An Efficient Tabu Search Heuristic for the School Timetabling Problem

Haroldo G. Santos[1], Luiz S. Ochi[1], and Marcone J.F. Souza[2]

[1] Computing Institute, Fluminense Federal University, Niterói, Brazil
{hsantos,satoru}@ic.uff.br
[2] Computing Department, Ouro Preto Federal University, Ouro Preto, Brazil
marcone@iceb.ufop.br

Abstract. The School Timetabling Problem (STP) regards the weekly scheduling of encounters between teachers and classes. Since this scheduling must satisfy organizational, pedagogical and personal costs, this problem is recognized as a very difficult combinatorial optimization problem. This work presents a new Tabu Search (TS) heuristic for STP. Two different memory based diversification strategies are presented. Computational experiments with real world instances, comparing with a previously proposed TS found in the literature, show that the proposed method produces better solutions for all instances, as well faster times are observed in the production of good quality solutions.

1 Introduction

The School Timetabling Problem (STP) embraces the scheduling of sequential encounters between teachers and students so as to insure that requirements and constraints are satisfied. Typically, the manual solution of this problem extends for various days or weeks and normally produces unsatisfactory results due to the fact that lesson periods could be generated which are inconsistent with pedagogical needs or could even serve as impediments for certain teachers or students. STP is considered a NP-hard problem [5] for nearly all of its variants, justifying the usage of heuristic methods for its resolution. In this manner, various heuristic and metaheuristic approaches have been applied with success in the solution of this problem, such as: Tabu Search (TS) [12,4,10], Genetic Algorithms [13] and Simulated Annealing (SA) [2].

The application of TS to the STP is specially interesting, since this method is, as local search methods in general, very well suited for the interactive building of timetables, a much recognized quality in timetable building systems. Furthermore, TS based methods often offer the best know solutions to many timetabling problems, when compared to other metaheuristics [3,11]. The diversification strategy is an important aspect in the design of a TS algorithm. Since the use of a tabu list is not enough to prevent the search process from becoming trapped in certain regions of the search space, other mechanisms have been proposed. In particular, for the STP, two main approaches have been used: adaptive relaxation [10,4] and random restart [12]. In adaptive relaxation the costs involved in the objective function are dynamically changed to bias the search process to newly, unvisited, regions of the search space. In random restart a new solution is generated and no previous information is utilized.

This work employs a TS algorithm that uses an informed diversification strategy, which takes into account the history of the search process to bias the selection of diversification movements. Although it uses only standard Tabu Search components, it provides better results than more complex previous proposals [12].

The article is organized as follows: section 2 presents related works; section 3 introduces the problem to be treated; section 4 presents the proposed algorithm; section 5 describes the computational experiments and their results; and finally, section 6 formulates conclusions and future research proposals.

2 Related Works

Although the STP is a classical combinatorial optimization problem, no widely accepted model is used in the literature. The reason is that the characteristics of the problem are highly dependent on the educational system of the country and the type of institution involved. As such, although the basic search problem is the same, variations are introduced in different works [3,4,10,12]. Described afterwards, the problem considered in this paper derives from [12] and considers the timetabling problem encountered in typical Brazilian high schools. In [12], a GRASP-Tabu Search (`GTS-II`) metaheuristic was developed to tackle this problem. The `GTS-II` method incorporates a specialized improvement procedure named "Intraclasses-Interclasses", which uses a shortest-path graph algorithm. At first, the procedure is activated aiming to attain the feasibility of the constructed solution, after which, it then aims to improve the feasible solution. The movements made in the "Intraclasses-Interclasses" also remain with the tabu status for a given number of iterations. Diversification is implemented through the generation of new solutions, in the GRASP constructive phase. In [11] three different metaheuristics that incorporate the "Intraclasses-Interclasses" were proposed: Simulated Annealing, Microcanonical Optimization (MO) and Tabu Search. The TS proposal outperformed significantly both SA and MO.

3 The Problem Considered

The problem considered deals with the scheduling of encounters with teachers and students over a weekly period. The schedule is made up of d days of the week with h daily periods, defining $p = d \times h$ distinct periods. There is a set T with t teachers that teach a set S of s subjects to a set C of c classes, which are disjoint sets of students with the same curriculum. The association of teachers to subjects in certain classes is previously fixed and the workload is informed in a matrix of requirements $R_{t\times c}$, where r_{ij} indicates the number of lessons that teacher i shall teach for class j. Classes are always available, and must have their time schedules, of size p, completely filled out, while teachers indicate a set of available periods. Also, teachers may request a number of double lessons per class. These lessons are lessons which must be allocated in two consecutive periods on the same day. This way a solution to the STP problem must satisfy the following constraints:

1. no class or teacher can be allocated for two lessons in the same period;
2. teachers can only be allocated respecting their availabilities;
3. each teacher must fulfill his/her weekly number of lessons;
4. for pedagogical reasons no class can have more than two lesson periods with the same teacher per day.

Also, there are the following desirable features that a timetable should present:

1. the time schedule for each teacher should include the least number possible of days;
2. double lessons requests must be satisfied whenever possible;
3. "gaps" in the time schedule of teachers should be avoided, that is: periods of no activity between two lesson periods.

