TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.4, April 2014, pp. 2985 ~ 2994
DOI: http://dx.doi.org/10.11591/telkomnika.v12i4.4798

An Intelligent Course Scheduling Model Based on Genetic Algorithm

Guofeng Qin[1], Haibin Ma*[2]
Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
*Corresponding author, e-mail: gfqing@aliyun.com[1], sunny_mhb@163.com[2]

Abstract

With the university expansion, how to maintain teaching order using limited resources make the intelligent course scheduling become a multiple-constraint and multi-objective optimization problem. Traditional intelligent course scheduling algorithm is inefficient, cannot solve curriculum conflict question and meet the requirements of the modern university education management. Given this situation, this paper analyzes the university timetabling problem, and establishes a general course scheduling model; then proposes an improved genetic algorithm to sovle the intelligent course scheduling problem. It can meet all of the education resources' constraints and the teachers' personal demands as much as possible. Test the performance of between the improved genetic algorithm and simple genetic algorithm under different scenarios, the experimental results show that the improved genetic algorithm has better performance, can schedule courses reasonable.

Keywords*: intelligent course scheduling, genetic algorithm, multiple-constraints, multi-objective*

1. Introduction

With the university expansion and the development of education, traditional manual course scheduling and computer-aided course scheduling has already cannot meet the practical needs [1]. Intelligent course scheduling which also called University Timetabling Problem (UTP), is a combinatorial optimization problem that schedule curriculum effectively using limited teacher, classroom and time resources [3]. It involvs many factors and has great complexity. In 1975, Even.S proved that UTP is NP-complete problem, announced the academic status and difficulty of this space-time combinatori problem [4, 9].

The genetic algorithm(GA) is a kind of search algorithm inspired by the process of biological evolution. It can self-adapt to global optimization and often been used to find near-optimal solution of optimization and search problems. It has already widely used to resolve the UTP. But the Simple Genetic Algorithm (SGA) also has its limits: random initialization method and fixed probability of crossover and mutation etc. These limits can easily lead to high collision rate, premature and slow convergence phenomena [5], will affect the result of course scheduling.

In order to improve the efficiency and success rate of GA, combining with intelligent course scheduling problem, this paper improve the genetic algorithm encoding mode, initialization method and crossover and mutation probability. This can speed up the convergence speed, avoid premature phenomenon. Then test the algorithm through experiment. The result show that improved genetic algorithm reduce the collision rate, enhance effciency while satisfy the constraints, and can solve the intelligent course scheduling problem well.

2. Intelligent Course Scheduling Model Based on GA

2.1. Intelligent course scheduling Problem Analysis

Intelligent course scheduling is a multi-dimensional assignment problem in which students, teachers (or faculty menbers) are assigned to courses or classes, events (individual meetings between students and teachers) are assigned to classrooms and times. Because of its large scale and many factors it involvs, it is a complex work. As a problem that must be faced in

Received September 9, 2013; Revised November 13, 2013; Accepted December 1, 2013

the process of teaching in colleges and universities, it is a Time Table Problem researched in operations research, has been deeply studied and is known to be NP problem.

A timetable is a placement of a set of eventgs in time. A event is a combination of resources (eg rooms, people and items of equipment), some of which may be specified by the problem, and some of which must be allocated as part of the solution. Timetabling has long been known to belong to the class of problems called NP-complete, ie no method of solving it in a reasonable amount of time is known [11].

The challenge of time table problem is to schedule events over limited resources so as to avoid collisions and to satisfy a number of side-constraints. The collisions will occur whenever a timetable requires any resource to be in two places at the same time. The timetabling process is made more difficult by the fact that so many people are affected by its outcome.

Timetabling constraints are varied and many. Here are some of the most common types:

1) Resource Assignmen: A resource may be assigned to a resource of a different type, or to a meeting. For example, a lecturer prefers to teach in a particular room.
2) Time Assignment: A event or a resource may be assigned to a time.
3) Time Constraints between Meetings: Common exampleof this class of constraint are that one particular meeting must take place before another one.
4) Event Spread: Events should be spread out in time. For example, no student should have more than one exam in any day.
5) Event Coherence: This constraint is designed to produce more organised and convenient timetables, and often run contrary to “meeting spread” constraints.
6) Room Capacities: The number of students in a room may not exceed the room’s capacity.
7) Continuity: Any constraints whose main purpose is to ensure that certain features of student timetables are constant or predictable.

