Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

A GRASP algorithm with Path Relinking for the University Courses Timetabling Problem

Edmar Hell Kampke[1]

Departamento de Computação, Centro de Ciências Agrárias, UFES, Alegre, ES

Walace de Souza Rocha[2]

Departamento de Informática, Centro Tecnológico, UFES, Vitória, ES

Maria Claudia Silva Boeres[3]

Departamento de Informática, Centro Tecnológico, UFES, Vitória, ES

Maria Cristina Rangel[4]

Departamento de Informática, Centro Tecnológico, UFES, Vitória, ES

Abstract. The timetabling problem is of great interest in the combinatorial optimization field. Given a set of disciplines, students, teachers and classrooms, the problem lies in allocating lectures in a limited number of timeslots and rooms respecting a set of constraints. In this paper, we propose a GRASP algorithm for the university timetabling formulated at ITC-2007. Hill Climbing and Simulated Annealing techniques are used as GRASP local search procedures and path relinking is implemented to improve its basic version. Computational tests were carried out, simulating the ITC-2007 competition rules and the results obtained are competitive.

Keywords. University Courses Timetabling, GRASP, Simulated Annealing, Path Relinking

1 Introduction

Scheduling problems deal with the allocation of resources into slots, respecting a set of constraints. Timetabling is a specific type of scheduling, with a wide variety of applications, such as employees scale and sports championships matches scheduling. More specifically, educational timetabling is concerned to the allocation of a set of lectures in a predetermined number of timeslots, satisfying various constraints involving teachers, students and physical space.

Real world timetabling problems are usually hard to solve manually, once they often require a great amount of time and resources. The timetabling problem is classified as NP-complete [12] for most of formulations.

[1]edmar.kampke@ufes.br
[2]walacesrocha@yahoo.com.br
[3]boeres@inf.ufes.br
[4]crangel@inf.ufes.br

 © 2015 SBMAC

Because of this, the timetabling research community organizes competitions on this problem in order to promote discussion of several formulations and resolution techniques and provide real world instances for benchmarks. Three competition called International Timetabling Competition (ITC) were held: ITC-2002, ITC-2007 and ITC-2011. The first two are devoted to the university timetabling problem and the last one, to the high school timetabling problem. In this paper we propose a Greedy Randomized Adaptive Search Procedures (GRASP) with Path Relinking (PR) method to solve the problem in track 3 of ITC-2007, which is concerned with the curriculum-based variant of the university timetabling problem. A set of 21 real world instances from the University of Udine [6] have feasible solutions and for some of them, the optimal is unknown. The computational results are compared to those obtained from other 17 algorithms, submitted to the same set of instances where results are reported in [6] and [9].

The literature for timetabling solution techniques can be broadly inserted into three important research areas: mathematical programming, logic programming and metaheuristics. We observe that the majority of the recent work in this area uses metaheuristics procedures [8] due to simplicity and quality of solutions. We can cite Simulated Annealing [2] and Tabu Search [9] as the most often employed for this problem. In [2], a Simulated Annealing solution is given to the second formulation of the ITC-2007 and the authors achieved good solutions, overcoming in certain cases, those encountered in the literature. In [9] tabu search has been applied successfully in curriculum-based course timetabling problem. For track 3 of ITC-2007 some competitors have proposed hybrid techniques, combining two or more metaheuristics, for example, [10] proposed an algorithm with four steps: generation of a initial solution with a graph coloring algorithm and application of three successive local search procedures using, respectively, hill climbing, great deluge and simulated annealing. The algorithm proposed by [10] has a specific movements of local search and was the ITC-2007 winner. In [1], a new method to compute lower bounds for instances of ITC-2007 are presented and results show that the proposed lower bound is often better than the ones found by the previous methods in the literature. In this paper we proposed an GRASP algorithm with hybrid techniques based on [10].

This paper is organized as follows. In the section 2, we describes the formulation of curriculum-based course timetabling problem adopted in track 3 of ITC-2007. Section 3 outlines our GRASP algorithm. Experimental results and conclusions are reported in Section 4.

2 Problem Definition

In the curriculum-based course timetabling problem, a curriculum is a group of courses which must be allocated in different periods because they have students in common. Availabilities for teacher and room capacities are also considered.