3.1 Solution Representation

A timetable is represented as a matrix $Q_{t\times p}$, in a such way that each row represents the complete weekly timetable for a given teacher. As such, the value $q_{ik} \in \{0, 1, \cdots, c\}$, indicates the class for which the teacher i is teaching during period k ($q_{ik} \in \{1, \cdots, c\}$), or if the teacher is available for allocation ($q_{ik} = 0$). The advantage of this representation is that it eliminates the possibility for the occurrence of conflicts in the timetable for teachers. The occurrence of conflicts in classes happens when in a given period k more than one teacher is allocated to that class. Allocations are only allowed in periods with availability of teachers. A partial sample of a timetable with 5 teachers can be found in Figure 1, with value "X" indicating the unavailabilities of teachers.

Teacher \ Period	1	2	3	4	5	$\cdots d \times h$
1	1	0	0	2	2	$\cdots$
2	0	X	X	0	1	$\cdots$
3	X	X	1	0	3	$\cdots$
4	0	1	0	1	0	$\cdots$
5	0	0	2	3	X	$\cdots$

Fig. 1. Fragment of generated timetable

3.2 Objective Function

In order to treat STP as an optimization problem, it is necessary to define an objective function that determines the degree of infeasibility and satisfaction of requirements; that is, pretends to generate feasible solutions with minimal number of unsatisfied requisites. Thus, a timetable Q is evaluated with the following objective function, which should be minimized:

$$f(Q) = \omega \times f_1(Q) + \delta \times f_2(Q) + \rho \times f_3(Q) \tag{1}$$

where f_1 counts, for each period k, the number of times that more than one teacher teaches the same class in period k and the number of times that a class has no activity in k. The f_2 portion measures the number of allocations that disregard the daily limits of lessons of subjects in classes (constraint 4). As such, the timetable can only be considered feasible if $f_1(Q) = f_2(Q) = 0$. The importance of the costs involved defines a hierarchy so that: $\omega > \delta \gg \rho$. The f_3 component in the objective function measures the satisfaction of personal requests from teachers, namely: double lessons, non existence of "gaps" and timetable compactness, as follows:

$$f_3(Q) = \sum_{i=1}^{t} \alpha_i \times b_i + \beta_i \times v_i + \gamma_i \times c_i \tag{2}$$

where α_i, β_i, and γ_i are weights that reflect, respectively, the relative importance of the number of "gaps" b_i , the number of week days v_i each teacher is involved in any teaching activity during the same shift, and the non negative difference c_i between the minimum required number of double lessons and the effective number of double lessons in the current agenda of teacher i.

4 Tabu Search for the School Timetabling Problem

Tabu Search (TS) is an iterative method for solving combinatorial optimization problems which explicitly makes use of memory structures to guide a hill-descending heuristic to continue exploration without being confounded by the absence of improvement movements. The "modern version" of TS was independently proposed by Glover [6] and Hansen [8]. This section presents a brief explanation of the tabu search principles followed by an specification of the tabu search custom implementation proposed in this paper. For a detailed description of the TS method, the reader is referred to [7].

Starting from an initial solution x, the method systematically explores the neighborhood $\mathcal{N}(x)$ and selects the best admissible movement m, such that the application of m in the current solution x (denoted by $x \oplus m$) produces the new current solution $x' \in \mathcal{N}(x)$. When no improvement movements are found, movements that deteriorate the cost function are also permitted. Thus, to avoid cycling, a mechanism called short-term memory is employed. The objective of short-term memory is try to forbid movements towards already visited solutions, which usually is done through the prohibition of the last reversal movements. These movements are stored in a tabu list and remain forbid (with tabu status), for a given number of iterations, called *tabu tenure*. Since this strategy can be too restrictive, to not disregard high quality solutions, movements with tabu status can be accepted if the cost of the new solution produced satisfy an *aspiration criterion*. Also, intensification and diversification procedures can be used. These procedures, respectively, aim to deeply investigate promising regions of the search space and to ensure that no region of the search space remain neglected. Follows a description of the constructive algorithm and the custom tabu search implementation proposed in this paper.

