Noname manuscript No.
(will be inserted by the editor)

GOAL Solver: A Hybrid Local Search Based Solver for High School Timetabling

George H.G. Fonseca · Haroldo G. Santos · Túlio A.M. Toffolo · Samuel S. Brito · Marcone J.F. Souza

Received: date / Accepted: date

Abstract This work presents a local search approach to the High School Timetabling Problem. The addressed timetabling model is the one stated in the Third International Timetabling Competition (ITC 2011), which considered many instances from educational institutions around the world and attracted seventeen competitors. Our team, named GOAL (Group of Optimization and Algorithms), developed a solver built upon the Kingston High School Timetabling Engine. Several neighborhood structures were developed and used in a hybrid metaheuristic based on Simulated Annealing and Iterated Local Search. The developed algorithm was the winner of the competition and produced the best known solutions for almost all instances.

Keywords Third International Timetabling Competition · High School Timetabling Problem · Simulated Annealing · Iterated Local Search · Metaheuristics

George Henrique Godim da Fonseca
Computing and Information Systems Department, Federal University of Ouro Preto, Brazil
E-mail: george@decea.ufop.br

Haroldo Gambini Santos
Computer Science Department, Federal University of Ouro Preto, Brazil
E-mail: haroldo@iceb.ufop.br

Túlio Ângelo Machado Toffolo
Computer Science Department, Federal University of Ouro Preto, Brazil
E-mail: tulio@toffolo.com.br

Samuel Souza Brito
Computer Science Department, Federal University of Ouro Preto, Brazil
E-mail: samuel.ufop@gmail.com

Marcone Jamilson Freitas Souza
Computer Science Department, Federal University of Ouro Preto, Brazil
E-mail: marcone@iceb.ufop.br

1 Introduction

The High School Timetabling Problem, also denoted as the Class×Teacher Timetabling Problem, consists of the assignment of timeslots and resources (e.g. rooms) to teachers×class meetings. These meetings are are also called events or lessons. The assignments must be scheduled in such a way that no teacher or class attends more than one event at the same time. Many other constraints can be considered, like limited availability of teachers and pre-allocations.

The automated construction of high school schedules has been the subject of much research in Computer Science and Operations Research. Surveys Schaerf (1999) and Pillay (2013) present some reasons for this interest:

hardness to solve: finding a timetable that satisfies the interest of all involved elements is a hard task; moreover, often the simple construction of a valid timetable is already a very complicated task;

practical importance: a good timetable can improve the staff satisfaction and allow the school to be more efficient in managing its resources; moreover a good schedule can improve the students performance;

theoretical importance: the problem addressed is classified as $\mathcal{NP}$-Hard (Even et al, 1976) and progress in solving such problems is a major goal in Computer Science and Mathematics.

Exact methods based in Integer Programming were proposed by Santos et al (2012), but they can only deal with a subset of instances in restricted processing times. Nowadays, metaheuristic approaches are commonly applied to the problem (Muller, 2009; Lú and Hao, 2010; Souza et al, 2003; Santos et al, 2005).

The diversity of School Timetabling models encountered around the world (Schaerf, 1999; Kingston, 2013; Kristiansen and Stidsen, 2013) motivated the definition of an XML format to express different entities and constraints considered when building timetables. The format evolved and a public repository[1] with a rich set of instances was built (Post et al, 2011). To stimulate the research in this area, the Third International Timetabling Competition (ITC 2011) was organized. The event took place on the Internet and the results were announced in Norway on 2012. This paper presents the algorithms of the solver that won the competition.

The remaining of this work is organized as follows: in Section 2 the model of High School Timetabling Problem proposed by the ITC 2011 is presented. Section 3 presents the competition itself. In Section 4 the solution approach developed is detailed. Section 5 shows computational experiments and, finally, concluding remarks are presented in Section 6.

[1] `http://www.utwente.nl/ctit/hstt/`, accessed on 30th April, 2013.

2 High School Timetabling Problem

It is well known that formulations of High School timetabling in different institutions are usually quite diverse in terms of decisions and constraints. Some papers describe problems commonly found in different parts of the world: Australia (Kingston, 2005), Brazil (Santos et al, 2012), England (Wright, 1996), Finland (Nurmi and Kyngas, 2007), Greece (Valourix and Housos, 2003) and The Netherlands (de Haan et al, 2007). Thus, to precisely formulate a representative set of problems described in literature, a domain specific language, XHSTT (Post et al, 2010, 2011), was used to encode problem instances in ITC 2011. XHSTT is powerful enough to specify problems with different sets of constraints, objective functions and decision variables.

The model in XHSTT is split in three main entities: (1) Time and Resources, (2) Events and (3) Constraints. A solution consists of a set of assignments of times and resources to the events.

2.1 Times and Resources

The time entity consists of a timeslot or a set of timeslots (time group). The resources are divided in three categories: students, teachers and rooms (Post et al, 2010):

class: a group of students attends to an event (lesson); important constraints associated with students are the control of their idle times and the number of lessons scheduled per day;
teachers: the schedule of teachers must be built considering workload limits and qualification constraints, while respecting pre-assignments;
rooms: most events take place on a room, and each room has a capacity and a set of available features.

2.2 Events

An event is the basic unit of assignment, representing a simple *lesson* or a set of lessons (event group). We refer to an assignment of a timeslot to an event as a *meet*, and to an assignment of a resource to an event as a *task*. The term *course* is used to designate a group of students who attend to the same events. Other kinds of events, like meetings, are also allowed by the model (Post et al, 2010). An event has the following attributes:

duration: represents the number of timeslots which have to be assigned to the event;
pre-assigned resources: some resources can be pre-assigned to attend the event (optional);
workload: amount of workload that the event will contribute to its resources total workload (optional);

pre-assigned timeslots: some events have pre-assigned timeslots and therefore must be assigned to that specific timeslot (optional).

2.3 Constraints

Post et al (2010) groups the constraints in three categories: basic constraints of scheduling, constraints of events and constraints of resources. The objective function $f(.)$ is calculated in terms of the violations of the constraints. Each violation is penalized according to its weight (so this is a minimization problem). The constraints are also divided in hard constraints, whose satisfaction is mandatory; and soft constraints, whose satisfaction is desirable but not obligatory. Each instance can define whether a constraint is hard or soft.

