Nandhini, M.; Kanmani, S.
Design of Mathematical Model and Local Search with Heuristics Optimization of Course Timetabling
International Journal of Combinatorial Optimization Problems and Informatics, vol. 1, núm. 2,
septiembre-diciembre, 2010, pp. 20-30
International Journal of Combinatorial Optimization Problems and Informatics
Morelos, México

Available in: http://www.redalyc.org/articulo.oa?id=265220654004
Design of Mathematical Model and Local Search with Heuristics Optimization of Course Timetabling

M.Nandhini 1, Dr.S.Kanmani 2
1Research Scholar, Research and Development Centre
Bharathiar University,Coimbatore-46, India.
2Prof & Head, Department of Information Technology
Pondicherry Engineering College, Puducherry-14, India.
mnandhini2005@yahoo.com, kanmani@pec.edu

Abstract. The goal of Combinatorial Optimization is finding the best possible solution from the set of feasible solutions. This can be solved using either Artificial Intelligence or Operation Research. Timetabling means scheduling activities to time slots in an order by satisfying hard constraints and soft constraints. Hard constraints should be satisfied but the violations of soft constraints to be minimized. In the paper, the problem of Course Timetabling of Department of Information Technology in Pondicherry Engineering College is studied extensively and its all hard and soft constraints are clearly represented in mathematical model. Also, the problem has been implemented with Local Search Steepest-Ascent Hill Climbing algorithm with heuristics. The results are tabulated and analyzed with different parameters, like generation, number of combinations.

Keywords: constraints, heuristics, states space, optimal solution, course timetabling problem (CTP), mathematical model, steepest ascent hill climbing.

1 Introduction

Timetabling problem [1] is concerned with the assignment of lectures in the specific timeslots and rooms. The course timetabling problem is a typical scheduling problem that appears to be a tedious job in every academic institute once or twice a year. To solve this hardly constrained NP-hard combinatorial problem with exhaustive and heuristic searches are needed. The problem involves the scheduling of classes, students, teachers and rooms at a fixed number of time-slots, subject to a certain number of constraints. Timetables satisfying these hard constraints only form the search space to get the optimal solution. The quality of the solution is measured in terms of a penalty value which gives the degree of optimization and is depending on the satisfaction of soft constraints.

In this paper, the problem of Course Timetabling of Department of Information Technology in Pondicherry Engineering College is studied extensively and its all hard and soft constraints are clearly represented using mathematical representation and the optimization of the problem is done by local search with heuristics. Presently it is prepared manually, by trial and hit method.

Our earlier work, focused on the optimization of Class Timetable of Undergraduate Science courses of Pondicherry University [2]. As an extension, in this work, optimization of course timetabling is done and the results of different generations with number of combinations (states) are tabulated and analyzed. While searching states space, satisfaction of hard constraints and reliability of soft constraints by the combination is guiding the search process.

The following section describes course timetabling problem and presents the proposed mathematical model for representing the constraints. This is followed by the description about various approaches and techniques available to solve all type of timetabling problems in Section 3. In Section 4, implementation of course timetabling is described. In Section 5, Experimental results are tabulated and analyzed against various parameters and is followed by Conclusion in Section 6.

Received June 23, 2010 / Accepted August 18, 2010
2 Course Timetabling Problem

The problem involves assigning lecture activities to timeslots in the Lecture hall and Laboratories subject to laborious hard and soft constraints. Hard Constraints must not be violated but the violation of soft constraints should be reduced. A timetable which satisfies the hard constraints is known as a feasible solution. The following section discusses the timetabling of the Bachelor of Technology Course offered in the department of Information Technology, Pondicherry Engineering College.

2.1 Problem Description

The Course contains 4 classes (each for a year of study). The framework of each course in the Institute is of the form 5 (days) * 8 (periods) slots (intersection of day and period). In each day, morning and afternoon session has four hours. Each course has six theory and three laboratory subjects. Each theory should be allotted for four hours and laboratory subject for 3 continuous hours in a week. Due to room conflict, each practical will be conducted for 3 days by dividing the students into 3 batches. Thereby, each practical should be monitored by a staff for nine hours. Extracurricular activities such as Placement and Training for 3 hours and seminar / group discussion for 2 hours must be allotted for each class. The layout of one class timetable with one theory and three laboratory subjects is given below. The free hours given for each class will be dedicated for students to make use of the library effectively.