4.1 Constructive Algorithm

The constructive algorithm basically consists of a greedy randomized constructive procedure [9]. While in other works the option for a randomized construction is to allow diversification, through the re-start of the process, in this case the purpose is only to have control of the randomization degree of the initial solution. To build a solution, step-by-step, the principle of allocating first the *most urgent* lessons in the *most appropriate* periods is used. In this case, the urgency degree θ_{ij} of allocating a lesson from teacher i for class j is computed considering the available periods V_i from teacher i, the available periods W_j from class j and the number of unscheduled lessons ζ_{ij} of teacher i for class j, as follows: $\theta_{ij} = \frac{\zeta_{ij}}{|V_i \cap W_j|+1}$. Also, let L be the set of required lessons, such that $r_{ij} > 0, \forall l_{ij} \in L$. The algorithm then builds a restricted candidate list (RCL) with the most urgent lessons, in a such a way that $RCL = \{l_{ij} \in L \mid \theta_{ij} \geq \overline{\theta} - (\overline{\theta} - \underline{\theta}) \times \alpha\}$, where $\overline{\theta} = \max\{\theta_{ij} \mid i \in T,\ j \in C\}$ and $\underline{\theta} = \min\{\theta_{ij} \mid i \in T,\ j \in C\}$. At each iteration, a lesson l_{ij} for allocation is randomly selected from RCL. The lecture is allocate in a teacher free period, attempting to maintain the timetable free of conflicts in classes. Allocations are made first in periods with least teacher availability. The α parameter allows tuning the randomization degree of the algorithm, varying from the pure greedy algorithm ($\alpha = 0$) to a completely random ($\alpha = 1$) selection of the teacher and class to allocation. At each step, the number of unscheduled lessons and the urgency degrees are recomputed. The process continue till no more unscheduled lessons are found (i.e., $\zeta_{ij} = 0, \forall i \in T, j \in C$).

4.2 Tabu Search Components

The TS procedure (Figure 2) starts from the initial timetable Q provided by the constructive algorithm and, at each iteration, fully explores the neighborhood $N(Q)$ to select the next movement m. The movement definition used here is the same as in [10], and consists in the swap of two values in the timetable of a teacher $i \in T$, which can be defined as the triple $\langle i, p_1, p_2 \rangle$, such that $q_{ip_1} \neq q_{ip_2}$, $p_1 < p_2$ and $p_1, p_2 \in \{1, \cdots, p\}$. Clearly, any timetable can be reached through a sequence of these movements that is at most, the number of lessons in the

```
procedure TSDS($Q, divActivation, iterationsDiv, minTT, maxTT$)
 1: $Q^* = Q$; $TabuList = \emptyset$;
 2: $noImprovementIterations = 0$; $iteration = 0$;
 3: $initializeMovementeFrequencies()$;
 4: repeat
 5:    $\Delta = \infty$; $iteration++$;
 6:    for all movement $m$ such that $(Q \oplus m) \in \mathcal{N}(Q)$ do
 7:       $penalty = 0$;
 8:       if $noImprovementIterations$ mod $divActivation < iterationsDiv$
          and $iteration \geq noImprovementIterations$ then
 9:          $penalty = computePenalty(m)$;
10:       end if
11:       if $(f(Q) - f(Q \oplus m) + penalty < \Delta$ and $(m \notin TabuList))$ or $(f(Q \oplus m) <$
          $f(Q^*))$ then
12:          $bestMov = m$;
13:          $\Delta = f(Q) - f(Q \oplus m)$;
14:       end if
15:    end for
16:    $Q = Q \oplus m$;
17:    $tabuTenure(m) = random(minTT, maxTT)$;
18:    $updateTabuList(m, iteration)$;
19:    $computeMovementFrequency(m)$;
20:    if $(f(Q) < f(Q^*))$ then
21:       $Q^* = Q$; $noImprovementIterations = 0$;
22:       $initializeMovementeFrequencies()$;
23:    else
24:       $noImprovementIterations$++;
25:    end if
26: until termination criterion reached;
end TSDS;
```

Fig. 2. Pseudo-code for TSDS algorithm

requirement matrix. Once a movements m is selected, its reversal movement will be kept in the tabu list along the next $tabuTenure(m)$ iterations, which is randomly selected (line 17) within the interval $[minTT, maxTT]$. The aspiration criterion defined is that the movement will loose its tabu status if its application produces the best solution found so far (line 11).

Since short-term memory is not enough to prevent the search process from become entrenched in certain regions of the search space, some diversification strategy is needed. In the proposed method, long-term memory is used to guide the diversification procedure in the following way: frequencies of movements involving each teacher and class are computed. While the diversification procedure is active, the selection of movements emphasizes the execution of few explored movements, through the incorporation of penalties (line 9) in the evaluation of movements. Each time a movement is done, movement frequencies shall be updated (line 19). These frequencies are zeroed each time that the best solution found so far is updated (line 22). Follows an explanation of how the penalties in function *computePenalty*() (line 9) are computed. Considering that the counts of movements made with each teacher and class are stored in a matrix $Z_{t\times c}$, the penalty for a given movement takes into account the transition ratio ϵ_{ij} of teacher i and class j, which can be computed as follows:

$$\epsilon_{ij} = \frac{z_{ij}}{\overline{z}} \tag{3}$$

where $\overline{z} = \max\{z_{ij} \mid i \in T,\ j \in C\}$. Since a movement can involve two lesson periods, or a lesson period and a free period, the penalty $\psi_{ia_1a_2}$ associated with a movement in the timetable of teacher i, in periods p_1 and p_2 with allocations $a_1 = q_{ip_1}$ and $a_2 = q_{ip_2}$, respectively, considering the cost of the best solution found so far Q^* is:

$$\psi_{ia_1a_2} = \begin{cases} \epsilon_{ia_1} \times f(Q^*) & \text{if } a_1 \neq 0 \text{ and } a_2 = 0 \\ \epsilon_{ia_2} \times f(Q^*) & \text{if } a_1 = 0 \text{ and } a_2 \neq 0 \\ (\epsilon_{ia_1} + \epsilon_{ia_2})/2 \times f(Q^*) & \text{if } a_1 \neq 0 \text{ and } a_2 \neq 0 \end{cases}$$