2.3.1 Basic Constraints of Scheduling

1. Assign Time: assigns timeslots to each event;
2. Assign Resource: assigns the resources to each event;
3. Prefer Times: indicates that some events have preference for a particular timeslot(s);
4. Prefer Resources: indicates that some event have preference for a particular resource(s).

2.3.2 Constraints to Events

1. Link Events: to schedule a set of events in the same starting time;
2. Spread Events: specifies the allowed number of occurrences for event groups in time groups between a minimum a maximum number of times; this constraint can be used, for example, to define a daily limit of lessons;
3. Avoid Split Assignments: for each event, assigns a particular resource to all of its meets;
4. Distribute Split Events: for each event, assigns between a minimum and a maximum number of meets of a given duration;
5. Split Events: places limits on the number of non-consecutive meets created for an event and their duration.

2.3.3 Constraints to Resources

1. Avoid Clashes: to assign resources without clashes (i.e. without assigning the same resource to more than one event per timeslot);
2. Avoid Unavailable Times: states that certain resources are unavailable to attend any events at certain times;
3. Limit Workload Constraint: restricts the workload of the resources between a minimum and a maximum bound;

4. Limit Idle Times: the number of idle times in each time group must lie between a minimum and a maximum bound for each resource; typically, a time group consists of all timeslots of a given week day;
5. Limit Busy Times: the number of busy times in each time group should lie between a minimum and a maximum bound for each resource;
6. Cluster Busy Times: forces the allocations of activities of a given resource to be grouped in specific timegroups; this can be used, for example, to avoid excessive spreading of teacher's activities in the whole timetable.

3 Third International Timetabling Competition

Following the first and the second International Timetabling Competition (ITC 2002[2] and ITC 2007[3], respectively), the third edition (ITC 2011[4]) aimed at stimulating the timetabling research. More specifically, it aimed to encourage the alignment of research with practice by offering real-world instances of timetabling problems to the research community. The competition was composed by three separate rounds. Post et al (2013) presents detailed information about the competition.

3.1 Competition Rounds

The first round focused on generating the best solutions to a public instance set. In this round, solvers could be executed without time restrictions. Also, competitors could use any available technology to improve the solutions.

In the second round, organizers ran the solvers on the same conditions for 1,000 seconds using a hidden instance set. The solvers sent for evaluation on this round should respect rules about the inclusion of proprietary software libraries. The best five teams were classified to the second round. To classify, the competitors had to run their solvers on the public instance set and send the obtained solutions to the organizers. Ten independent executions were considered for each instance. Each solution for each instance from all solvers were compared and ranked from 1 (best) to 5 (worst), according to their quality. The team with the smallest average rank was claimed the winner of this round.

In the third round, competitors ran their solvers in the hidden instance set (that became public) and sent their solutions to the organizers. Like in the first round, time and technology restrictions were not imposed to the solvers. The solutions from all competitors were compared and ranked from 1 (best) to 5 (worst). Again, the team with the smaller average rank was claimed winner of the round.

[2] `http://www.idsia.ch/Files/ttcomp2002/`, accessed on 30th April, 2013.
[3] `http://www.cs.qub.ac.uk/itc2007/`, accessed on 30th April, 2013.
[4] `http://www.utwente.nl/ctit/hstt/itc2011/welcome/`, accessed on 30th April, 2013.

The competition presented a wide range of instances, but some of them were provided by competitors. To make the competition fair, results obtained by the competitors on their own instances were not considered for the ranking.

4 Solution Approach

Our approach employed the Kingston High School Timetabling Engine (KHE) school timetabling engine (Kingston, 2012) as a platform to efficiently manage instances and solutions. One of its best features is the incremental cost recalculation, which speeds up the computation of cost changes when modifying solutions using our local search methods. We also used KHE to quickly produce an initial solution. Following, Simulated Annealing is used to improve this solution. Finally, Iterated Local Search is applied to further polish the solution. Figure 1 presents a summary of our approach. Its elements are explained in the following subsections.

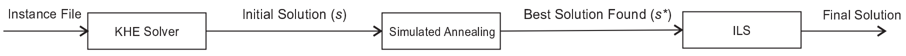


Fig. 1 Solution Approach Scheme

4.1 Constructive Algorithm

KHE is a platform for handling instances of the addressed problem. It also provides a routine to quickly build an initial solution, called `KheGeneralSolve`. We used this routine because it produces a solution quickly even for harder instances (see Table 5). As expected, for harder instances these solutions are usually infeasible and very poor in terms of cost. This construction method is based on the concept of Hierarchical Timetabling (Kingston, 2006), where smaller allocations are combined to generate bigger blocks of allocations until a complete solution is produced. Hierarchical Timetabling is supported by the Layer Tree data structure. It consists of nodes that represent the required meet and task allocations. An allocation may appear in at most one node. A layer is a subset of nodes with the property that none of them can be overlapped in time. Commonly, nodes are grouped in a layer when they share resources.

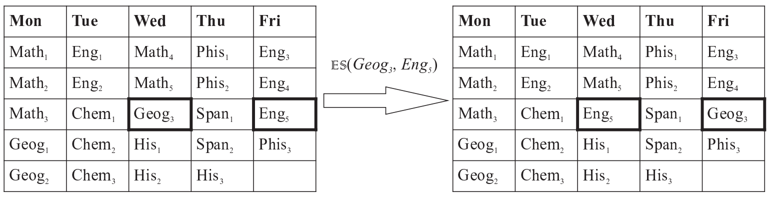
The hard constraints of the problem are modeled by this data structure and then a matching problem is solved to assign times and resources to event allocations. The matching is solved by connecting each node to a timeslot or resource respecting the property of layer. For further details, see Kingston (2006, 2012).

4.2 Neighborhood Structure

The neighborhood $N(s)$ of a solution s is defined as the set of all solutions which can be reached using one of our move types. The neighborhood considering one specific move type k is denoted by $N_k(s)$. The following subsections present each one of the seven move types used, as well as the procedures which explore these neighborhoods. Moves described in subsections 4.2.1 to 4.2.5 are intend to quickly perform small changes in the solution and correspond to operations in KHE. Moves *Kempe Chain* (subsection 4.2.6) and *Reassign Resource Times* (subsection 4.2.7) perform larger changes and were implemented from scratch.

