Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

Examination Timetabling Automation using Hybrid Meta-heuristics

Miguel De Brito e Nunes

(Licenciado em Engenharia Informática e de Computadores)

Trabalho de projeto realizado para obtenção do grau
de Mestre em Engenharia Informática e de Computadores

Relatório Preliminar

Orientadores:
Artur Jorge Ferreira
Nuno Miguel da Costa de Sousa Leite

30 de Março de 2015

Contents

List of Figures

1

Introduction

Many people believe that AI (Artificial Intelligence) was created to imitate human behavior and the way humans think and act. Even though people are not wrong, AI was also created to solve problems that humans are unable to solve, or to solve them in a shorter time window, with a better solution. Humans may take days to find a solution, or may not find a solution that fits their needs. Optimization algorithms, that is, methods that seek to minimize or to maximize some criterion, may deliver a very good solution in minutes, hours or days, depending on how much time the human is willing to use in order to get a proper solution.

A concrete example of this type of problems is the creation of timetables. Timetables can be used for educational purposes, sports scheduling, transportation timetabling, among other applications. The timetabling problem consists in scheduling a set of events (e.g., exams, people, trains) to a specified set of time slots, while respecting a predefined set of rules. In some cases, the search space is so limited by the constraints that one is forced to relax them in order to find a solution.

1.1 Educational Timetabling Problems

This class of timetabling problems involve the scheduling of classes, lectures or exams on a school or university in a predefined set of timeslots while satisfying a set of rules. Examples of rules are: a student can't be present in two classes at the same time, a student can't have two exams on the same day or even an exam must be scheduled before another.

Depending on the institution type and if we're scheduling classes/lectures or exams the timetabling problem is divided into three main types:

- Examination timetabling - consists on scheduling university course exams avoiding the overlap of exams containing students from the same course and spreading the exams as much as possible in the timetable.
- Course timetabling - consists on scheduling lectures considering the multiple university courses, avoiding the overlap of lectures with common students.
- School timetabling - consists on scheduling all classes in a school, avoiding the need of students being present at multiple classes at the same time.

In this project, the main focus is the examination timetabling problem.

The process of creating a timetable requires that the final solution follows a set of constraints. The type of constraint depends on the timetabling problem type and problem specifications. The constraints are divided in two groups: *hard constraints* and *soft constraints*. Hard constraints are a set of rules which must be followed in order to get a feasible solution. On the other hand, soft constraints represent the views of the different interested players (e.g., institution, students, nurses, train operators) in the resulting timetable. The satisfaction of this type of constraints is not mandatory as is for the case of the hard constraints. In the timetabling problem, the goal is usually to optimize a function with a weighted combination of the different soft constraints, while satisfying the set of hard constraints.

1.2 Objectives

This project's main objective is the production of a prototype application which serves as an examination timetabling generator tool. The problem at hand focus on the specifications introduced in the International Timetabling Competition 2007 (ITC 2007), *First track*, which includes 12 benchmark instances. In the ITC 2007 specification, the examination timetabling problem considers a set of periods, room assignment, and the existence of constraints analogous to the ones present in real instances.

The application's requirements are the following:

- Automated generation of examination timetables, considering the ITC 2007 specifications (*mandatory*).
- Validation (correction and quality) of a timetable provided by the user (*mandatory*).
- Graphical User Interface to allow the user to edit generated solutions and to optimize user's edited solutions (*optional*).

This project is divided into two main phases. The first phase consists on studying some techniques and solutions for this problem emphasizing meta-heuristics like: Genetic Algorithms (GA), Simulated Annealing (SA), Tabu Search (TS), and some of its hybridizations. The second phase consists in the development of the selected algorithms and promising hybridizations, and test them using ITC 2007 data. A performance comparison between the proposed algorithms and the state-of-the-art algorithms will be made.

1.3 Planning

In this section, more details are given about the development of the two project phases. A Gantt diagram is presented showing each project's phase timeline. On the first phase, the author studied the educational timetabling problem and in particular the examination timetabling problem and related literature. This work is documented in this progress report. The second phase's main goal is to develop solutions for the ITC 2007 problem instance.