Another penalty function proposed in this paper considers also the teacher workload to promote diversification. In this case, the objective is to favor movements involving teachers whose timetable changes would probably produce bigger modifications in the solution structure. The value of the penalty function $\tau_{ia_1a_2}$ for allocations a_1 and a_2 of teacher i is:

$$\tau_{ia_1a_2} = \frac{\psi_{ia_1a_2}}{\sum_{j=1}^{c} r_{ij} / \max_{l=1}^{t} \left(\sum_{j=1}^{c} r_{lj}\right)} \tag{4}$$

The diversification strategy is applied whenever signals that regional entrenchment may be in action are detected. In this case, the number of non-improvement iterations is evaluated before starting the diversification strategy (line 8). The number of non-improvement iterations necessary to start the diversification (*activationDiv*) process and the number of iterations that the process will remain active (*iterationsDiv*) are input parameters. Movements performed in this phase can be viewed as *influential movements* [7], in a way that these movements try to modify the solution structure in a influential (non-random) way. The function *computePenalty* (line 9) can use one of the penalty functions previously presented. In the following sections, the implementation that considers the penalty function that only takes into account the frequency ratio of transitions will be referred as `TSDS`, while the implementation that use the penalty function that takes into account also the workload of teachers will be referred as `TSDSTL`. For comparison purposes, an implementation without the diversification strategy (`TS`), also will be considered in the next section.

5 Computational Experiments and Discussion

Experiments were done in the set of instances originated from [12], and the data referred to Brazilian high schools, with 25 lesson periods per week for each class,

Instance	Teachers	Classes	Total Lessons	Double Lessons	Sparseness Ratio (sr)
1	8	3	75	21	0.43
2	14	6	150	29	0.50
3	16	8	200	4	0.30
4	23	12	300	66	0.18
5	31	13	325	71	0.58
6	30	14	350	63	0.52
7	33	20	500	84	0.39

Table 1. Characteristics of problem instances

in different shifts. In Table 1 some of the characteristics of the instances can be verified, such as dimension and sparseness ratio (sr), which can be computed considering the total number of lessons ($\#lessons$) and the total number of unavailable periods (u): $sr = \frac{t \times p - (\#lessons + u)}{t \times p}$. Lower sparseness values indicate more restrictive problems and likewise, more difficult resolution.

The algorithms were coded in C++. The implementation of `GTS-II` was the same presented in [12], and was implemented in C. The compiler used was GCC 3.2.3 using flag `-O2`. The experiments were performed in a micro-computer with an AMD Athlon XP 1533 MHz processor, 512 megabytes of RAM running the Linux operating system.

The weights in the objective function were defined as in [12]: $\omega = 100$, $\delta = 30$, $\rho = 1$, $\alpha_i = 3$, $\beta_i = 3$ and $\gamma_i = 1$, $\forall i = 1, \cdots, t$.

In the first set of experiments, the objective was to verify the average solution cost produced by each algorithm, within a given time limit. The results (Table 2) consider the average best solution found in 20 independent executions, with the following time limits to instances $1, \cdots, 7$, respectively: {90, 280, 380, 870, 1930, 1650, 2650}. The parameters for `GTS-II` and the time limits are the same proposed in [12]. The parameters for `TSDS` and its variations are: $\alpha = 0.1$ (constructive algorithm), $minTT = 20$, $maxTT = 25, activationDiv = 500$ and $iterationsDiv = 10$. Best results are shown in bold.

Instance	`GTS-II`	`TSDSTL`	`TSDS`	`TS`
1	204.80	203.42	**203.37**	207.05
2	350.10	**344.84**	345.36	349.26
3	455.70	439.94	**439.05**	455.58
4	686.30	**669.69**	672.15	670.92
5	796.30	782.74	**780.74**	782.84
6	799.10	783.38	**781.77**	787.85
7	1,076.20	1,060.84	**1,059.05**	1,071.21

Table 2. Average results with fixed time limits

Constructive Algorithm					TSDS				
$f_1(Q^*)$	$f_2(Q^*)$	$\#d$ ($\%d$)	$\#g$ ($\%g$)	cr	$f_1(Q^*)$	$f_2(Q^*)$	$\#d$ ($\%d$)	$\#g$($\%g$)	cr
0.0	0.5	15.1 (71.5)	17.2 (22.9)	1.6	0.0	0.0	2.0 (9.5)	4.0 (5.3)	1.2
0.0	0.0	24.3 (83.8)	24.8 (16.5)	1.3	0.0	0.0	8.2 (28.3)	1.2 (0.8)	1.0
0.3	2.5	2.0 (50.0)	31.2 (15.6)	1.4	0.0	0.0	0.4 (8.8)	5.0 (2.5)	1.1
4.3	0.9	35.5 (53.8)	21.0 (7.0)	1.2	0.0	0.0	19.9 (30.2)	3.7 (1.2)	1.0
0.0	0.2	54.1 (76.1)	46.4 (14.3)	1.5	0.0	0.0	13.6 (19.2)	4.1 (1.2)	1.1
0.2	0.0	53.7 (85.2)	53.4 (15.3)	1.4	0.0	0.0	13.5 (21.4)	9.9 (2.8)	1.0
0.5	0.2	69.6 (82.9)	74.1 (14.8)	1.3	0.0	0.0	24.4 (29.0)	10.7 (2,1)	1.0