Constraints shown above can be divided into “hard” and ”soft” categories:

1) Hard constraints: A timetable which breaks a hard constraint is not a feasible solution, and must be repaired or rejected by the timetabling algorithm [6]. Hard constraints include “firstorder conflicts”, ie no person may be required to attend more than one meeting at any time.
2) Soft constraints: Soft constraints are less important than hard constraints, and it is usually impossible to avoid breaking at least some of them. Whichever timetabling method is applied, timetables are usually rated by a penalty function, which calculates the extent to which a timetable has violated its soft constraints. Some soft constraints are more important than others, and this is often specified with a priority value.

2.2. Intelligent Course Scheduling Mathematical Model

Given complexity of intelligent course scheduling problem, there is great potential for computational techniques to help ease the task of university timetabling. Firstly, we establish mathematical model for it.

Denote the number of week of one term as WK, number of classes as CL, the number of courses as CR, the number of teacher as TE, the number of classroom as RM,the number of time silce in one week is TS. So one term has TS*WK time slices, then set POINTS = TS*WK. Assume that class collection of course is $\{cl_1, \quad cl_2, \quad \cdots \quad c\mathrm{l}_n\}$,time collection of course is $\{pt_1, \quad pt_2, \quad \cdots \quad pt_n\}$, and classroom collection allocated for it is $\{rm_1, \quad rm_2, \quad \cdots \quad rm_n\}$.

Array ClassM [CL] [POINTS], TeacherM [TE] [POINTS], RoomM [RM] [POINTS] are used to represent class, teacher and classroom constraint data model. They are used to record the resource allocation result and detect collision. For example Equation (1).

$$ClassM[i][j] = \begin{cases} 1 & \text{class i has been arranged course in point j} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

So when allocate time and classroom resources for course, constraint data model can be used to guarantee the result satisfy constraints of time table problem. For example:

1) Students in the same class can be only arranged one course at the same time. it is a event spread constraint, we use Eq.(2) to make resouce allocation meet it.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} ClassM[cl_i][pt_j] \le 1 \quad (2)$$

2) One teacher can only be arranged one course at the same time, we use Equation (3).

$$\sum_{j=1}^{n} TeacherM[i][pt_j] \le 1 \quad (3)$$

3) One classroom can only be arranged one course at the same time, we use Equation (4).

$$\sum_{j=1}^{n} RoomM[i][pt_j] \le 1 \quad (4)$$

From above, we know class, teacher and classroom constraint data model can be used to detect collision and record data: when allocate time slice and classroom resources for course, then check whether Equation (2)-(4) are satisfied. If so, the resource allocation is correct, then record the result in constraint data model; if not, it shows that resource allocation cause collision and should reallocate resources.

From the mathematical model, it's known that intelligent course scheduling is multi-constraint and multi-objective optimization problem. Because of the constraints, algorithm efficiency will be very low and collision rate will be very high if we use traditional course scheduling algorithm. Genetic algorithm is intelligent heuristic algorithm that simulate the biological mechanisms of biological evolution. It's intelligent, parallelism and robustness while solving combinatorial optimization problem. So we use adaptive genetic algorithm to solve intelligent course scheduling problem.

2.3. Design of GA in Course Scheduling Problem

Since SGA has its drawbacks, encoding and genetic manipulation mode and so on must be correctly defined while solve the intelligent course scheduling problem.

2.3.1. Gene Encoding Mode

Gene is the basic execution unit of the genetic manipulations. Encoding and decoding mode affects the expression of information, design and implementation efficiency of genetic manipulation and the speed of convergence.