A period is an interval of time in which lectures can be scheduled. Given a number D of weekly days, usually five or six, and a fixed number P of periods per day, a $H[i, j]$ matrix represents a weekly timetable and a timeslot is a pair $(i \in D,\ j \in P)$.

Each course has a number of weekly lectures that must be allocated in different periods

 © 2015 SBMAC

and it is taught by a teacher and attended by a number of students.

A problem solution consists of the allocation of each lecture at a timeslot and a room. For this definition hard and soft constraints can be established. The hard constraints must always be respected. Any violation of a hard constraint generates an unfeasible timetable, which in practice is not valid. On the other hand, the soft constraints must be satisfied as much as possible, and the less the number of violations, the best is the timetable solution.

The hard and soft constraints for the formulation of curriculum-based course timetabling problem, adopted in track 3 of ITC-2007, are outlined below:

- **Hard Constraints:**
  1. **Lectures**: All courses lectures should be allocated in different timeslots. A violation occurs when a lecture is not allocated.
  2. **Conflicts**: Courses lectures of the same curriculum or taught by the same teacher should be allocated at different timeslots.
  3. **Room Occupancy**: Two lectures can not occupy the same room at the same time.
  4. **Availability**: The problem considers some unavailable periods for some courses. A lecture can not be allocated in a unavailable timeslot for the course.
- **Soft Constraints:**
  1. **Minimum Working Days**: The lectures in each course must be spread over a minimum amount of days. Each day below the minimum is counted as a violation.
  2. **Curriculum Compactness**: Courses lectures from the same curriculum must be allocated in adjacent timeslots. Each single lecture is counted as a violation.
  3. **Room Capacity**: The number of students attending the course must be less than or equal to the number of seats in the room in which the lecture is allocated. Each additional student accounts for a violation.
  4. **Room Stability**: All lectures of a course must be allocated in the same room. Each distinct room attributed to the same course is counted as a violation.

The number of conflicts of each soft constraint is penalized with different weights. The objective function f of this problem is $f = \sum_{i=1,4} w_i \cdot c_i$ where w_i and c_i are the weight and total number of violations of soft constraint i, respectively.

3 An University Timetabling GRASP with Path Relinking Algorithm: UT-GRASP-PR

GRASP is a multi-start metaheuristic, introduced by [3], where the search for a solution evolves over a set of independent iterations. In each of them, an initial solution is built,

 © 2015 SBMAC

using a construction algorithm, and submitted to a local search procedure. GRASP output is the best overall solution.

The GRASP construction phase is a random adaptative greedy algorithm that generates a solution for the problem. In this paper, we propose such algorithm to yield a feasible solution to the timetabling problem, if possible, with few violations of soft constraints. The algorithm focuses only on eliminating violations of soft constraints.

In general, random algorithms are very suitable for achieving diversification, but the solutions are commonly poor. On the other hand, greedy algorithms focus on solutions of better quality, but they fail to diversificate. Our constructive algorithm starts from an empty timetable, adding lectures one by one, iteratively checking feasibility, until all are allocated. For this, a strategy is used: the most conflicting lectures should be assigned first [11].

In each iteration of constructive method, a candidate list (CL) is built containing all unallocated lectures. Let tm be the number of available timeslots to allocate the lecture $\bar{l}$, already discount timeslots that could generate infeasible solutions. Then, for each lecture $\bar{l}$, there are tm options of timeslots to allocate the lecture in the timetable. The lecture with less tm options is chosen to be allocated. The cost of allocating a lecture at all the tm timeslots is calculated taking into account the soft constraints. The CL is ordered according to this cost and a restricted candidate list (RCL) is built based on the lowest (c^{min}) and highest (c^{max}) costs. The RCL elements consists of the timeslot whose costs belong to $[c^{min}, c^{min} + \alpha(c^{max} - c^{min})]$, where $0 \leq \alpha \leq 1$. A timeslot is chosen randomly from RCL and the lecture is added to the solution.

In some cases, may be no timeslot to keep the feasibility solution when a lecture is selected to allocate. To tackle this problem, a procedure called explosion was implemented. It removes a lecture of the timetable previously allocated to increase the options of timeslots for the lecture that had the problem. To perform the explosion procedure, you must choose randomly a feasible timeslot and remove the allocated lecture, because this avoids cycling.