*4.2.1 Event Swap (*ES*)*

In this move, two lessons l_1 and l_2 are selected and have their timeslots t_1 and t_2 swapped. Figure 2 presents an example of this move, where the timeslots of lessons Geog_3 and Eng_5 are swapped.



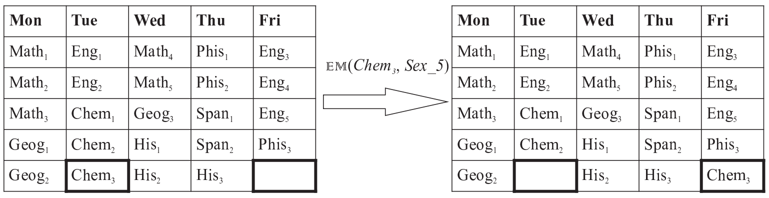
Mon	Tue	Wed	Thu	Fri
Math_1	Eng_1	Math_4	Phis_1	Eng_3
Math_2	Eng_2	Math_5	Phis_2	Eng_4
Math_3	Chem_1	Geog_3	Span_1	Eng_5
Geog_1	Chem_2	His_1	Span_2	Phis_3
Geog_2	Chem_3	His_2	His_3	

Mon	Tue	Wed	Thu	Fri
Math_1	Eng_1	Math_4	Phis_1	Eng_3
Math_2	Eng_2	Math_5	Phis_2	Eng_4
Math_3	Chem_1	Eng_5	Span_1	Geog_3
Geog_1	Chem_2	His_1	Span_2	Phis_3
Geog_2	Chem_3	His_2	His_3	

Fig. 2 Example of an Event Swap

*4.2.2 Event Move (*EM*)*

In this move, a lesson l_1 is selected and moved from its original timeslot t_1 to a new timeslot t_2. Figure 3 presents an example of this move, in which lesson Chem_3 moves from the last timeslot of tuesday to the last timeslot of friday.



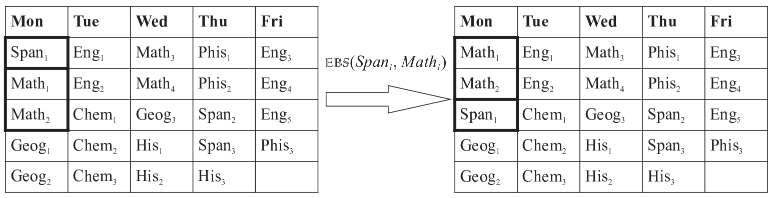
Mon	Tue	Wed	Thu	Fri
Math_1	Eng_1	Math_4	Phis_1	Eng_3
Math_2	Eng_2	Math_5	Phis_2	Eng_4
Math_3	Chem_1	Geog_3	Span_1	Eng_5
Geog_1	Chem_2	His_1	Span_2	Phis_3
Geog_2	Chem_3	His_2	His_3	

Mon	Tue	Wed	Thu	Fri
Math_1	Eng_1	Math_4	Phis_1	Eng_3
Math_2	Eng_2	Math_5	Phis_2	Eng_4
Math_3	Chem_1	Geog_3	Span_1	Eng_5
Geog_1	Chem_2	His_1	Span_2	Phis_3
Geog_2		His_2	His_3	Chem_3

Fig. 3 Example of an Event Move

*4.2.3 Event Block Swap (*EBS*)*

Similarly to the move ES, the move EBS changes the timeslot of two lessons l_1 and l_2. However, if involved lessons are in adjacent timeslots and have different durations, l_1 is moved to the timeslot immediately after the last timeslot assigned to l_2. This move allows changes in the timetable without losing the contiguity of allocations. Figure 4 presents an example of this move, where the positions of Span$_1$ and Math$_1$ are swapped.



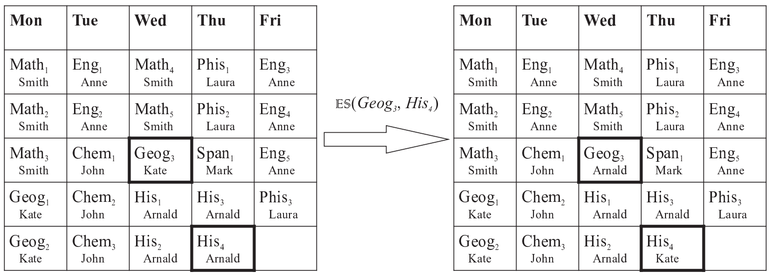
Mon	Tue	Wed	Thu	Fri
Span$_1$	Eng$_1$	Math$_3$	Phis$_1$	Eng$_3$
Math$_1$	Eng$_2$	Math$_4$	Phis$_2$	Eng$_4$
Math$_2$	Chem$_1$	Geog$_3$	Span$_2$	Eng$_5$
Geog$_1$	Chem$_2$	His$_1$	Span$_3$	Phis$_3$
Geog$_2$	Chem$_3$	His$_2$	His$_3$	

Mon	Tue	Wed	Thu	Fri
Math$_1$	Eng$_1$	Math$_3$	Phis$_1$	Eng$_3$
Math$_2$	Eng$_2$	Math$_4$	Phis$_2$	Eng$_4$
Span$_1$	Chem$_1$	Geog$_3$	Span$_2$	Eng$_5$
Geog$_1$	Chem$_2$	His$_1$	Span$_3$	Phis$_3$
Geog$_2$	Chem$_3$	His$_2$	His$_3$	

Fig. 4 Example of an Event Block Swap

*4.2.4 Resource Swap (*RS*)*

In this move, two lessons l_1 and l_2 have their resources r_1 and r_2 of a specific role changed. Figure 5 presents an example of this move, where lessons Geog$_3$ and His$_4$ have their teachers, Kate and Arnald, swapped.