One of the approaches to be developed will use a local search algorithm (e.g., Simulated Annealing) to improve the solution obtained using a graph coloring heuristic. The results of this

approach will be compared against other approaches applied to the ITC 2007's timetabling problem, namely the first five winners. In a subsequent approach, instead of using a single-solution based meta-heuristic, the author intends to use a population-based meta-heuristic (e.g., Genetic Algorithm) hibridized with a local search algorithm (e.g., Simulated Annealing), or a combination of other meta-heuristics. A performance comparison will be made in which the proposed solution methods are compared with the five ITC 2007 finalists and other state-of-the-art approaches.

As an optional project requirement, a GUI for editing generated solutions by the human planner will be developed.

The Gantt diagram presented on Figure 1.1 shows a detailed planning for the development of all the main topics of this project including both phases mentioned above. The colors presented on the diagram are green and blue, which designate the first and second phase, respectively.

1.4 Document Organization

The rest of the document is organized as follows. Section 2, entitled State-of-the-Art, specifies the timetabling problem focusing on examination timetabling and existing approaches applied to the ITC 2007 benchmark data. This chapter includes a survey of the most important paradigms and algorithmic strategies used to tackle the timetabling problem. Next, the methods of the five ITC 2007 finalists are resumed. The next chapters should address the implementation aspects of the solution method. A chapter containing the evaluation of the proposed algorithms and comparison with other competing algorithms, and a final chapter containing the conclusions of the work, will also be included.