The proposed constructive algorithm is similar to that used in [10]. However, this algorithm had never been used as the construction phase of GRASP. The solution generated in the construction phase is given as initial to a local search procedure, searching for better solutions.

In this paper, Hill Climbing(HC) [4] and Simulated Annealing(SA) [7] were adopted as local search strategies. We use two different movements to guide the search of a solution into the neighborhood: **MOVE**, where a lecture is moved to a free timeslot; and **SWAP**, that changes two lectures positions in the timetable. These movements were chosen because they are simple and widely used in other works, especially by the ITC-2007 winner [10].

HC algorithm has two main input parameters: number of neighbors generated in each iteration (k) and the number of iterations without improvement in the solution (N). The SA algorithm has four main input parameters: initial (T_i) and final (T_f) temperatures, cooling rate (β) and the number of neighbors generated in each temperature (N_v).

PR was first introduced in the context of tabu search [5], as an approach to integrate intensification and diversification. It consists of exploring trajectories that connect

DOI: 10.5540/03.2015.003.02.0108 © 2015 SBMAC

high-quality solutions, by starting from an initial solution and generating a path in the neighborhood of this solution towards another solution, called the guiding solution. This path is generated by selecting movements that introduce in the initial solution attributes of the guiding solution. At each step, all movements that incorporate attributes of the guiding solution are analyzed and the movement that best improves (or least deteriorates) the initial solution is chosen. The PR maintains a set E of elite solutions (local optima). The procedure begins with a random selection of the guiding solution in E and the initial solution is the one returned from the local search procedure. Further details about this technique can be found in [5].

4 Computational Results and Concluding Remarks

The 21 instances used are those of the ITC-2007 [6]. After empirics tests, the UT-GRASP-PR parameters were set for the computational experiments: the RLC parameter $\alpha = 0.15$ and in the HC algorithm the number of iterations without improvement $N = 10000$. For the PR procedure, we limited the elite solutions set size $MaxElite = 20$. Each instance was run 10 times, with different seeds for the generation of random numbers. Two versions of the HC algorithm were tested, one version with $k = 1$ neighbors generated in each iteration and another with $k = 10$. Therefore, to evaluate the efficiency of local search types addressed in this paper, three versions of UT-GRASP-PR were tested:

- **GHC1**: Local Search HC with $k = 1$ (Only one neighbor is generated by iteration).
- **GHC10**: Local Search HC with $k = 10$ (10 neighbors are generated by iteration).
- **GSA**: Local Search SA, with $T_i = 1.5$, $T_f = 0.005$, $\beta = 0.999$ and $N_v = 500$.

The algorithm stop condition is a maximum computational time, stipulated in 324 seconds for the machine where the tests were performed, according to a time computing criterion available as an executable code provided by the ITC-2007 organization.

The three versions of UT-GRASP-PR were implemented in C and compiled using the GCC compiler (version 4.1.2) on a PC running Linux with Fedora Core 8 distribution and processor Intel Quad-Core with a 2.4GHz and 2GB of RAM.

Table 1 presents the best result for each instance in each version of the UT-GRASP-PR algorithm. In this Table, we can observe that the GSA version of UT-GRASP-PR algorithm, obtained the best results of the three versions described. GSA obtained better results in 19 instances and tying with GHC10 in 2 instances (comp01 and comp11). On average, GSA is better than GHC1 in 158% and GHC10 in 30%. This is explained because the SA algorithm can escape of solutions that are local optima. According to the analysis, the results of GSA version was chosen to be compared to the results of ITC-2007 competitors.

The ITC-2007 competition consisted of two phases: the first with participants applying different algorithms for the instances comp01 to comp14. The top five competitors were selected to the second phase, where they submitted their algorithms to another seven instances (comp15 to comp21).

DOI: 10.5540/03.2015.003.02.0108 © 2015 SBMAC

Table 1: Best results obtained by the three versions of the UT-GRASP-PR algorithm.