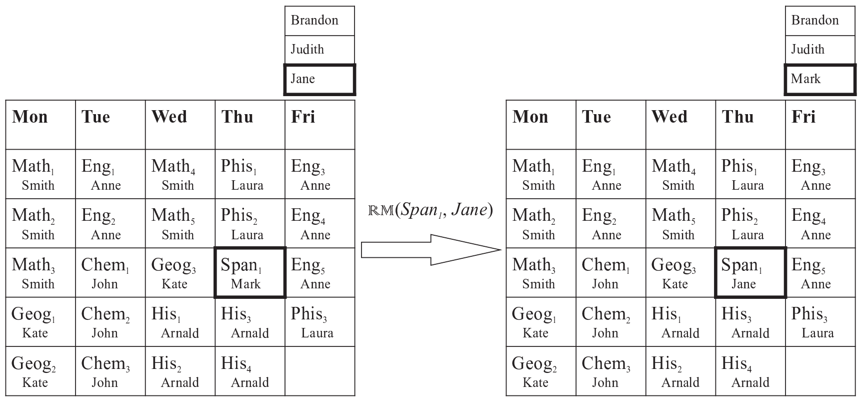
Mon	Tue	Wed	Thu	Fri
Math$_1$ Smith	Eng$_1$ Anne	Math$_4$ Smith	Phis$_1$ Laura	Eng$_3$ Anne
Math$_2$ Smith	Eng$_2$ Anne	Math$_5$ Smith	Phis$_2$ Laura	Eng$_4$ Anne
Math$_3$ Smith	Chem$_1$ John	Geog$_3$ Kate	Span$_1$ Mark	Eng$_5$ Anne
Geog$_1$ Kate	Chem$_2$ John	His$_1$ Arnald	His$_3$ Arnald	Phis$_3$ Laura
Geog$_2$ Kate	Chem$_3$ John	His$_2$ Arnald	His$_4$ Arnald	

Mon	Tue	Wed	Thu	Fri
Math$_1$ Smith	Eng$_1$ Anne	Math$_4$ Smith	Phis$_1$ Laura	Eng$_3$ Anne
Math$_2$ Smith	Eng$_2$ Anne	Math$_5$ Smith	Phis$_2$ Laura	Eng$_4$ Anne
Math$_3$ Smith	Chem$_1$ John	Geog$_3$ Arnald	Span$_1$ Mark	Eng$_5$ Anne
Geog$_1$ Kate	Chem$_2$ John	His$_1$ Arnald	His$_3$ Arnald	Phis$_3$ Laura
Geog$_2$ Kate	Chem$_3$ John	His$_2$ Arnald	His$_4$ Kate	

Fig. 5 Example of a Resource Swap

*4.2.5 Resource Move (*RM*)*

In this move a lesson l_1 is assigned to a new resource r_2 instead of the previously assigned resource r_1. Figure 6 presents an example of this move, where lesson Span$_1$ had its teacher changed from Mark to Jane.



Brandon
Judith
Jane

Mon	Tue	Wed	Thu	Fri
Math$_1$ Smith	Eng$_1$ Anne	Math$_4$ Smith	Phis$_1$ Laura	Eng$_3$ Anne
Math$_2$ Smith	Eng$_2$ Anne	Math$_5$ Smith	Phis$_2$ Laura	Eng$_4$ Anne
Math$_3$ Smith	Chem$_1$ John	Geog$_3$ Kate	Span$_1$ Mark	Eng$_5$ Anne
Geog$_1$ Kate	Chem$_2$ John	His$_1$ Arnald	His$_3$ Arnald	Phis$_3$ Laura
Geog$_2$ Kate	Chem$_3$ John	His$_2$ Arnald	His$_4$ Arnald	

Brandon
Judith
Mark

Mon	Tue	Wed	Thu	Fri
Math$_1$ Smith	Eng$_1$ Anne	Math$_4$ Smith	Phis$_1$ Laura	Eng$_3$ Anne
Math$_2$ Smith	Eng$_2$ Anne	Math$_5$ Smith	Phis$_2$ Laura	Eng$_4$ Anne
Math$_3$ Smith	Chem$_1$ John	Geog$_3$ Kate	Span$_1$ Jane	Eng$_5$ Anne
Geog$_1$ Kate	Chem$_2$ John	His$_1$ Arnald	His$_3$ Arnald	Phis$_3$ Laura
Geog$_2$ Kate	Chem$_3$ John	His$_2$ Arnald	His$_4$ Arnald	

Fig. 6 Example of a Resource Move

*4.2.6 Kempe Chain (*KC*)*

The application of the previously presented moves is likely to produce infeasible solutions, specially in very constrained instances. Consider for example moving only one meeting of a teacher which has a full schedule to a different timeslot. In this case the selection of any other timeslot will produce a conflict. The resolution of conflicts may require a *chain* of moves where each conflict produced by a move is solved with another move. We implemented a method which searches for these chains of moves involving two timeslots. Our implementation was inspired by the Kempe Chain Interchanges (KCI), proposed by Johnson et al (1991) to the vertex coloring problem.

The search for the best the chain of moves in timeslots t_1 and t_2 starts with the creation of an undirected bipartite conflict graph $G = (V, E)$. The set of vertices $V = V_1 \cup V_2$ for the first (V_1) and second (V_2) partitions are created for meetings which are, respectively, in timeslots t_1 and t_2. The set E of edges is created by linking vertices from V_1 to V_2 which share some resource (e.g. meetings of the same class or which use the same room). Since our method does not assumes that the current solution is feasible (differently from the KCI implementation), conflicts among vertices in the same partition may exist but are ignored in order to keep the graph bipartite.

For each vertex $v_i \in V_1$ we execute a depth-first search (DFS) in this conflict graph and compute the longest path starting at v_i, denoted here as P_i. In order to precisely evaluate the final result in terms of solution cost of applying the sequence of moves corresponding to P_i, a new solution with all these moves applied is generated and evaluated. The method continues for all vertices of the first partition and returns the best of these long chains found. We opted to consider only large chains so that this move has also an important role in diversification.

We present an example in Figure 7 where the conflict graph for timeslots Mon_2 and Wed_4 is shown at left (assuming that all lessons of a subject are taught by the same teacher). In this example there are two maximal paths starting with meeting Eng$_4$: Eng$_4$$\rightarrow$ Eng$_3$$\rightarrow$ Phis$_3$ and Eng$_4$$\rightarrow$ Span$_4$$\rightarrow$ Span$_5$ $\rightarrow$ Math$_5$. The latter one is selected as the path to be evaluated for vertex Eng_4, since it is the largest one starting at it. If this is the best path from all evaluated in this conflict graph then the corresponding chain of moves is selected and the resulting solution is presented at the right side.