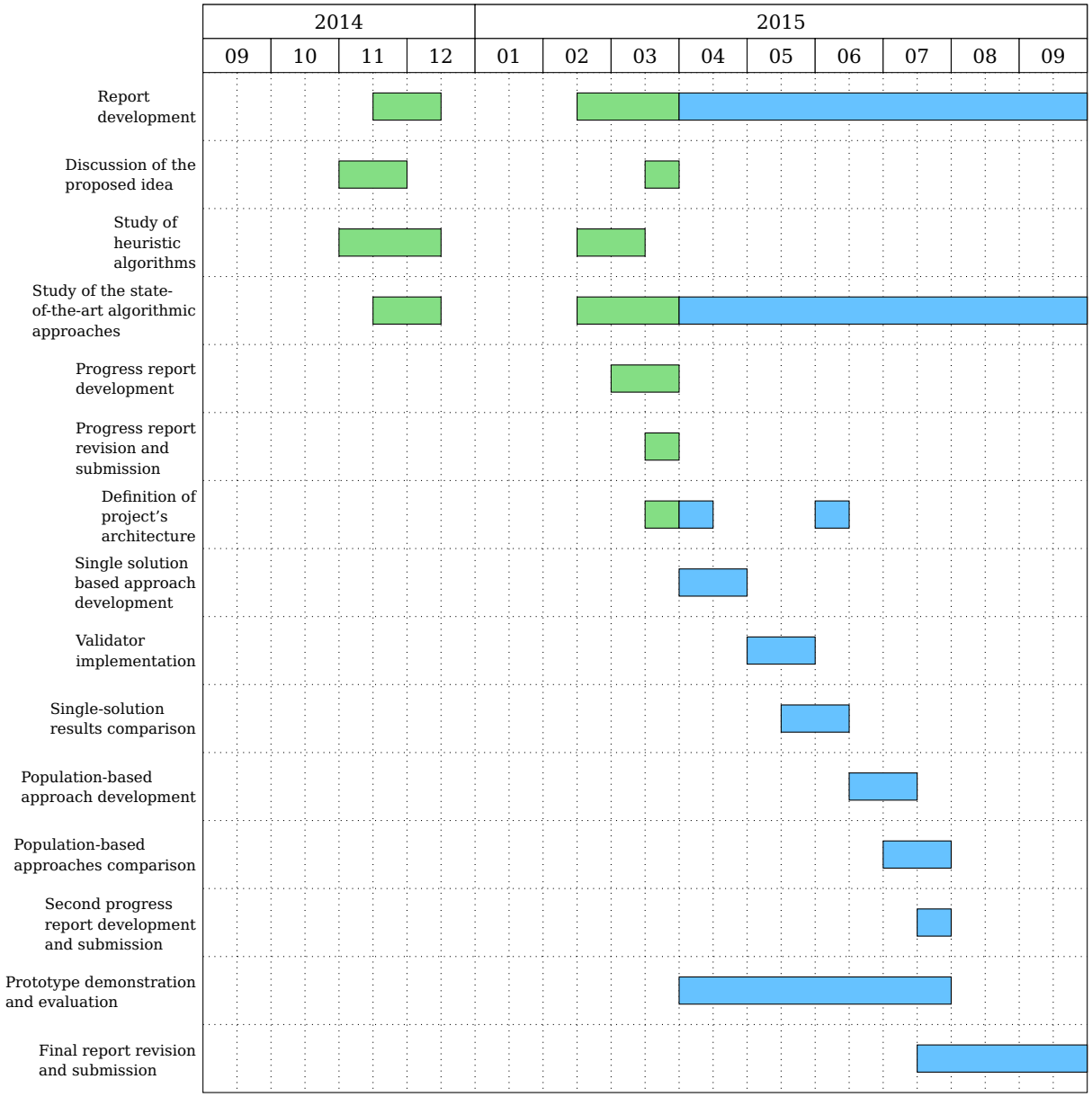

Figure 1.1: Gantt diagram of the project activities.

2

State of the Art

In this section, we review the state-of-the-art of the problem at hand. We start by describing why timetabling is a rather complex problem, some possible approaches to solve it and some of the existing solutions, specifically for the ITC 2007 benchmarks.

2.1 Timetabling Problem

When solving timetabling problems, it is possible to generate one of multiple types of solutions which are *feasible*, *non feasible*, *optimal* or *sub-optimal*. A feasible solution solves all the mandatory constraints, unlike non feasible solutions. An optimal solution is the best possible feasible solution given the problem constraints. A problem may have multiple optimal solutions. Lastly, non-optimal solutions are feasible solutions that have sub-optimal values.

Timetabling automation is a subject that has been a target of research for about 50 years. The timetabling problem may be formulated as a search or an optimization problem [2]. As a search problem, the goal consists on finding a feasible solution that satisfies all the hard constraints, while ignoring the soft constraints. By posing the timetabling problem as an optimization problem, one seeks to minimize (considering a minimization problem) the violations of soft constraints while satisfying the hard constraints. Typically, the optimization is done after the use of a search procedure for finding an initial feasible solution.

The basic examination timetabling problem, where only the *clash* hard constraint is observed, reduces to the graph coloring problem [3], which is a well studied problem. The clash hard constraint specifies that no conflicting exams should be scheduled at the same time slot. Deciding whether a solution exists in the graph coloring problem, is a NP-complete problem [4]. Considering the graph coloring as an optimization problem, it is proven that the task of finding the optimal solution is also a NP-Hard problem [4]. Graph Coloring problems are explained further in Section 2.2

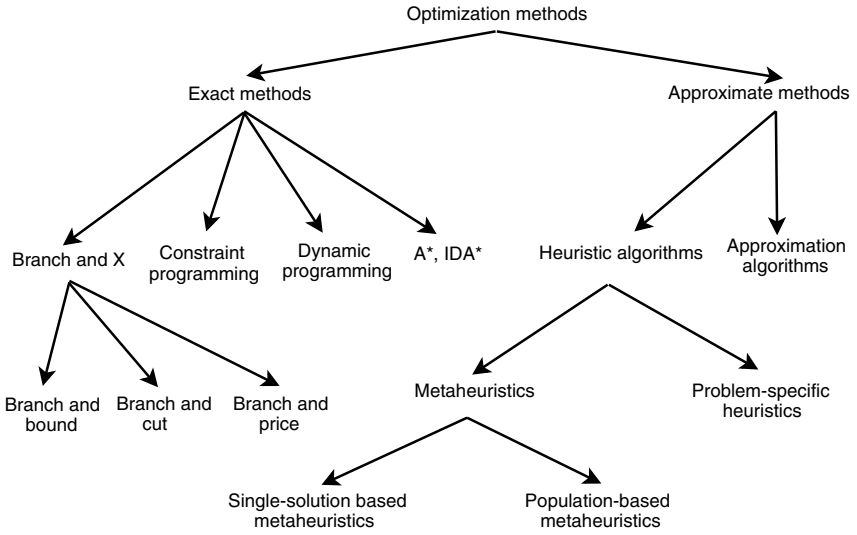

Figure 2.1: Optimization methods: taxonomy and organization (adapted from [1]).

2.2 Existing Approaches

Figure 2.1 depicts a taxonomy for the known optimization methods. These methods are divided into *Exact methods* and *Approximate methods*.