Instance	GHC1	GHC10	GSA	Instance	GHC1	GHC10	GSA
comp01	15	**5**	**5**	comp12	847	455	**375**
comp02	260	130	**73**	comp13	206	110	**97**
comp03	223	125	**98**	comp14	191	91	**72**
comp04	168	73	**48**	comp15	218	141	**101**
comp05	707	525	**409**	comp16	233	96	**69**
comp06	293	116	**75**	comp17	271	127	**105**
comp07	266	68	**36**	comp18	179	113	**102**
comp08	186	77	**58**	comp19	238	122	**87**
comp09	269	144	**119**	comp20	356	106	**88**
comp10	245	68	**41**	comp21	301	176	**136**
comp11	9	**0**	**0**	Average	270,52	136,57	**104,47**

Table 2 presents the average of results (first phase) over the first 14 instances obtained by the 17 competitors and average of results of UT-GRASP-PR in the GSA version. Analogously, Table 3 presents the average results of the second phase, which are presented in [9], together with the average of results of UT-GRASP-PR, also in the GSA version.

Table 2: Results of the First Phase of the ITC-2007.

1^{st}	2^{nd}	3^{rd}	**UT-GRASP-PR**	4^{th}	5^{th}
85.64	86.57	89.29	**107.57**	135.21	166.79
6^{th}	7^{th}	8^{th}	9^{th}	10^{th}	11^{th}
171.21	192.86	231.86	240.43	246.14	272.29
12^{th}	13^{th}	14^{th}	15^{th}	16^{th}	17^{th}
296.64	415.21	523.64	870.36	3562.29	132338.14

Table 3: Results of the Second Phase of the ITC-2007.

1^{st}	2^{nd}	3^{rd}	**UT-GRASP-PR**	4^{th}	5^{th}
68.00	69.71	78.71	**98.29**	116.86	127.00

According to Table 2, UT-GRASP-PR had an average performance 17% lower of the 3^{rd} competitor and 20% higher of the 4^{th} competitor, on the first phase of ITC-2007. In the Table 3, we note that the average performance of UT-GRASP-PR in the second phase of ITC-2007 is 25% lower of the 3^{rd} competitor and 16% higher of the 4^{th} competitor.

DOI: 10.5540/03.2015.003.02.0108 © 2015 SBMAC

We conclude that the UT-GRASP-PR (GSA version) results were competitive when comparing with most of the ITC-2007 competitors. As future work, we aim to investigate the use of other metaheuristics such as Biased Random Keys Genetic Algorithm (BRKGA), as well as other neighborhood structures for local search, like Kempe chain.

References

[1] V. Cacchiani, A. Caprara, R. Roberti and P. Toth, A new lower bound for curriculum-based course timetabling, Computers & Operations Research, vol. 40, 2466–2477, (2013).

[2] S. Ceschia, L. D. Gaspero and A. Schaerf, Design, engineering, and experimental analysis of a simulated annealing approach to the post enrolment course timetabling problem, Computers & Operations Research, vol. 39, 1615–1624, (2012).

[3] T. A. Feo and M. G. C. Resende, A probabilistic heuristic for a computationally difficult set covering problem, Operations Research Letters, vol. 8, 67–71, (1989).

[4] F. Glover, Tabu Search-Part I, ORSA Journal of Computing, 190–206, (1989).

[5] F. Glover, Tabu search and adaptive memory programming - advances, applications and challenges, In: R. S. Barr, R. V. Helgason and J. L. Kenngington editors. Interfaces in Computer Science and Operations Research, vol. 7, 1-75, (1996).

[6] PATAT, International Timetabling Competition, URL:http://www.cs.qub.ac.uk/itc2007, (2008).

[7] S. Kirkpatrick, Optimization by simulated annealing: Quantitative studies. Journal of Statistical Physics, Kluwer Academic Publishers Plenum Publishers, vol. 34, 975–986, (1984).

[8] R. Lewis, A survey of metaheuristic based techniques for university timetabling problems, OR Spectrum, vol. 30, 167–190, (2008).

[9] Z. P. Lu and J. K. Hao, Adaptive tabu search for course timetabling, European Journal Operation Research, vol. 200, 235–244, (2010).

[10] T. Müller, ITC2007 solver description: a hybrid approach, Annals of Operations Research, Springer US, vol. 172, 429–446, (2009).

[11] R. Raghavjee and N. Pillay, The Effect of Construction Heuristics on the Performance of a Genetic Algorithm for the School Timetabling Problem, Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment, 187–194, (2011).

[12] A. Schaerf, A survey of automated timetabling, Artificial Intelligence Review, vol. 13, 87–127, (1999).

DOI: 10.5540/03.2015.003.02.0108 © 2015 SBMAC