<table>
<thead>
<tr>
<th>Days/Periods</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Lab -1</td>
<td>Lab -2</td>
<td>Lab -3</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 1 Framework of Class Timetable

2.2 Problem definition

This is done from a well defined requirement specification phase. The initial population is then generated by satisfying all the hard constraints. A fitness evaluation of the generated population is then done. The specifications of course timetabling for the Department of Information Technology of Pondicherry Engineering College is as shown in Table 1. It contains 4 classes of which one is common to all the courses in the institute and the other three are for a particular course. Therefore the three classes which contain the subjects related to a particular department are considered in this case of CTP.

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No. of classes</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>No. of Theory Subjects per Class</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>No. of Practicals per class</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>No. of timeslots/lab</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>No. of Teachers</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>No. of days</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>No. of timeslots in a day</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>No. of non-academic hours</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>No. of free hours</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>Total hours per week (including free hours)</td>
<td>40</td>
</tr>
</tbody>
</table>

Table 1 Problem Specification

2.3 Hard Constraints

Subject Conflicts
a) More than one period in a day cannot be assigned for one subject.
Student Conflicts
  b) No student can be assigned more than a course at the same period.

Teacher Conflicts
  c) No teacher can be scheduled for two classes /one class and a lab at the same period.
  d) Maximum workload of teachers must not be exceeded.

Room Conflicts
  e) Laboratory periods of different classes / in a physical laboratory location must not overlap.
  f) Laboratory periods should come in the continuous timeslot either in the morning or in the evening session but not in the first hour of both sessions.

2.4 Soft Constraints

  g) At least one period gap should be given between the lecture periods of a teacher in a day.
  h) In adjacent days, no two same periods should have the same subject.
  i) All staff should be given with first hour at least once in a week.
  j) No two morning first hours, should have the same subject.
  k) Free periods should come in the afternoon session.
  l) Maximum of 2 theories/ one theory and one lab/ 2 theories and one practical in a day only can be scheduled for a teacher in a day.

2.5 Mathematical Modeling of the Constraints

Enumerated types of parameters used in this course timetabling are:

- Classes: C[1:4]
- Days: D[1:5]
- Timeslots: TS[1:8]
- Subjects: Sub[1:24]
- Teachers: T[1:12]
- Laboratory: Lab[1:3]

Predicates used to represent the components of Course Timetabling are listed as follows:

- Class Timetable: Class[C][D][TS]
- Teacher Timetable: Teacher[T][D][TS]
- Lab Timetable: Lab[L][D][TS]

The constraints of this problem have been represented with the above mentioned data set used in our institution and multi objectives of this solver as predicates in our mathematical model.

a. More than one period in a day cannot be assigned for one subject.
   \[ \forall c \in C, \exists d \in D, \exists t_{s1}, t_{s2} \in T_{S},
   \text{class}[c][d][t_{s1}] \neq \text{class}[c][d][t_{s2}] \]

b. No student can be assigned more than a course at the same period.
   \[ \exists c_1, c_2 \in C, \exists d \in D, \exists t_s \in T_{S},
   \text{class}[c_1][d][t_{s}] = \text{class}[c_2][d][t_{s}] \]

c. No teacher can be scheduled for two classes /one class and a lab at the same period.
   \[ \forall t \in T, \exists d \in D, \exists t_s \in T_{S}, \exists c_1, c_2 \in C,
   \text{If Teacher}[t][d][t_{s}] = \text{class}[c_1][d][t_{s}] \text{ then}
   \text{Teacher}[t][d][t_{s}] \neq \text{class}[c_2][d][t_{s}] \]
d. Maximum workload of teachers must not be exceeded.
   \[ \sum_{\text{Teacher}(t)(d)(ts)} \text{workload}(t) \leq \text{Maxworkload}(t) \]

e. Laboratory periods of different classes in a physical laboratory location must not overlap.
   \[ \exists d \in D, \exists c_1, c_2 \in C, \exists ts \in TS, \exists l \in \text{Lab} \]
   \[
   \text{If Class}[c_1][d][ts] = \text{Lab}[l][d][ts] \text{ then Class}[c_2][d][ts] \neq \text{Lab}[l][d][ts]
   \]

f. Laboratory periods should come in the continuous timeslot either in the morning or in the evening session but not in the first hour of both sessions.
   \[ \exists c \in C, \exists d \in D, \exists l \in \text{Lab}, ts \in \{(2,3,4), (6,7,8)\}, \]
   \[
   \text{Class}[c][d][ts] = \text{Lab}[l][d][ts]
   \]