Table 3. Average costs of objective function components obtained by the constructive algorithm and at the end of the tabu search heuristic TSDS

As it can be seen in Table 2, although only minor differences can be observed among the two implementations that use different penalty functions in the diversification strategy, results show that versions that use the informed diversification strategy perform significantly better than GTS-II and TS. In order to evaluate the quality of the solutions obtained by the proposed method, and to verify how significant is the improvement of TSDS over the solution received from the constructive algorithm, Table 3 presents the average costs involved in each objective function component, for the solution provided by the constructive algorithm and for the improved solution from TSDS. Columns $\#d$ ($\%d$), $\#g$ ($\%g$) and cr are related to the f_3 component of the objective function, in the following way: $\#d$ ($\%d$) indicates the unsatisfied double lessons (and the percentage of unsatisfied double lessons, considering the number of double lesson requests), $\#g$ ($\%g$) indicates the number of "gaps" in the timetable of teachers (and the percentage considering the total number of lessons) and cr measures the compactness ratio of the timetable of teachers. To compute cr, the actual number of days ad that teachers must attend to some lesson in the school the the lower bound for this value $\underline{ad}$ are used. The $\underline{ad}$ value considers the minimum number of days $md_i = \lceil \frac{\sum_{j=1}^{c} cr_{ij}}{days} \rceil$ that each teacher i must attend some lecture in the school, such that $\underline{ad} = \sum_{i=1}^{t} md_i$. This way, $cr = ad/\underline{ad}$. Values near to one indicate that the timetable is as compact as it can be. As it can be seen in Table 3, the solution provided by the constructive algorithm usually contains some type of infeasibility. These problems were always solved by the TSDS algorithm, in a way that no infeasible timetable was produced. Regarding the preferences of teachers, the timetable compactness, which has the highest weight in the f_3 component of the objective function, it can be seen that in most cases the optimal value was reached ($cr = 1$). Also, small percentage values of "gaps" and unsatisfied double lessons were reached.

In other set of experiments, the objective was to verify the empirical probability distribution of reaching a given sub-optimal target value (i.e. find a solution with cost at least as good as the target value) in function of time in different instances. The sub-optimal values were chosen in a way that the slowest algorithm could terminate in a reasonable amount of time. In these experiments, TSDSTL and GTS-II were evaluated and the execution times of 150 independent runs for each instance were computed. The experiment design follows the proposal of [1]. The results of each algorithm were plotted associating with the i-th smallest running time t_i a probability $p_i = (i - \frac{1}{2})/150$, which generates points $z_i = (t_i, p_i)$, for $i = 1, \cdots, 150$. As it can be be seen in Figures 3 to 6 the TSDSTL heuristic achieves high probability values ($\geq 50\%$) of reaching the target values in significantly smaller times than GTS-II. This difference is enhanced mainly in instance 4, which presents a very low sparseness ratio. This result may be related to the fact that the "Intraclasses-Interclasses" procedure of GTS-II works with movements that use free periods, which are hard to find in this instance. Another analysis show that at the time when 95% of TSDSTL runs have achieved the target value, in average, only 64% of GTS-II runs have achieved the target value. Considering the time when 50% of TSDSTL runs have achieved the target value, only 11%, in average, of GTS-II runs have achieved the target value.

6 Concluding Remarks

This paper presented a new tabu search heuristic to solve the school timetabling problem. Experiments in real world instances showed that the proposed method outperforms significantly a previously developed hybrid tabu search algorithm, and it has the advantage of a simpler design.

Contributions of this paper include the empirical verification that although informed diversification strategies are not commonly employed in tabu search implementations for the school timetabling problem, its incorporation can significantly improve the method robustness. The proposed method not only produced better solutions for all test instances but also performed faster than a hybrid tabu search approach.

Although the proposed method offers quite an improvement, future researches may combine the "Intraclasses-Interclasses" procedure with an informed diversification strategy, which could lend to even better results .