Timetabling solution approaches are usually divided into the following categories [5]: *exact algorithms* (Branch-and-Bound, Dynamic Programming), *graph based sequential techniques, local search based techniques* (Tabu Search, Simulated Annealing, Great Deluge), *population based algorithms* (Evolutionary Algorithms, Memetic algorithms, Ant Colony algorithms, Artificial immune algorithms), *Multi-criteria techniques*, *Hyper-heuristics*, *Decomposition/clustering techniques*. Hybrid algorithms, which combine features of several algorithms, comprise the state-of-the-art. Due to its complexity, approaching the examination timetabling problem using exact method approaches can only be done for small size instances. Real problem instances found in practice are usually of large size, making the use of exact methods impracticable. Heuristic solution algorithms have been usually employed to solve this problem.

Real problem instances are usually solved by applying algorithms which use both *heuristics* and *meta-heuristics*. Heuristic algorithms are problem-dependent, meaning that these are adapted to a specific problem in which one can take advantage of its details. Heuristics are used to generate a feasible solution, focusing on solving all hard constraints only. Meta-heuristics, on the other hand, are problem-independent. These are used to, given the feasible solution obtained using heuristic algorithms, generate a better solution focusing on satisfying as many soft constraints as possible.

Most of the existing meta-heuristic algorithms belong to one of the following three categories: One-Stage algorithms, Two-Stage algorithms and algorithms that allow relax-

ations [6]. The One-Stage algorithm is used to get a solution, which the goal is to satisfy both hard and soft constraints at the same time. The Two-Stage algorithms are the most used types of approaches. This category is divided in two phases: the first phase consists in not considering the soft constraints and focusing on solving hard constraints to obtain a feasible solution; the second phase is an attempt to find the best solution, trying to solve the largest number of soft constraints as possible, given the solution of the first phase. Algorithms that allow relaxation can weaken some constraints in order to solve the *relaxed problem*, while considering the elimination the used relaxations.

2.2.1 Exact methods

Approximation algorithms like heuristics and meta-heuristics proceed to enumerate partially the search space and, for that reason, they can't guarantee finding the optimal solution. Exact approaches perform a complete enumeration of the search space and thus guarantee that the encountered solution is optimal. A negative aspect is the time taken to find the solution. If the decision problem is very difficult (e.g. NP-Complete), in practical scenarios, given large size problem instances, using this approach may not be possible due to the long execution time.

Constraint-Programming Based Technique

The Constraint Programming Based Technique (CPBT) allows direct programming with constraints which gives ease and flexibility in solving problems like timetabling. Two important features about this technique are the use of backtracking and logical variables. Constraint programming is different from other types of programming, as in these types it is specified the steps that need to be executed, but in constraint programming only the properties (hard constraints) of the solution, or the properties that should not be in the solution, are specified [5].

Integer Programming

The Integer Programming (IP) is a mathematical programming technique in which the optimization problem to be solved must be formulated as an Integer Problem. If the objective function and the constraints must be linear, and all problem variables are integer valued, then the IP problem is termed Integer Linear Problem (ILP). In the presence of both integer and continuous variables, then the problem is called Mixed-Integer Linear Programming (MILP). Schaerf [2] surveys some approaches using the MILP technique to school, course, and examination timetabling.

2.2.2 Graph Coloring Based Techniques

As mentioned previously, timetabling problems can be reduced to a graph coloring problem. Considering this relation between the two problems, several authors used two-phased algorithms in which graph coloring heuristics were applied in the first phase, to obtain an initial feasible solution.

Graph Coloring Problem

The Graph Coloring (GC) problem consists in assigning colors to an element type of a graph which must follow certain constraints. The simplest sub-type is the *vertex coloring*, which the main goal is to, given a number of vertices and edges, color the vertices so that no adjacent vertices have the same color. In this case the goal is to find a solution with the lowest number of colors as possible.

The examination timetabling problem can be transformed into a graph coloring problem as follows. The exams corresponds to vertices and there exists an edge connecting each pair of conflicting exams (exams that have students in common). With this mapping only the clash hard constraint is take into consideration. Thus, soft constraints are ignored [5].

Given the mapping between the GC problem and the examination timetabling problem, GC heuristics like *Saturation Degree Ordering* are very commonly used to get the initial solutions. Others like *First Fit* and other *Degree Based Ordering* techniques (*Largest Degree Ordering*, *Incidence Degree Ordering*) are also heuristic techniques for coloring graphs [7].