g. At least one period gap should be given between the lecture periods of a teacher in a day.
   \[ \forall t \in T, \forall d \in D, \forall c \in C, \forall ts \in TS, \]
   \[
   \text{If Class}[c][d][tsi] = \text{Teacher}[t][d][tsi] \text{ then Class}[c][d][tsi+1] \neq \text{Teacher}[t][d][tsi]
   \]

h. In adjacent days, no two same periods should have the same subject.
   \[ \forall c \in C, \exists d_1, d_i + 1 \in D, \forall ts \in TS, \]
   \[
   \text{Class}[c][d][ts] \neq \text{Class}[c][d_i+1][ts]
   \]

i. All staff should be given with first hour at least once in a week.
   \[ \exists t \in T, \forall d \in D, \forall c \in C, \forall ts \in TS, \]
   \[
   \sum_{\text{Teacher}(t)(d)(ts)} \geq 1
   \]

j. No two morning first hours should have the same subject.
   \[ \exists c \in C, \forall d \in D, \exists s_1 \in \text{Sub} \]
   \[
   \sum_{i=1}^{5} \left( \text{Class}[c][d][0] = s_1 \right) = 1
   \]

k. Free periods should come in the afternoon session.
   \[ \exists c \in C, \exists d \in D, ts \in \{5/6/7/8\} \]
   \[
   \text{Class}[c][d][ts] = \emptyset \text{ and } \sum \text{Class}[c][d][ts] = 2
   \]

l. Maximum of 2 theories/ one theory and one lab/ 2 theories and one practical in a day only can be scheduled for a teacher in a day.
   \[ \exists t \in T, \exists d \in D, \forall ts \in TS, \]
   \[
   \sum_{i=1}^{4} \text{Teacher}(t)[d][ts] = 2 \text{Theory} \quad / 1 \text{Theory and 1 Laboratory work} / 2 \text{Theory and 1 Laboratory work in a day} \]
2.6 Fitness Function

The fitness function of course timetabling is described in the mathematical form with objective function (1) (Evaluation Function) as follows. The Evaluation Function is,

\[ \min f(T) = \sum_{j=1}^{n} p(j) V(j) \]

(1)

Where: \( p(j) \) = Penalty cost of soft constraint \( j \) on \( T \). \( V(j) \) = Validity of Soft constraint \( j \). If \( j \in SC \) on \( T \) is satisfied, then \( V(j) = 0 \). Otherwise \( V(j) = 1 \).

2.7 Penalty Calculation

To exhibit the degree of violation of soft constraints, while evaluating the fitness value, penalty costs will be assigned. The association of penalty for violations in each constraint has been mathematically represented as follows.

1. The penalty value 5 is assigned for each occurrence of continuous lecture periods of a teacher.

   If \( \forall c \in C, \exists d \in D, \exists t \in T, \forall ts \in TS, \) 
   Class[c][d][ts] = Teacher[t][d][ts] then 
   Class[c][d][ts+1] = Teacher[t][d][ts] 
   penalty = penalty + 5.

2. In adjacent days, no two same periods should have the same subject.

   If \( \forall c \in C, \exists di, \exists di+1 \in D, \forall ts \in TS, \) 
   Class[c][di][ts] = Class[c][di+1][ts] 
   Then penalty = penalty + 5.

3. All staff should be given with first hour at least once in a week.

   If \( \exists t \in T, \forall d \in D, \forall c \in C, \forall ts \in TS, \) 
   \( \sum Teacher[t][d][ts] < 1 \) 
   Then penalty = penalty + 5.

4. No two mornings first hours should have the same subject.

   \( \exists c \in C, \forall d \in D, \exists s1 \in Sub, \) 
   if \( \sum_{i=1}^{5} (Class[c][d][0] = s1) \) = i 

   Then penalty = penalty + (i-1)*5.

5. Free periods should come in the afternoon session.

   If \( \exists c \in C, \exists d \in D, \forall ts \in \{1/2/3/4\} \) 
   Class[c][d][ts] = \( \Phi \) and \( \sum Class[c][d][ts] = \Phi \) > 2 
   Then penalty = penalty + 5.

6. Maximum of 2 theories/ one theory and one lab/ 2 theories and one practical in a day only can be scheduled for a teacher.

   If \( \exists t \in T, \exists d \in D, \forall ts \in TS, \) 
   if \( \sum Teacher[t][d][ts] \leq 2 \) Theory / 1 Theory and 1 Laboratorywork / 2 Theory and 1 Laboratory work in a day only 
   Then penalty = penalty + 5.
3 Approaches for Solving Timetables

Burke, Kingston and de Werra (2004) [3] defined timetabling as: a problem with four parameters: a finite set of timeslots; a finite set of resources; a finite set of meetings; and 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.