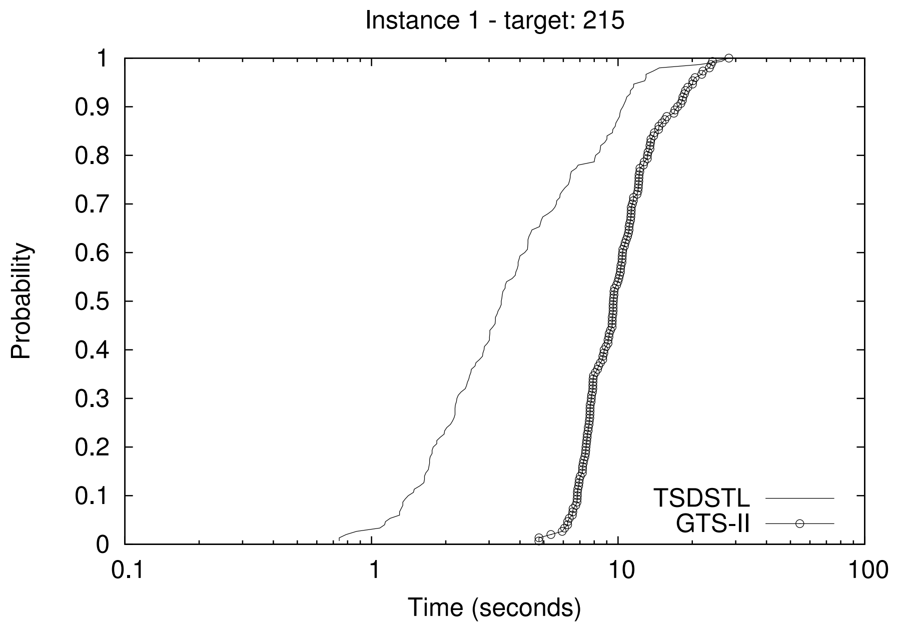
Acknowledgements

This work was partially supported by CAPES. The authors would like thank Olinto C. B. Araújo, from DENSIS-FEE-UNICAMP, Brazil for their valuable comments on the preparation of this paper.

References

1. Aiex, R. M., Resende, M. G. C., Ribeiro, C. C.: Probability distribuition of solution time in GRASP: an experimental investigation, Journal of Heuristics, **8** (2002), 343–373

2. Abramson, D.: Constructing school timetables using simulated annealing: sequential and parallel algorithms. Management Science. **37** (1991) 98–113.
3. Colorni, A., Dorigo, M., Maniezzo, V.: Metaheuristics for High-School Timetabling. Computational Optimization and Applications. **9** (1998) 277–298.
4. Costa, D.: A Tabu Search algorithm for computing an operational timetable. European Journal of Operational Research Society. **76** (1994) 98–110.
5. Even, S., Itai, A., Shamir, A.: On the complexity of timetabling and multicommodity flow problems. SIAM Journal of Computation. **5** (1976) 691–703.
6. Glover, F.: Future paths for integer programming and artificial intelligence. Computers & Operations Research. **13** (1986) 533–549.
7. Glover, F., Laguna, M.: Tabu Search. Kluwer Academic Publishers, Boston Dordrecht London (1997)
8. Hansen, P.: The steepest ascent mildest descent heuristic for combinatorial programming. Congress on Numerical Methods in Combinatorial Optimization. Capri (1986)
9. Resende, M.G.C., Ribeiro. C.C.: Greedy randomized adaptive search procedures. Handbook of Metaheuristics. Kluwer. (2003) 219–249
10. Schaerf, A.: Tabu search techniques for large high-school timetabling problems. Report CS-R9611. Centrum voor Wiskunde en Informatica, Amsterdam (1996)
11. Souza, M.J.F.: Programação de Horários em Escolas: Uma Aproximação por Metaheurísticas, D.Sc. Thesis (in Portuguese), Universidade Federal do Rio de Janeiro - Rio de Janeiro (2000)
12. Souza, M.J.F., Ochi, L.S., Maculan, N.: A GRASP-Tabu search algorithm for solving school timetabling problems. In: Resende, M.G.C., Souza, J.P. (eds.): Metaheuristics: Computer Decision-Making. Kluwer Academic Publishers, Boston (2003) 659–672
13. Wilke, P, Gröbner, M., Oster, N.: A hybrid genetic algorithm for school timetabling. In: AI 2002: McKay B. and Slaney J. (eds.): Advances in Artificial Intelligence. Springer Lecture Notes in Computer Science, Vol. 2557. Springer-Verlag, New York (2002) 455–464