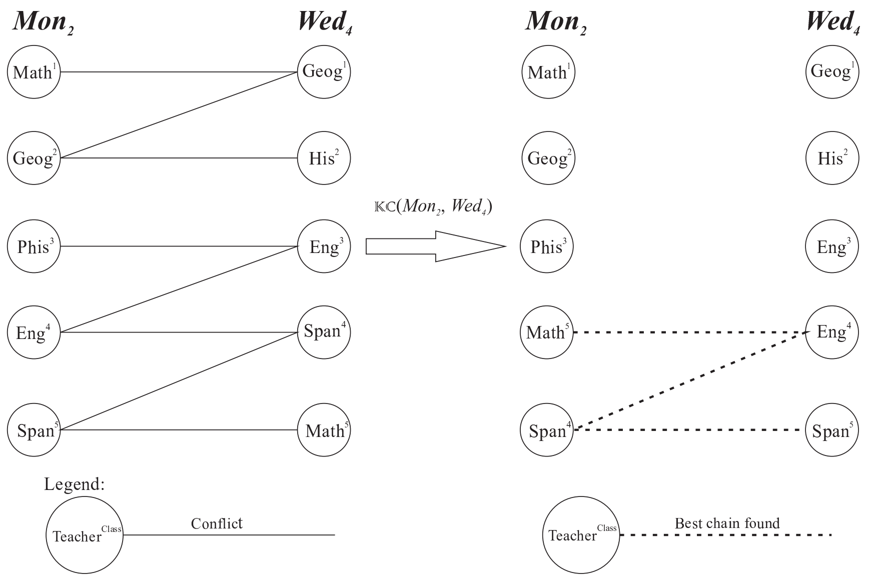


Fig. 7 Example of a Kempe Chain

*4.2.7 Reassign Resource Times (*RRT*)*

In this move, a resource r_1 and the events that use this resource are selected. Each permutation of the timeslots of these events is a neighbor in this neighborhood. All possible permutations are analyzed and the best one - the one that incurs the biggest decrease/smallest increase in the solution - is returned. Our implementation limits the number of events in this neighborhood, for each resource, since there are $n!$ possible permutations for a set of n elements. The size of n is limited such that $n \leq 7$ ($n! \leq 5040$). If there are more than seven events related to a resource, we take only the first seven events.

Figure 8 shows an example of this move. In this example Adam (resource of type teacher) is considered. Five new solutions are produced from the permutations of the events associated with Adam. The best one of these permutations, highlighted in bold, is selected.

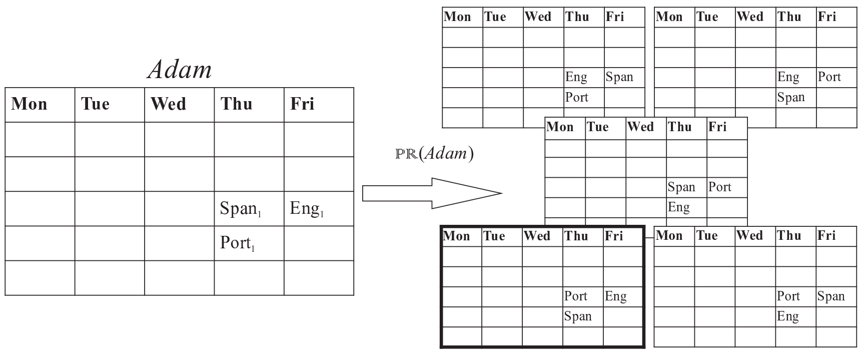


Fig. 8 Example of a Reassign Resource Times

4.2.8 Neighborhood Exploration

Due to the large size of $N(s)$, which comprises every solution which can be reached by the application of any of the previously presented move types, $N(s)$ is explored by sampling. After performing an initial set of experiments, we observed that better results were produced when the exploration of some neighborhoods was more extensive than others. Also, some move types can be disabled for some instances, which is the specific case of instances which do not have resource assignment tasks. Thus, if the instance requires some resource assignment (i.e. has an AssignResourceConstraint constraint), the probabilities of selecting each one of the move types are defined as: ES = 20%, EM = 38%, EBS = 10%, RS = 20%, RM = 10% and KC = 2%. Otherwise, the moves RS and RM can be ignored and the following probabilities were used: ES = 40%, EM = 38%, EBS = 20% and KC = 2%. The move RRT is used only in the perturbation phase of the Iterated Local Search algorithm.

4.3 Simulated Annealing Implementation

Proposed by Kirkpatrick et al (1983), the metaheuristic Simulated Annealing (SA) is a probabilistic method based on an analogy of thermodynamics simulating the cooling of a set of heated atoms. This technique starts its search from an initial solution and improvements are made by local search. The main procedure consists of a loop that randomly generates, at each iteration, one neighbor s' of the current solution s by applying one moment.

Each move has an associated cost variation, denoted here as Δ ($\Delta = f(s') - f(s)$). Improvement and sideway moves, i.e., those with $\Delta \leq 0$, are unconditionally accepted. If $\Delta > 0$, the neighbor can also be accepted, but in this case, with a probability of $e^{-\Delta/T}$, where T is a parameter called temperature. The temperature regulates the probability of accepting worse solutions. The higher the value of T, the higher the chance of accepting a worse solution. Thus, the exploratory behavior of the search can be controlled by setting T to high values (emphasis on diversification) or low values (emphasis on intensification).

A common setup is to assign initially a high value T_0. After a fixed number of iterations SA_{max}, which represents the number of iterations needed for the system to reach the thermal equilibrium at a given temperature, the temperature is lowered by a cooling rate $\alpha \in (0, 1]$, such that for a given iteration k, $T_k \leftarrow \alpha \times T_{k-1}$. With this procedure, a greater chance of escaping from local optima occurs at the initial iteration and as T approaches zero, the algorithm behaves like a descent method (Gendreau and Potvin, 2010; Kirkpatrick et al, 1983). When the system reaches the thermal equilibrium ($T \to 0$), the algorithm ends. It is also possible to reheat the system to continue the exploration. The implemented algorithm in this work reheats the system up to $SA_{reheats}$ times.