In the timetabling literature, many approaches have been reported for automation of class timetabling over the period of 40 years. This is solved either in operational research domain or in artificial intelligence domain. Graph-based sequential techniques, constraint-based techniques, local search-based techniques (Tabu Search, Simulated Annealing etc.), population-based algorithms (Evolutionary Algorithms, Memetic Algorithms, Ant Algorithms, etc.) have been used to solve these problems. Recently, meta-heuristics, hyper-heuristics and hybridization of above techniques are used to find the optimum in the search space.

Graph coloring is one of the earliest procedures used to find a feasible solution for this type of problems. In his thesis (1969), De Werra used graph coloring approach to find a solution for the course timetabling problem. De Werra (1995)[4] discussed some combinatorial methods for course timetabling problem. He provided formulations with graph coloring for a set of class-teacher timetabling problems and discussed the complexity of this formulation.

Integer programming has been used to formulate this problem, and good solutions have been generated in spite of the scale of this type of problems: Tripathy (1984) [5] formulated the timetabling problem as large integer linear programming problem with zero-one variables. In order to reduce the scale of the problem, he used a grouping operation for a new definition of the variables. A Lagrangian relaxation technique combined with a branch and bound method is used to solve the original problem. Daskalaki et al. (2004) [6] propose a zero-one integer program formulation, similar to our real case. They include, in this formulation, constraints concerning a large number of various rules and regulations that are commonly encountered in academic institutions.

Ferland and Roy (1985) [7] formulated the timetabling problem as assignment of activities to resources using a mathematical programming procedure. They decompose it into two sub-problems solved sequentially: the first concerns the scheduling of classes to periods and the second concerns the assignment of classes to classrooms.

Aubin and Ferland (1989) [8] developed a more general procedure dealing with large scale timetabling problems. They also decompose the problem into two sub-problems: the timetabling sub-problem and the grouping sub-problem. A heuristic approach was proposed to solve this problem by working simultaneously on the timetabling and the grouping sub-problems until a feasible solution is found and no improvement is possible.

Al-Yakoob and Sherali (2006) [9] consider the faculty-class assignment problem. Their goal was to minimize dissatisfaction among faculty members.

Galley and Mizrach (1986) [10] formulated the problem of assigning classes to rooms as a large 0-1 integer program, and they used a fast heuristic which is embedded into a decision support system. This system was developed for the problem at the University of California at Berkeley. Dinkel et al. (1989) [11] solved the problem of scheduling teacher to courses, to 4-classrooms and to time slots using a network based decision support system approach. Ferland and Fleurent (1994) [12] developed a decision support system for course timetabling called SAPHIR. This system is flexible since the user can modify some parameters such as data on courses, instructors, rooms and students. This automatic system embeds the heuristic algorithm approaches developed in Aubin and Ferland (1989) [8].

Burke et al. are the first authors to adapt the case-based reasoning to university timetabling problems. Burke et al. (2000) [13] used case-based-reasoning (CBR) with attribute graphs to solve the course timetabling problems structurally. The main idea of this procedure is to exploit previous solutions of the same problem to find good quality solutions to the new problem. Further he improved the structured case-based-reasoning approach for course timetabling problems developed in the previous paper to tackle extended problems. Burke et al. (2003) [14] developed a first version of a hyper-heuristic method using CBR to select the heuristic to use for solving course timetabling problems. They used knowledge discovery techniques to predict the best heuristic for the new problem on the basis of the previous knowledge. Burke et al. (2006) [15] investigated a case-based heuristic selection approach for the automation of both course and exam timetabling. They used knowledge discovery techniques to get the knowledge of problems modeling and compare cases to select heuristics and refine the case base. Burke et al. (2006) [16] tried to find a procedure to deal with large and complex timetabling problems. They used multiple-retrieval
approach to decompose the original problem into a set of sub-problems with sufficiently small size to be solved with simple procedures.

Recently, meta-heuristic search techniques (Glover and Kochenberger, 2003) [17] have been investigated and seem to have been very successful in solving a variety of timetabling problems. These include Tabu Search (e.g. Costa, 1994 [18], Schaerf (1996) [19]) Simulated Annealing (e.g. Dowsland, 1996 [20]) and Evolutionary Algorithms. Burke et al. (2007) [21] investigated a hyper-heuristic framework in which both Tabu search and graph heuristic are used as general methods for solving a large spectrum of university timetabling problems.