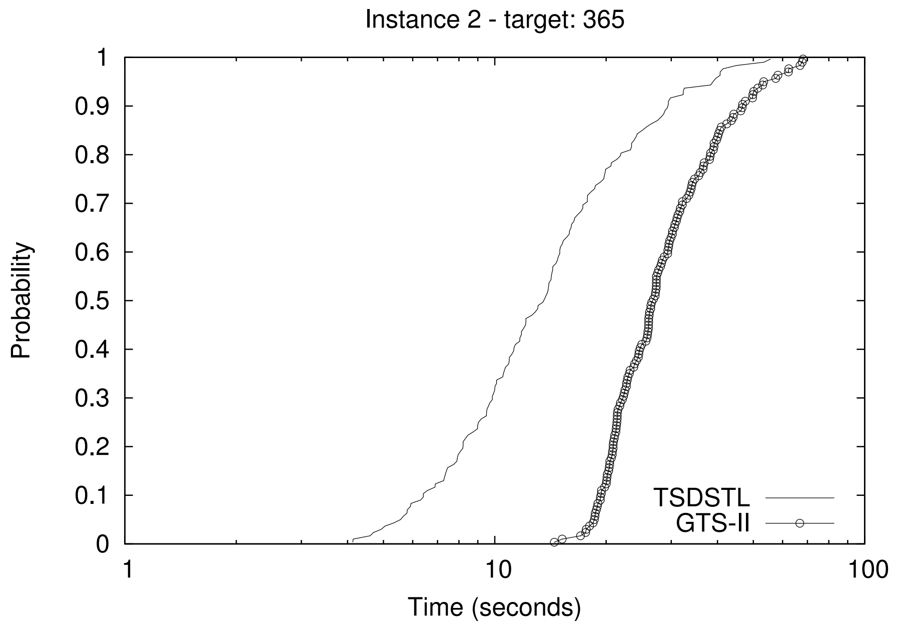
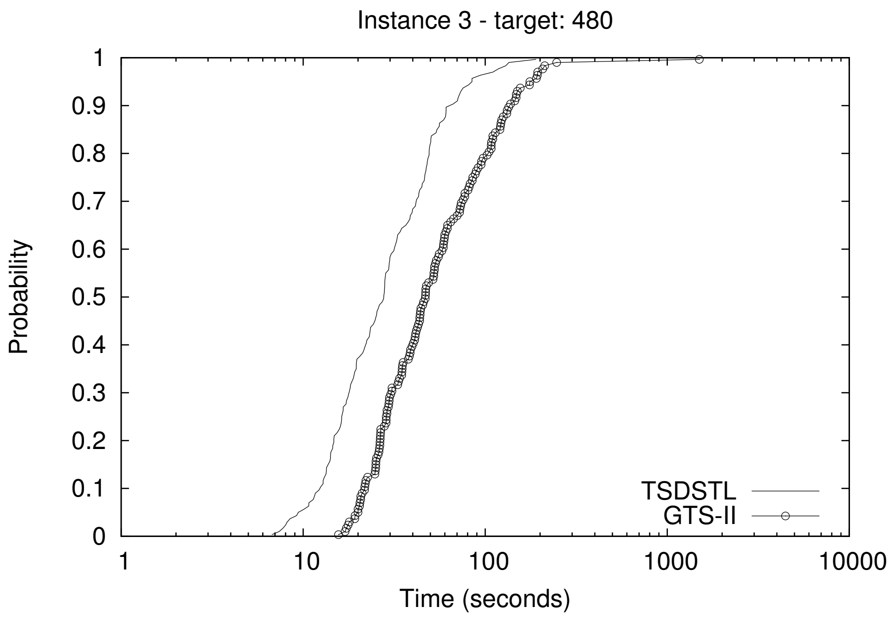


Fig. 3. Empirical probability distribution of finding target values in function of time for instances 1 and 2



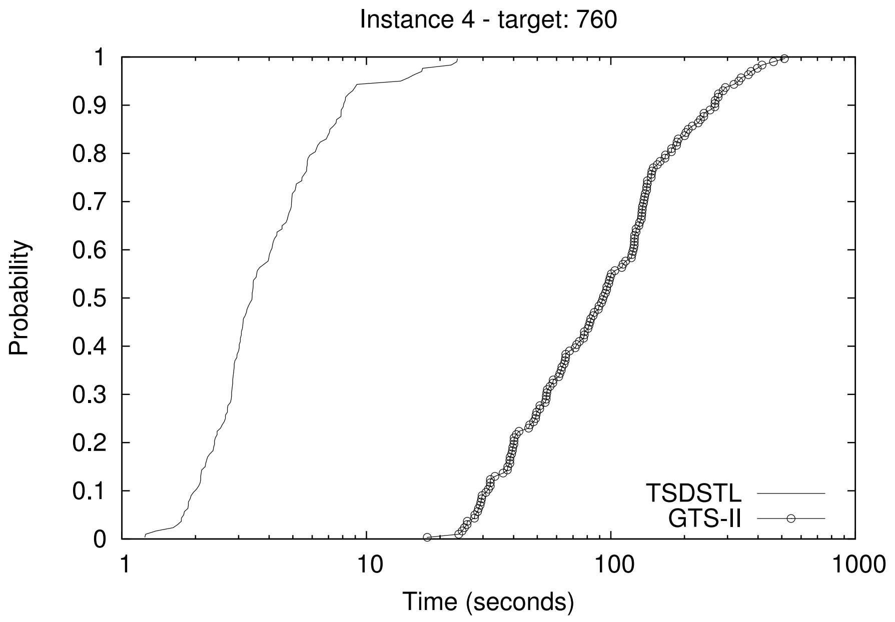
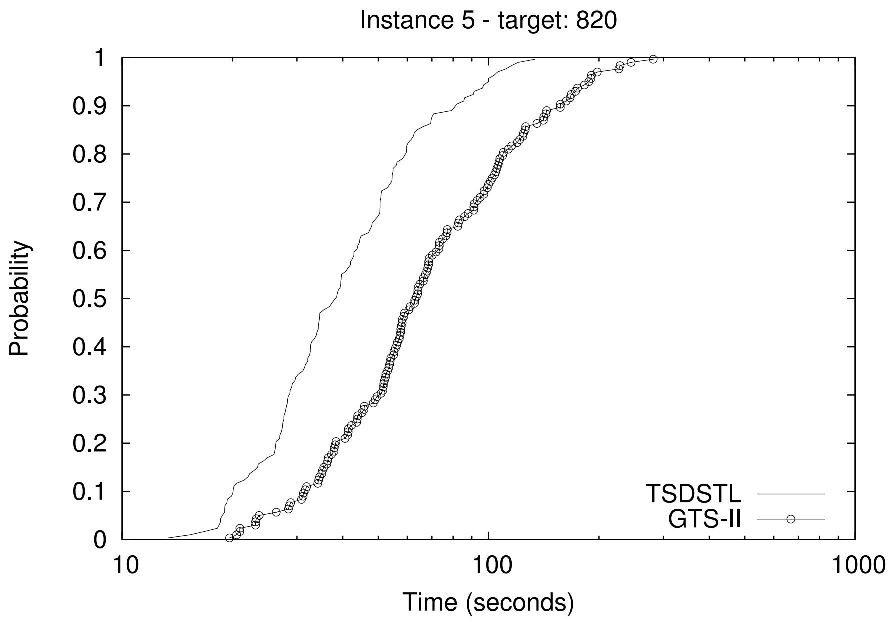