The developed implementation of Simulated Annealing is described in Algorithm 1.

Algorithm 1: Developed implementation of Simulated Annealing

```
Input: $f(.), N(.), \alpha, SAmax, T_0, SAreheats, s, timeout$
Output: Best solution $s^*$ found.
$s^* \leftarrow s$; $IterT \leftarrow 0$; $T \leftarrow T_0$; $reheats \leftarrow 0$;
while $reheats < SAreheats$ and $elapsedTime < timeout$ do
    while $IterT < SAmax$ do
        $IterT \leftarrow IterT + 1$;
        $k \leftarrow selectNeighborhood()$;
        Generate a random neighbor $s' \in N_k(s)$;
        $\Delta = f(s') - f(s)$;
        if $\Delta < 0$ then
            $s \leftarrow s'$;
            if $f(s') < f(s^*)$ then $s^* \leftarrow s'$;
        else
            Take a random $x$, $x \in [0, 1]$;
            if $x < e^{-\Delta/T}$ then $s \leftarrow s'$;
    $T \leftarrow \alpha \times T$;
    $IterT \leftarrow 0$;
    if $T < 0.1$ then
        $reheats \leftarrow reheats + 1$;
        $T \leftarrow T_0$
return $s^*$;
```

4.4 Iterated Local Search Implementation

The Iterated Local Search (ILS) method, proposed by Lourenco et al (2003), is based on the idea that a local search procedure can achieve better results by optimizing different solutions generated through perturbations on the local optimum solution.

The ILS algorithm implemented in this work starts from an initial solution s_0, obtained by the Simulated Annealing procedure. Then, it makes perturbations of size p_{size} on s_0, creating a new solution s'. After that, a descent method is applied on s'. A perturbation consists of the unconditional acceptance of a neighbor generated by neighborhoods RRT or KC, each one with 50% of probability.

The descent phase uses a Randomic Non-Ascendant Method, which accepts only neighbors if they are better than or match the current solution. Differently from our SA implementation, where the composition of $N(s)$ could change based on probabilities, in this phase we systematically enumerate all neighbors in $N(s)$, changing only its exploration order.

The local search runs until ILS_{lsMax} iterations without improvement are reached. It produces a solution s' which will be accepted if it is better than the best solution s^* found. In such case, the perturbation size p_{size} gets back to the initial size p_0. If the iteration $Iter$ reaches a limit ILS_{max}, the perturbation size is increased. Yet if the perturbation size reaches a bound of p_{max}, it is reset to the initial size p_0. This process is repeated until the timeout is reached. Algorithm 2 presents the developed implementation of ILS.

Algorithm 2: Developed implementation of ILS

```
Input: $f(.), N(.), ILS_{max}, ILS_{lsMax}, , p_0, p_{max}, s, timeout$
Output: Best solution $s^*$ found.
$s \leftarrow descentPhase(s, ILS_{lsMax})$; $s^* \leftarrow s$;
$p_{size} \leftarrow p_0$; $Iter \leftarrow 0$;
while timeout not reached do
    for $j \leftarrow 0$ until $p_{size}$ do
        $s \leftarrow$ random neighbor $s_p \in N_{\mathrm{KC}}(s) \cup N_{\mathrm{RRT}}(s)$;
    $s' \leftarrow descentPhase(s, ILS_{lsMax})$;
    if $f(s') < f(s^*)$ then
        $s \leftarrow s'$; $s^* \leftarrow s'$;
        $Iter \leftarrow 0$; $p_{size} \leftarrow p_0$;
    else
        $s \leftarrow s^*$;
        $Iter \leftarrow Iter + 1$;
    if $Iter = ILS_{max}$ then
        $Iter \leftarrow 0$;
        $p_{size} \leftarrow (p_{size} + p_0)$ mod $p_{max}$;
return $s^*$;
```

5 Computational Experiments

In this section we report the computational experiments with our solver. At first we focus in the results obtained in ITC 2011. These results are detailed in subsections 5.1 and 5.2. Following, in subsection 5.3, we provide an additional set of post competition experiments: these include an extensive parameter tuning considering the recently released new set of instances XHSTT-2013.

All experiments ran on an Intel® i5 2.4 Ghz computer with 4GB of RAM running Linux Ubuntu 11.10 operating system. The algorithms were coded on C++ and compiled with GCC 4.6.1. Our results were validated by the HSEval validator[5]. The presented results are expressed by pairs x/y, where x contains the sum of feasibility penalties and y the sum of quality penalties.

Our solver, as well as our solutions and reports can be found in our website: `http://www.goal.ufop.br/hstt`. We invite the interested readers to experiment and possibly even improve our solver.

5.1 Datasets from Third ITC

The set of instances[6] available from ITC2011 was originated from many countries and ranges from small instances to huge, challenging ones. Table 1 presents the main features of the public instances and Table 2 presents the main features of the hidden instances.

Table 1 Features of public instances from ITC2011

Instance	Timeslots	Teachers	Rooms	Classes	Lessons
AustraliaBGHS98	40	56	45	30	1564
AustraliaSAHS96	60	43	36	20	1876
AustraliaTES99	30	37	26	13	806
BrazilInstance1	25	8	-	3	75
BrazilInstance4	25	23	-	12	300
BrazilInstance5	25	31	-	13	325
BrazilInstance6	25	30	-	14	350
BrazilInstance7	25	33	-	20	500
EnglandStPaul	27	68	67	67	1227
FinlandArtificialSchool	20	22	12	13	200
FinlandCollege	40	46	34	31	854
FinlandHighSchool	35	18	13	10	297
SecondarySchool	35	25	25	14	306
GreeceHighSchool1	35	29	-	66	372
GreecePatras3rdHS2010	35	29	-	84	340
GreecePreveza3rdHS2008	35	29	-	68	340
ItalyInstance1	36	13	-	3	133
NetherlandsGEPRO	44	132	80	44	2675
NetherlandsKottenpark2003	38	75	41	18	1203
NetherlandsKottenpark2005	37	78	42	26	1272
SouthAfricaLewitt2009	148	19	2	16	838

[5] `http://sydney.edu.au/engineering/it/~jeff/hseval.cgi`, accessed on April, 2013.
[6] `http://www.utwente.nl/ctit/hstt/archives/XHSTT-2012`, accessed on April, 2013.