2.2.3 Meta-heuristics

Meta-heuristics, as mentioned above, usually provide solutions for optimization problems. In timetabling problems, meta-heuristic algorithms are used to optimize the feasible solutions provided by heuristics, like the GC heuristics. Meta-heuristics are divided in two main subtypes, which are *Single-solution meta-heuristics* and *Population-based meta-heuristics* [1].

Single-solution meta-heuristics

Single-solution meta-heuristics main goal is to modify and to optimize one single solution, maintaining the search focused on local regions. This type of meta-heuristic is therefore exploitation oriented. Some examples of this type are *SA*, *Variable-Neighborhood Search*, *TS*, and *Guided Local Search* [1].

Population-based meta-heuristics

Population-based meta-heuristics main goal is to modify and to optimize multiple candidate solutions, maintaining the search focused in the whole space. This type of meta-heuristic is therefore exploration oriented. Some examples of this type are *Particle Swarm*, *Evolutionary Algorithms*, and *Genetic Algorithms* [1].

2.2.4 ITC 2007 Examination timetabling problem: some approaches

In this section, the five ITC 2007 - Examination timetabling track - finalists approaches are described. This timetabling problem comprises 12 instances of different degree of complexity. Through the available website, competitors could submit their solutions for the given

benchmark instances. Submitted solutions were evaluated as follows. First, it is checked if the solution is feasible and a so-called distance to the feasibility is computed. If it is feasible, the solution is further evaluated based on the fitness function, which measures the soft constraints total penalty. Then, competitors' solutions are ranked based on the distance to feasibility and solution's fitness value. The method achieving the lower distance to feasibility value is the winner. In the case of a tie, the competitor's solution with the lowest fitness value wins. A solution is considered feasible if the value of distance to feasibility is zero. The set of hard constraints is the following:

- no student must be elected to be present in more than one exam at the same time;
- the number of students in a class must not exceed the room's capacity;
- exam's length must not surpass the length of the assigned timeslot;
- exams ordering hard constraints must be followed; e.g., $Exam_1$ must be scheduled after $Exam_2$;
- room assignments hard constraints must be followed; e.g., $Exam_1$ must be scheduled in $Room_1$.

It is also necessary to compute the fitness value of the solution which is calculated as an average sum of the soft constraints penalty. The soft constraints are listed below:

- two exams in a row - a student should not be assigned to be in two adjacent exams in the same day;
- two exams in a day - a student should not be assigned to be in two non adjacent exams in the same day;
- period spread - reduce the number of times a student is assigned to be in two exams that are N timeslots apart;
- mixed durations - reduce the number of exams with different durations that occur in the same room and period;
- larger exams constraints - reduce the number of large exams that occur later in the timetable;
- room penalty - avoid assigning exams to rooms with penalty;
- period penalty - avoid assigning exams to periods with penalty.

To get a detailed explanation on how to compute the values of fitness and distance to feasibility based on the weight of each constraint, please check the ITC 2007 website [8].

We now review the ITC 2007's five winners approaches. The winners list of the ITC 2007 competition is as follows:

- 1st Place - Tomáš Müller
- 2nd Place - Christos Gogos
- 3rd Place - Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki
- 4th Place - Geoffrey De Smet
- 5th Place - Nelishia Pillay

We now briefly describe these approaches.

Tomáš Müller's approach [9] was actually used to solve all three problems established by the ITC 2007 competition. He was able to win two of them and to be finalist on the third. For solving the problems, he opted for an hybrid approach, organized in a two-phase algorithm. In the first phase, Tomáš used the Iterative Forward Search (IFS) algorithm [10] to obtain

feasible solutions and Conflict-based Statistics [11] to prevent IFS from looping. The second phase consists in using multiple optimization algorithms. These algorithms are applied in this order: Hill Climbing (HC) [12], Great Deluge (GD) [13], and optionally SA [14].

Gogos was able to reach second place in Examination Timetabling track, right after Müller. Gogos' approach [15], like Müller's, is a two-phase approach but with a pre-processing stage. The first phase starts by the application of a pre-processing stage, in which the hidden dependencies between exams are discovered in order to speed up the optimization phase. After the pre-processing stage, a construction stage takes place, using a meta-heuristic called *Greedy Randomized Adaptive Search Procedure (GRASP)*. In the second phase, optimization methods are applied in this order: HC, SA, IP (the Branch and Bound procedure), finishing with the so-called Shaking Stage, which is only used on certain conditions. Shaking Stage *shakes* the current solution creating an equally good solution, which is given to the SA. The objective of this stage to make SA restart with more promising solutions and generate better results.