Fig. 4. Empirical probability distribution of finding target values in function of time for instances 3 and 4



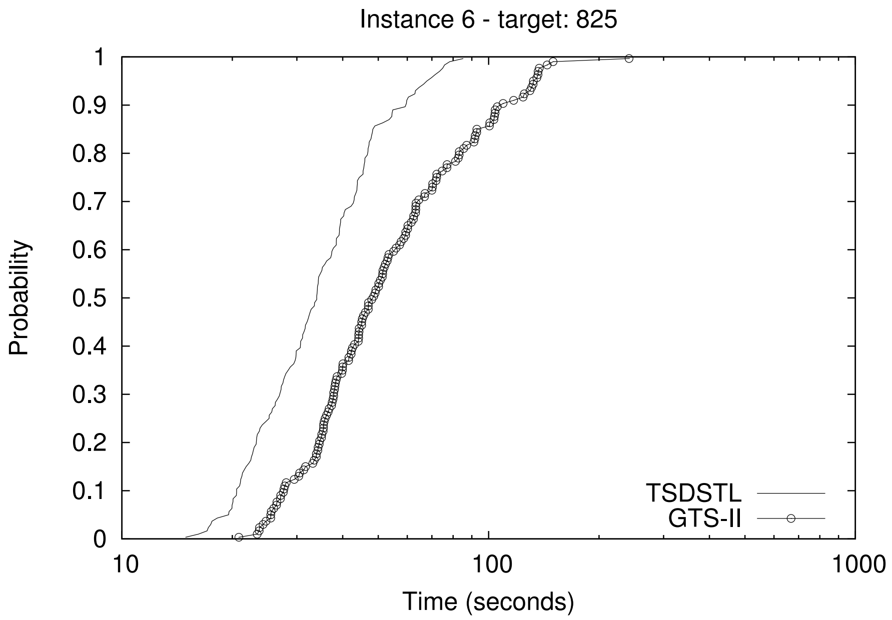


Fig. 5. Empirical probability distribution of finding target values in function of time for instances 5 and 6

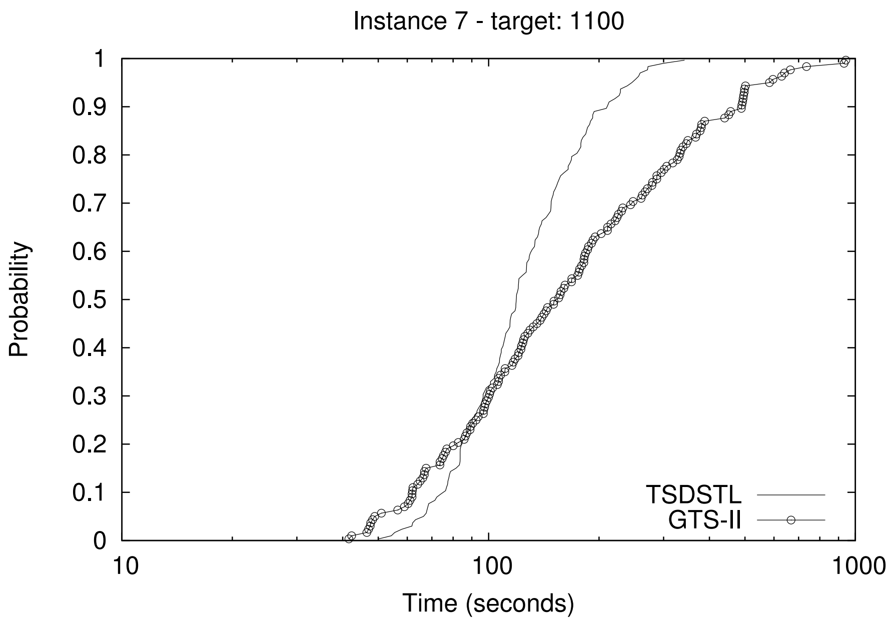


Fig. 6. Empirical probability distribution of finding target value in function of time for instance 7