genetic Algorithm for University Timetabling", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science 1153, Burke & Ross, eds., 1996.

50 Ross, P., Corne, D. and Fang, H.-L., "Successful Lecture Timetabling with Evolutionary Algorithms", Dept of A.I. Working Paper, University of Edinburg, 1994.

51 Sabin, G.C.W. and Winter, G.K., "The Impact of Automated Timetabling on Universities - A Case Study", J. Operational Research Soc. 37, No.7, pp. 689-693, 1986.

52 Sampson, S.E., Freeland, J.R. and Weiss, E.N., "Class Scheduling to Maximize Participant Satisfaction", Interfaces 25, No. 3, pp. 30-41, May-June 1995.

53 Schaerf, A., "Tabu Search Techniques for Large High-School Timetabling Problems", Comp. Science/Dept. of Interactive Systems, CS-R9611, Centrum voor Wiskunde en Informatica, SMC, Netherlands Organization for Scientific Researchm 1996.

54 Schniederjans, M.J. and Kim, G.C., "A Goal Programming Model to Optimize Departmental Preference in Course Assignments", Comput. Opns. Res. 14, No. 2, pp. 87-96, 1987.

55 Selim, S.M., "An Algorithm for Constructing a Unversity Faculty Timetable", Comput. Educ. 6, No. 4, pp. 323-334., 1982.

56 White, G.M., and Zhang, J., "Generating Complete University Timetables by Combining Tabu Search with Constraint Logic", in Practice and Theory of Automated Timetabling, Springer-Verlag Lecture Notes in Computer Science, Burke & Carter, eds., 1998.

57 Wright, M., "School Timetabling using Heuristic Search", Journal of the Operational Research Society, Vol. 47, pp. 347-357, 1996.

58 Yoshikawa, M., Kaneko, K., Yamanouchi, T. and Watanabe, N., "A Constraint-Based High School Scheduling System", IEEE Expert, Vol. 11, No. 1, IEEE Coputer Society, Los Alamitos, CA., pp. 63-72, February 1996.