SGA use binary encoding mode. It cannot express course scheduling information well because of the complexity of intelligent course scheduling problem. So we use natural encoding mode: each chromosome represent curriculum of one class. Each gene of one chromosome consists of two parts: ID of course and ID of classroom. So the population is one result of course scheduling.The chromosome structure is shown in Figure 1.

courseID roomID	courseID roomID	……	courseID roomID	courseID roomID

Figure 1. Chromosome Structure

2.3.2. Collision Detection

Resource allocation may lead to collision. Constraint data model can be used to detect collision. While allocate resource for every course, we should check whether hard constraints are satisfied. If not, it shows that collision exist and resource should be reallocated. For example, we can use Equation (5)-(7) to check the constraints listed in 2.2.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} ClassM[cl_i][wk_i] = 0 \quad (5)$$

$$\sum_{j=1}^{n} TeacherM[i][wk_j] = 0 \quad (6)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} RoomM[rm_i][wk_j] = 0 \quad (7)$$

2.3.3. Population Initialization

In the process of population initialization, we allocate time and classroom resources for every course and then init the chromosomes using the time and classroom information.

Random initialization mode will cause lots of collisions, so we improve the initialization mode: when initialize the population, detect and avoid the collision to guarantee the Initialization result is feasible. The improved algorithm of population initialization is shown as follows:

```
Begin
        for each course {
            do {
                allocate time slice resource for course;
                detect collision;
            } while (collison exsits);
            do {
                allocate classroom resource for course;
                detect collision;
        } while (collision exsits);
            initialize the chromosomes using resource allocation information;
            record inforamtion in constraint data model to show resources allocated
are not available anymore;
        }
End
```

2.3.4. Fitness Function

In the evolution process of the genetic algorithm, fitness function determines the evolution direction. Therefore the fitness function directly determines the course scheduling optimization speed.

Design of fitness function and fitness values calculation depends on soft constraints. Soft constraints are denoted as $soft_1$,...,$soft_n$. Fitness function is designed as follows:

Total score of every soft constrait is 100.Set weight for every soft constraint, recorded as w_1... w_n, and $\sum_{i=1}^{n} w_i = 1$.

Record the score of $soft_i$ as ScoreOfSoft(i),so score of course j can be calculated using Equation (8).

$$ScoreOfCourse(j) = \sum_{i=1}^{n} ScoreOfSoft(i) * w_i \quad (8)$$

Assume total number of courses of a class in one week is COUNT, so score of this class k record as Equation (9).

$$ScoreOfClass(k) = \left(\sum_{j=0}^{COUNT-1} ScoreOfCourse(j) \right) \Big/ COUNT \quad (9)$$

And then, fitness value of class k Adaptability(k) = $ScoreOfClass(k)/100$.

2.3.5. Genetic Manipulation

Genetic manipulations of the GA mainly contains selection, crossover and mutation.

Selection: we use roulette method, individual with greater fitness value have a greater chance to be selected. In order to calculate the proportion of the individual's fitness value, record the population size as IndividualCount, fitness value of class k is Adaptability(k). So proportion of the individual's fitness value can be calculated using Equation (10).

$$Adaption\,\mathrm{Pr}\,oportion(k) = Adaptability(k) \Big/ \sum_{i=0}^{IndividualCount-1} Adaptability(i) \quad (10)$$

Crossover: this paper take curriculum of one class as one individual, so for a gene, we only exchange the time slice and classrooms information, not the entire gene. The process of crossover is shown as follows:

```
Begin
        find two chromosomes randomly, recorded as C1 and C2;
        choose one gene from C1 and C2 respectively, recorded as R1 and R2;
        if (classrooms' type and capacity are the same)
            exchange classroom of R1 and R2;
        else{
       reallocate classroom for R1 and R2;
       detect collision;
        }
End
```

Mutation: mutation will mutate gene of chromosome. Mutation operation is conducive to increasing diversity of the population.The process of mutation is:

```
Begin
    find one chromosome as C1, and choose one gene as R1;
    define an array temp [] to save the course's ideal time slices and 0-19;
    choose one number from temp [] randomly, record corresponding gene as R2;
    detect collision between R1 and R2;
    if (collision not exsit )
            exchange course and classroom of R1 and R2;
   else
            mutation end;
End
```

Control parameters: control parameters mainly contain crossover probability and mutation probability. Fixed crossover and mutation probability will decrease the diversity of evolution population, lead to local optimal solution [2]. So we use adaptive crossover and mutation probability, shown in Figure 2.

$$\text{Crossover Probality } P_c = \begin{cases} k_1 & f^l \le f_{avg} \\ \dfrac{k_1(f_{\max} - f_{avg})}{f_{\max} - f^l} & f^l > f_{avg} \end{cases}$$

$$\text{Mutation Probality } P_m = \begin{cases} k_2 & f \ge f_{avg} \\ \dfrac{k_2(f_{\max} - f)}{f_{\max} - f_{avg}} & f < f_{avg} \end{cases}$$

Figure 2. Control Parameters

P_c and P_m represents the crossover probability and mutation probability.f^l is the bigger fitness value of the two crossover individuals, f represents the variation of individual fitness value, k_1 and k_2 are random number between (0, 1) [10].