Table 2 Features of hidden instances from ITC2011

Instance	**Timeslots**	**Teachers**	**Rooms**	**Classes**	**Lessons**
BrazilInstance2	25	14	-	6	150
BrazilInstance3	25	16	-	8	200
BrazilInstance4	25	23	-	12	300
BrazilInstance6	25	30	-	14	350
FinlandElementarySchool	35	22	21	291	445
FinlandSecondarySchool2	40	22	21	469	566
Aigio1stHighSchool2010-2011	35	37	-	208	532
Italy_Instance4	36	61	-	38	1101
KosovaInstance1	62	101	-	63	1912
Kottenpark2003	38	75	41	18	1203
Kottenpark2005A	37	78	42	26	1272
Kottenpark2008	40	81	11	34	1118
Kottenpark2009	38	93	53	48	1301
Woodlands2009	42	40	-	-	1353
Spanishschool	35	66	4	21	439
WesternGreeceUniversity3	35	19	-	6	210
WesternGreeceUniversity4	35	19	-	12	262
WesternGreeceUniversity5	35	18	-	6	184

5.2 Competition Results

The solver sent to the competition was previously tuned with the following parameters for Simulated Annealing: $SAmax = 10000$, $T_0 = 5$, $\alpha = 0.5$ and $SA_{reheats} = 5$. Iterated Local Search used: $ILS_{lsMax} = 10,000$, $ILS_{max} = 50$, $p_0 = 1$ and $p_{max} = 10$. Note that Simulated Annealing runs until it reaches five reheats and the remaining time, if any, is dedicated to the ILS algorithm.

5.2.1 First Round

In the first round of ITC 2011 solvers had no time limit to run nor technology restrictions. In this round we executed our solver considering as initial solution the best known solutions stored in the XHSTT archives [7] and set an 1000 seconds timeout for our solver. Whenever an improvement was achieved, we ran the solver again, taking the improved solution as input. Figure 9 presents how we ran our solver in this round.

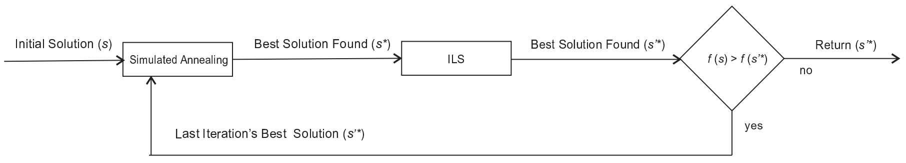


Fig. 9 Solver adaption to the first round of ITC 2011

[7] On instances *AustraliaSAHS96* and *AustraliaTES99*, the initial solution generated by KHE was better than the provided solution, so KHE was used in these cases.

Table 3 presents the results obtained by our solver in the first round, where ITC 2011 $f(s^*)$ denotes the provided solution at the beginning of the competition and GOAL $f(s^*)$ denotes the improved solution produced by our solver. Table 4 presents the winners for each instance of round 1.

Table 3 Results obtained with GOAL solver in the first round of ITC 2011

Instance	**ITC 2011** $f(s^*)$	**GOAL** $f(s^*)$
AustraliaBGHS98	7 / 433	**4 / 367**
AustraliaSAHS96	23 / 44	**10/ 12**
AustraliaTES99	26 / 134	**5 / 148**
BrazilInstance1	0 / 24	**0 / 15**
BrazilInstance4	0 / 112	**0 / 103**
BrazilInstance5	0 / 225	**0 / 198**
BrazilInstance6	0 / 209	**0 / 156**
BrazilInstance7	0 / 330	**0 / 294**
EnglandStPaul	0 / 18444	**0 / 11732**
FinlandArtificialSchool	0 / 0	0 / 0
FinlandCollege	0 / 0	0 / 0
FinlandHighSchool	0 / 1	0 / 1
FinlandSecondarySchool	0 / 106	**0 / 102**
GreeceHighSchool1	0 / 0	0 / 0
GreecePatras3rdHS2010	0 / 0	0 / 0
GreecePreveza3rdHS2008	0 / 0	0 / 0
ItalyInstance1	0 / 28	**0 / 23**
NetherlandsGEPRO	1 / 566	**1 / 382**
NetherlandsKottenpark2003	0 / 1410	**0 / 1189**
NetherlandsKottenpark2005	0 / 1078	**0 / 963**
SouthAfricaLewitt2009	0 / 58	**0 / 0**

5.2.2 Second Round

In the second round organizers ran each solver ten times on each instance with a time limit of 1000 seconds. We reproduced the organizers experiments (considering the provided benchmark) and the obtained results are shown in Table 5. The second column presents the results obtained with KHE constructive algorithm, which is deterministic and the third column presents the results obtained with the improvements of our solver. $f(s^*)$ indicates the best solution found, $f(\overline{s})$ the average and σ the standard deviation of ten executions of the solver.

Table 6 presents the final ordering of competitors in the second round of ITC 2011. Only four teams participated of this round, so the ranking ranges from 1 (best) to 4 (worst). It is important to highlight that our team could not compete in instances *BrazilInstance2*, *BrazilInstance3*, *BrazilInstance4* and *BrazilInstance6* because we were the providers of these instances. Thus, we were excluded from ranking in these instances and they were ranked from 1 to 3 in these cases.