4 Implementation of Course Timetabling

In Artificial Intelligence domain, Hill climbing is a heuristic search which belongs to the family of informed local search. Steepest-Ascent Hill Climbing is a variation of simple hill climbing which considers all the moves from the current state and selects the best one as the next state.

Likely to get the better results in the local search and to analyze the progress of getting optimal results for CTP at our Institute, implements the problem in steepest-ascent hill climbing.

4.1 Combination (State) Representation

Combination is represented in a three dimensional matrix. Lower index represents periods, middle represents a day and upper represents a class. Then, the value of each cell (timeslot) of the matrix represents allotment scheduled in the corresponding class and period. N number of such combinations is used to form the states space in each generation.

4.2 Heuristic Approach

Constructive heuristic approach is used for initializing the population with number of chromosomes. The initialization process is even more important due to the fact that it is the process that encodes the input information into the chromosome’s representation. This approach starts with empty timetable. A feasible timetable is obtained by adding or removing appropriate subjects from the schedule until the hard constraints are met. The initialization procedure consists of the following steps:

```plaintext
for each chromosome
  for each class
    for each Lab Subject
      make entry for continuous time slots in either of the sessions except first period and without room conflicts
    end
  for each theory subject
    make entry for all periods in class and teachers timetable without violating hard constraints
  end
end
```

Fig. 2 Population Initialization Procedure

4.3 Methods for controlling the execution

Searching for optimal combination could be controlled by the following strategies.
Reaches the optimum. This option enables if a combination satisfying all hard and soft constraints has been obtained and saves the result of combination and quit from the execution.

Backtracking. This option enables when a combination having the best evaluation value is not reached in the successive levels. By enabling this, another initial timetable or next best adjacent combination in the current level has been taken for further iterations.

Breadth Wise Search. When two successive combinations of same evaluation value are found in two successive levels, to continue the neighborhood search from the current level, the adjacent combination (breadth wise) is taken for further processing.

4.4 Execution of Steepest Ascent – Hill Climbing

With the elements described in the problem description and by satisfying hard constraints, a timetable (state) with minimum fitness in the states space is to be found by Steepest- Ascent Hill climbing local search algorithm [22]. The process starts from random generation of a timetable. A neighborhood feasible state establishes state space formed by altering timeslots (Theory Subject only) called as generation. Fitness value of every combination is found from its penalties of soft constraints. To get the better neighbor, when traverse from current state, table with minimum fitness (optimality) is taken from the space and is taken for further exploration. This process can be repeated either till gets optimum or for specific number of generations or reaching with certain number of combinations.

5 Experimental Results

The algorithm has been implemented using Java (Jdk 1.6) and the results have been computed by executing the program in an Intel Pentium IV 2.40Ghz processor computer. This algorithm is tested with a standard timetable requirements specified in Section II. It has been observed that the timetable created by this algorithm is more optimal than one which is created manually.

This has been tested by exploring fixed and varying number of feasible combinations in each generation. The results for different parameters involved are tabulated.

<table>
<thead>
<tr>
<th>Generation Number</th>
<th>Actual Fitness Cost</th>
<th>Min Fitness Value</th>
<th>Min fitness location</th>
<th>Max. fitness value</th>
<th>Max. fitness location</th>
<th>No. of feasible combinations</th>
<th>Cum. No. of Infeasible Combinations</th>
<th>Time (in ms)</th>
<th>Space (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>105</td>
<td>80</td>
<td>85</td>
<td>130</td>
<td>21</td>
<td>100</td>
<td>81</td>
<td>1750</td>
<td>45167232</td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>65</td>
<td>33</td>
<td>105</td>
<td>21</td>
<td>100</td>
<td>159</td>
<td>1985</td>
<td>45167232</td>
</tr>
<tr>
<td>3</td>
<td>65</td>
<td>50</td>
<td>59</td>
<td>90</td>
<td>85</td>
<td>100</td>
<td>237</td>
<td>2188</td>
<td>45167232</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>40</td>
<td>11</td>
<td>75</td>
<td>85</td>
<td>100</td>
<td>315</td>
<td>2406</td>
<td>45254776</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>30</td>
<td>48</td>
<td>65</td>
<td>10</td>
<td>100</td>
<td>393</td>
<td>2610</td>
<td>45254776</td>
</tr>
<tr>
<td>6</td>
<td>30</td>
<td>25</td>
<td>23</td>
<td>55</td>
<td>10</td>
<td>100</td>
<td>471</td>
<td>2813</td>
<td>45254776</td>
</tr>
<tr>
<td>7</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>50</td>
<td>10</td>
<td>100</td>
<td>549</td>
<td>3031</td>
<td>45254776</td>
</tr>
<tr>
<td>8</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>50</td>
<td>10</td>
<td>100</td>
<td>627</td>
<td>3235</td>
<td>45254776</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>50</td>
<td>10</td>
<td>100</td>
<td>705</td>
<td>3438</td>
<td>45342272</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>50</td>
<td>10</td>
<td>100</td>
<td>783</td>
<td>3656</td>
<td>45342272</td>
</tr>
</tbody>
</table>
From the above illustrations, it is observed that after certain generations, rate of fitness improvement is very low and even no improvement also found.