2.3.6. End Condition

It's very important that when to terminate the GA. If GA is terminated prematurely, the algorithm has not been convergence, the results are not optimal; when terminated too late, time and hardware resources will be wasted [8]. Therefore, we set two end conditions of the genetic algorithm:

1) When the average fitness value of population meets the expectation, the algorithm ends;
2) The population evolutionary algebra reach to the largest algebra, the algorithm ends.

2.4. Algorithm Flow of Intelligent Course Scheduling Model

Genetic algorithm simulates the genetic process by selection operation, crossover operation and mutation operation. Individuals are selected based on the fitness value, making individual that have greater fitness value have a greater chance to exist. Evaluating each individual through fitness function realize the "survival of the fittest". After genetic manipulation, the individual collection form the next generation population, then put them into the next round of evolve.

According to genetic manipulations, workflow of intelligent course scheduling model based on GA is as follows:

1) Course scheduling solution initialization:allocate resources for course, and record informations in;
2) Compute fitness value of population using the fitness function;
3) Choose better individuals using roulette method, and generate the next generation;
4) Carry out genetic manipulations to generate new individuals useing the adaptive crossover and mutation probability ;
5) Judge whether algorithm meet the end conditions. If so, we get the global optimal solution, jump to 7); if not, jump to 2);
6) Get information from the population, decoding the information and output the scheduling result;
7) Algorithm ends.

Flow char of of Intelligent Course Scheduling is shown in Figure 3.

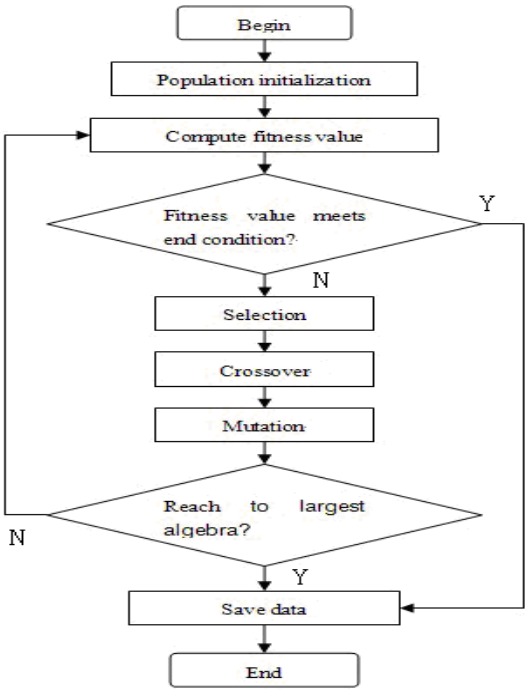

Figure 3. Workflow of Intelligent Course Scheduling Based on xx GA

3. Experient and Analysis

To prove the reliability of the algorithm, we conduct simulation experiment to validate the algorithm. Experiment is divided into three modules: problem specify, course scheduling parameter setting and performance comparison.

3.1. Problem Specify

According to established mathematical model and analysis of time table poblem, while taking into account the different education system, we should specify the problem: classify the course,partition the time and summary the constraints.

3.1.1. Course Classification

Different kind of course has different priority and feature, we cannot take them as the same. In this paper, the course is divided into six levels of the class A-F. The priority of courses increase from class A to class F. Classification of courses is shown in Table 1.

Table 1. Course Classification

Class	Name
A	PE class、experimental course
B	Public basic course
C	Compulsory course
D	Selective course
E	Professional practice course
F	Public elective course

3.1.2. Partition of Time Slice

We divide one day into five time slices. So one week contains twenty five time slices while Saturday and Sunday are not used to arrange course,denoted as S0-S24. So one chromosome contains 25 genes. Time slice partition is shown in Table 2.