Mitsunori Atsuta, Koji Nonobe and Toshihide Ibaraki ended up in third place on the Examination Timetabling track and won third and second place on the other tracks as well, with the same approach for all of them. The approach [16] consists on applying a constraint satisfaction problem solver adopting an hybridization of TS and Iterated Local Search.

Geoffrey De Smet's approach [17] differs from all others because he decided not to use a known problem-specific heuristic to obtain a feasible solution, but instead used what is called the *Drool's rule engine*, named *drools-solver* [18]. By definition, the drools-solver is a combination of optimization heuristics and meta-heuristics with very efficient score calculation. A solution's score is the sum of the weight of the constraints being broken. After obtaining a feasible solution, Geoffrey opted to use a local search algorithm to improve the solutions obtained using the drools-solver.

Nelishia Pillay opted to use a two-phase algorithm variation, using *Developmental Approach based on Cell Biology* [19], which the goal consists in forming a well-developed organism by the process of creating a cell, proceeding with cell division, cell interaction and cell migration. In this approach, each cell represents a timeslot. The first phase represents the process of creating the first cell, cell division and cell interaction. The second phase represents the cell migration.

References

1. E. Talbi, *Metaheuristics - From Design to Implementation*. Wiley, 2009.
2. A. Schaerf, "A survey of automated timetabling," *Artif. Intell. Rev.*, vol. 13, no. 2, pp. 87–127, 1999.
3. T. Jensen, *Graph Coloring Problems*. John Wiley & Sons, Inc., 2001.
4. S. Arora and B. Barak, *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.
5. R. Qu, E. Burke, B. McCollum, L. Merlot, and S. Lee, "A survey of search methodologies and automated system development for examination timetabling," *J. Scheduling*, vol. 12, no. 1, pp. 55–89, 2009.
6. R. Lewis, "A survey of metaheuristic-based techniques for university timetabling problems," *OR Spectrum Volume 30, Issue 1 , pp 167-190*, 2007.
7. M. Carter, G. Laporte, and S. Lee, "Examination timetabling: Algorithmic strategies and applications," *The Journal of the Operational Research Society*, 1996.
8. B. McCollum, "Evaluation function." `http://www.cs.qub.ac.uk/itc2007/examtrack/exam_track_index_files/examevaluation.htm`, 2008.
9. T. Müller, "ITC2007 solver description: A hybrid approach," *Annals of Operations Research*, vol. 172, no. 1, pp. 429–446, 2009.
10. T. Müller, *Constraint-Based Timetabling*. PhD thesis, Charles University in Prague Faculty of Mathematics and Physics, 2005.
11. T. Müller, R. Barták, and H. Rudová, *Conflict-Based Statistics*. PhD thesis, Faculty of Mathematics and Physics, Charles University Malostranské nám. 2/25, Prague, Czech Republic, 2004.
12. S. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.
13. G. Dueck, "New optimization heuristics: The great deluge algorithm and the record-to-record travel," *Journal of Computational Physics*, 1993.
14. S. Kirkpatrick, D. Gelatt, and M. Vecchi, "Optimization by simmulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
15. C. Gogos, P. Alefragis, and E. Housos, "An improved multi-staged algorithmic process for the solution of the examination timetabling problem," *Annals Operations Research*, vol. 194, no. 1, pp. 203–221, 2012.
16. M. Atsuta, K. Nonobe, and T. Ibaraki, *ITC-2007 Track2: An Approach using General CSP Solver*. PhD thesis, Kwansei-Gakuin University, School of Science and Technology, Tokyo, Japan, 2007.
17. G. Smet, "Drools-solver." `http://www.cs.qub.ac.uk/itc2007/winner/bestexamsolutions/Geoffrey_De_smet_examination_description.pdf`, 2007.
18. T. J. Drools, "Drools planner user guide." `http://docs.jboss.org/drools/release/5.4.0.Final/drools-planner-docs/html_single/`.
19. N. Pillay, "A developmental approach to the examination timetabling problem." `http://www.cs.qub.ac.uk/itc2007/winner/bestexamsolutions/pillay.pdf`, 2007.