**Table 3 Local Search with Varying Combinations**

<table>
<thead>
<tr>
<th>Gen No.</th>
<th>Actual Fitness Cost</th>
<th>Min Fitness Value</th>
<th>Min fitness Location</th>
<th>Max. fitness Value</th>
<th>Max. fitness Location</th>
<th>Cum. No. of feasible combinations</th>
<th>Cum. No. of Infeasible Combinations</th>
<th>Total Combinations in each gen.</th>
<th>Total Time (in ms)</th>
<th>Space (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>105</td>
<td>75</td>
<td>36</td>
<td>145</td>
<td>9</td>
<td>148</td>
<td>294</td>
<td>262</td>
<td>278</td>
<td>45167248</td>
</tr>
<tr>
<td>2</td>
<td>75</td>
<td>60</td>
<td>215</td>
<td>145</td>
<td>147</td>
<td>294</td>
<td>294</td>
<td>262</td>
<td>278</td>
<td>45254792</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>50</td>
<td>335</td>
<td>145</td>
<td>293</td>
<td>440</td>
<td>588</td>
<td>524</td>
<td>278</td>
<td>45254792</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>45</td>
<td>452</td>
<td>145</td>
<td>439</td>
<td>588</td>
<td>738</td>
<td>652</td>
<td>278</td>
<td>45429784</td>
</tr>
<tr>
<td>5</td>
<td>45</td>
<td>40</td>
<td>643</td>
<td>145</td>
<td>587</td>
<td>738</td>
<td>888</td>
<td>780</td>
<td>278</td>
<td>45517280</td>
</tr>
<tr>
<td>6</td>
<td>40</td>
<td>35</td>
<td>823</td>
<td>145</td>
<td>737</td>
<td>888</td>
<td>888</td>
<td>780</td>
<td>278</td>
<td>45692272</td>
</tr>
<tr>
<td>7</td>
<td>35</td>
<td>35</td>
<td>115</td>
<td>145</td>
<td>256</td>
<td>1050</td>
<td>1234</td>
<td>990</td>
<td>278</td>
<td>45806968</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>35</td>
<td>353</td>
<td>145</td>
<td>369</td>
<td>1234</td>
<td>1234</td>
<td>1234</td>
<td>278</td>
<td>45963258</td>
</tr>
<tr>
<td>9</td>
<td>35</td>
<td>35</td>
<td>115</td>
<td>145</td>
<td>258</td>
<td>1424</td>
<td>1424</td>
<td>1424</td>
<td>278</td>
<td>46025463</td>
</tr>
<tr>
<td>10</td>
<td>35</td>
<td>35</td>
<td>789</td>
<td>145</td>
<td>147</td>
<td>1590</td>
<td>1590</td>
<td>1590</td>
<td>278</td>
<td>46256983</td>
</tr>
</tbody>
</table>
Possible combinations are explored for a timetable and forming varying numbers of feasible and infeasible combinations. Even explored more in numbers, in this data set no change found in the maximum cost of each generation. As in above, after some generation, no change is found in the fitness value and causes local optima. Increase in numbers, increases space and time complexity.

Trying to get global optima, diversity of populations planned to generate by evolutionary algorithms (genetic algorithms) with variation in parameters to increase the convergence rate.

6 Conclusion

As a first step of static multi constrained combinatorial problems optimization, course timetabling problem is implemented using local search algorithms (steepest ascent hill climbing) with heuristics and search algorithms significance has been tested and is proved with parameters such as number of generations, fixed and varying number of combinations in state space etc.

Further, hybridization of evolutionary algorithms (genetic algorithm) with variation in operators and local search algorithms is planned to explore importance of local search and genetic operators over the progress of convergence.

References