Table 2. Time Slice Partition

	Monday	Tuesday	Wednesday	Thursday	Friday
morning	S0	S4	S8	S12	S16
	S1	S5	S9	S13	S17
afternoon	S2	S6	S10	S14	S18
	S3	S7	S11	S15	S19
evening	S20	S21	S22	S23	S24

3.1.3. Constraint Conditions

Because of the many elements and the constraints the time table problem involved, we should arrang the curriculum reasonable to satisfy the constraints.

Constraints contain hard and soft constraint. Hard constraint should be satisfied when allocate time slice and classroom resources for courses. It guarantees the correctness of the result [7]. Hard constraints are shown in Table 3.

Table 3. Hard Constraints

	Hard Constraints
1	The same class can only be arranged one course at the same time
2	The same teacher can only be arranged one course at the same time
3	The same classroom can only be arranged one course at the same time
4	The classroom capacity needs to meet the requirements of the course
5	Type of classroom meet the need of course
6	Class A courses must be arranged in section 3-4 of the morning or afternoon

Soft constraint is to meet humanized requires of different factors, and shold be satisfied as far as possible to make the result more humane. Soft constrain are shown in Table 4.

Table 4. Soft Constraints

	Soft Constraints
1	Different times of one course in one week should be in different days
2	Try to arrange courses in their best time slices
3	Classroom capacity should match with the demand, to avoid resource waste
4	Try to arrange the course in the ideal building
5	Try to arrange the course on the ideal floor
6	Different times of the same course should be arranged in the same classroom

3.2. Course Scheduling Parameter Setting

According to the actual need, parameters of the improved genetic algorithm can be set, as shown in Table 5.

Table 5. Parameter Setting

Parameter	value
Largest algebra	1000
Ideal fitness value	90
Crossover probability	Self-adapting
Mutation probability	Self-adapting

3.3. Performance Comparison

To make the simulation results more convincing, we take comparison experient using improved genetic algorithm and SGA to solve the intelligent course scheduling probelm. Then compare the performance of them in three aspects: only crossover and mutation adaptively; only initialize the population with collision detect and crossover and mutation adaptively and initialize the population with collision detect at the same time.

While only crossover and mutation adaptively in the improved genetic algorithm, fitness value of population comparison between improved genetic algorithm and SGA is shown in Figure 4.

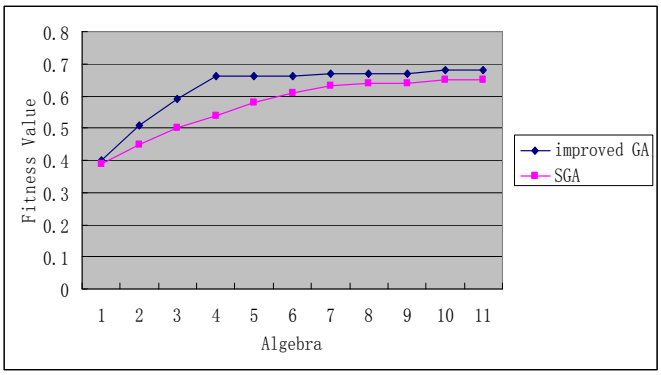

Figure 4. Performance Comparison

While only initialize the population with collision detect in the improved GA, performance comparison between improved GA and SGA is shown in Figure 5.

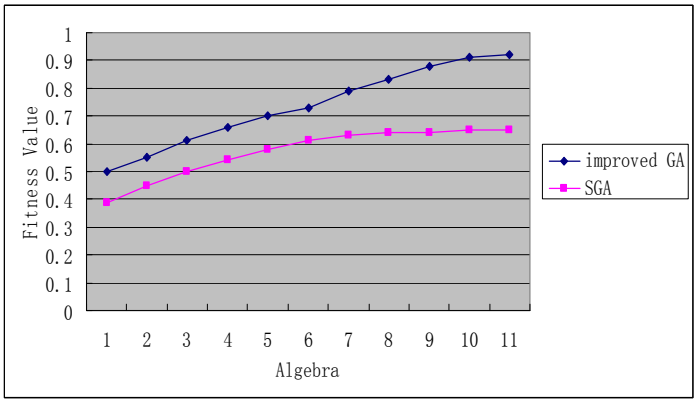

Figure 5. Performance Comparison

From Figure 4 and Figure 5, we find that adaptive crossover and mutation probability can make the genetic algorithm converges faster, and opulation initialization with collision detection will make the result has a better fitness value. Both of them can elevate the performance of the genetic algorithm.

While solving the intelligent course scheduling problem, if the improved genetic algorithm crossover and mutation adaptively and initialize the population with collision detects at the same time, the algorithm will converge faster and get a better result then simple genetic algorithm as shown in Figure 6. Finally we can conclude that the improved genetic algorithm has better performance than simple genetic algorithm while solve the intelligent course scheduling problem.

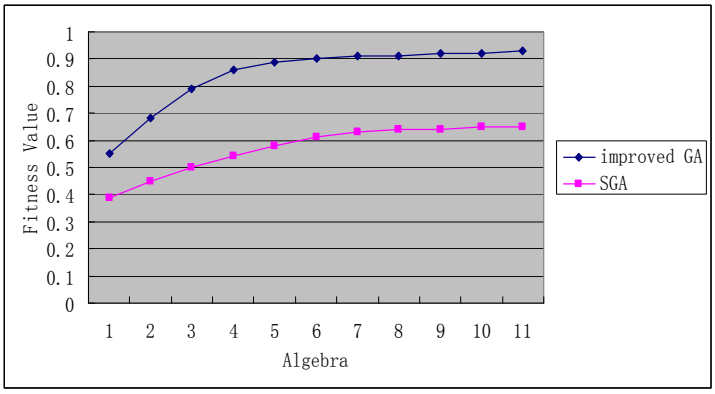

Figure 6. Performance Comparison

4. Conclusion

Intelligent course scheduling is a very important part of university teaching. Traditional course scheduling method is inefficient, has high collision rate and cannot allocate teaching resources reasonably. All of these factors will affect the teaching quality of university.

Analyzing the characteristics of intelligent course scheduling problem and combined with characteristics of genetic algorithm, this paper use genetic algorithm to resolve this problem. Meanwhile improve the SGA aiming at the limits of it, find adaptive genetic manipulation mode and population initialization method with collision detect.

The experiment results show that, the improved genetic algorithm can solve the intelligent course scheduling problem in a certain extent, reduce the workload of academic staff. It can play a very important role in intelligence of the university education management and has a certain popularization value.

References

[1] Zong Wei. Research and Realization of University Timetable System Algorithm. *Computer Simulation.* 2011; 28(12): 389-392.
[2] Liao Yuan, Huang Qin, Gao Peixia. Application of Self-adaptive Genetic Algorithm Based on Three-dimension Coding in Curriculum Scheduling System. *Computer and Modernization*. 2008; (12): 23-28.
[3] Li Yuji, Lu Caiwu, Liu Guan. Application of Ant-colony Genetic Algorithm in Smart Course schedule Systems of Colleges. *Modern Electronics Technique*. 2012; (14): 121-123.
[4] A Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*. 1999; 87-127.
[5] Zhu Jianbing, Li Zhanhuai, Zhao Na. Research on the Problem of Auto-composing Test Paper Based on Hybrid Genetic Algorithm. *Computer Simulation.* 2009; (5): 328-331.
[6] R Lewis, B Paechter. *Application of the Grouping Genetic Algorithm to University Course Timetabling.* In proceedings of the 5 International Conference on Recent Advances in Soft Computing. 2004; 189-194.
[7] Jian Ni, Ning-Ning Yang. Genetic Algorithm and its Application in Scheduling Ststem. *TELKOMNIKA Indonesian Journal of Electrical Engineering.* 2013; 11(4): 1934-1939.
[8] Gotlieb. *The Construction of Class-Teacher Timetables.* Proceeding IFIP Congress, Amsterdam. 1963: 73-74.
[9] S Even, A Itai, A Shamir. On The complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*. 1976; (5): 691-703.
[10] Xia Bing. A New Genetic Algorithm and Its Application in Evaluating Chorus Course. 2013; 11(8): 4517-4522.
[11] EK Burk, KS Jackson, JH Kingston, RF Weare. Automated University Timetabling: The State of the Art. *The Computer Journal.* 1997; 40(9): 565-571.