Table 4 Results of first round of ITC 2011

Instance	**ITC 2011 $f(s^*)$**	**Best $f(s^*)$**	**Winner Team**
AustraliaBGHS98	7 / 433	3 / 494	HySTT
AustraliaSAHS96	23 / 44	8 / 52	HySTT
AustraliaTES99	26 / 134	1 / 140	HySTT
BrazilInstance1	0 / 24	0 / 11	VAGOS
BrazilInstance4	0 / 112	0 / 44	VAGOS
BrazilInstance5	0 / 225	0 / 43	VAGOS
BrazilInstance6	0 / 209	0 / 77	VAGOS
BrazilInstance7	0 / 330	0 / 122	VAGOS
EnglandStPaul	0 / 18444	0 / 136	Lectio
FinlandArtificialSchool	0 / 0	-	-
FinlandCollege	0 / 0	-	-
FinlandHighSchool	0 / 1	-	-
FinlandSecondarySchool	0 / 106	0 / 88	Lectio
GreeceHighSchool1	0 / 0	-	-
GreecePatras3rdHS2010	0 / 0	-	-
GreecePreveza3rdHS2008	0 / 0	-	-
ItalyInstance1	0 / 28	0 / 12	VAGOS
NetherlandsGEPRO	1 / 566	1 / 382	GOAL
NetherlandsKottenpark2003	0 / 1410	0 / 532	Lectio
NetherlandsKottenpark2005	0 / 1078	0 / 533	Lectio
SouthAfricaLewitt2009	0 / 58	0 / 0	VAGOS

5.2.3 Third Round

The third round of ITC 2011 was quite similar to the first one, but the hidden instance set was considered instead of the public ones. Competitors had one month to produce the solutions. In this round, the best known solutions for instances were not provided, so we used the same procedure as shown in Figure 9, but instead of taking the initial solution from the XHSTT archive, GOAL solver generated it with KHE. Table 7 presents the results obtained by GOAL solver in this round.

Table 8 presents the ordering of competitors in this round. Again, GOAL team could not compete in instances *BrazilInstance2*, *BrazilInstance3*, *BrazilInstance4* and *BrazilInstance6*. VAGOS team sent solutions only to a few instances, and so they received the worst rank in instances which they did not sent any solution.

5.3 XHSTT-2013 and Parameter Tuning

The GOAL solver presented great results in ITC 2011. To provide another validation of the robustness of our solver, we performed an additional set of experiments with the recently released XHSTT-2013[8] set of instances.

Differently from the competition, when a single set of parameters obtained with limited tuning was used in all executions, in these experiments we per-

[8] `http://www.utwente.nl/ctit/hstt/archives/XHSTT-2013/`

Table 5: Results obtained with GOAL solver in the second round of ITC 2011

Instance	KHE		GOAL		
	$f(s^*)$	t_s	$f(s^*)$	$f(\bar{s})$	σ
BrazilInstance2	4 / 90	0	1 / 54	1.0 / 63.9	±0.0 / ±6.5
BrazilInstance3	3 / 240	1	0 / 117	0.0 / 127.8	±0.0 / ±7.9
BrazilInstance4	39 / 144	1	17 / 92	17.2 / 99.6	±0.4 / ±5.9
BrazilInstance6	11 / 291	0	4 / 207	4.0 / 223.5	±0.0 / 10.4
FinlandElementarySchool	9 / 30	6	0 / 3	0.0 / 4.0	±0.0 / ±0.5
FinlandSecondarySchool2	2 / 1821	109	0 / 0	0.0 / 0.4	±0.0 / ±0.7
Aigio1stHighSchool2010-2011	14 / 757	20	0 / 4	0.0 / 15.3	±0.0 / ±7.9
Italy_Instance4	39 / 21238	28	0 / 305	0.0 / 658.4	±0.0 / ±280.2
KosovaInstance1	1333 / 566	152	0 / 4238	14.0 / 6934.4	±10.7 / ±1862.4
Kottenpark2003	3 / 78440	402	0 / 41479	0.6 / 90195.8	±0.7 / ±17996.0
Kottenpark2005A	35 / 23677	489	33 / 27929	33.9 / 27480.4	±0.9 / ±2759.2
Kottenpark2008	63 / 140083	333	25 / 27410	25.7 / 31403.7	±0.7 / ±3969.3
Kottenpark2009	55 / 211095	229	33 / 159895	36.6 / 154998.5	±2.1 / ±24265.8
Woodlands2009	19 / 0	22	2 / 10	2.0 / 15.8	±0.0 / ±2.8
Spanishschool	1 / 4103	17	0 / 642	0.0 / 865.2	±0.0 / ±177.4
WesternGreeceUniversity3	0 / 30	126	0 / 5	0.0 / 5.6	±0.0 / ±0.5
WesternGreeceUniversity4	0 / 41	92	0 / 6	0.0 / 7.4	±0.0 / ±1.0
WesternGreeceUniversity5	17 / 44	50	0 / 0	0.0 / 0.0	±0.0 / ±0.0
Average	91.5 / 26816.1	115.4	6.4 / 14577.6	7.5 / 17394.4	±0.9 / ±2853.0

formed an extensive parameter tuning. The objective was to verify how sensitive the solver was with respect to parameter changes, as well as to determine a new set of improved default parameters. Different parameter settings were evaluated using the same metric of the competition. The results are presented in Table 9. For each one of the 24 XHSTT-2013 instances, we ran 5 independent executions for the 37 different parameter settings tested. In total we

Table 6 Results of second round of ITC 2011

Instance	**GOAL**	**HFT**	**HySST**	**Lectio**
BrazilInstance2	-	3	2	1
BrazilInstance3	-	3	1	2
BrazilInstance4	-	3	1.9	1.1
BrazilInstance6	-	3	2	1
FinlandElementarySchool	1.9	4	2.95	1.15
FinlandSecondarySchool2	1	4	2.4	2.6
Aigio1stHighSchool2010-2011	1	4	2.8	2.2
Italy_Instance4	1.1	4	3	1.9
KosovaInstance1	1	3	4	2
Kottenpark2003	1.4	4	1.6	3
Kottenpark2005A	1.4	3.6	1.6	3.4
Kottenpark2008	1	4	2.1	2.9
Kottenpark2009	1	4	2	3
Woodlands2009	1.7	4	2.8	1.5
Spanishschool	1	4	2	3
WesternGreeceUniversity3	1	3	2	4
WesternGreeceUniversity4	1	4	2	3
WesternGreeceUniversity5	1	4	2	3
Average	1.18	3.64	2.23	2.32

Table 7 Results obtained with GOAL solver in the third round of ITC 2011

Instance	**GOAL** $f(s^*)$
BrazilInstance2	0 / 32
BrazilInstance3	0 / 101
BrazilInstance4	1 / 136
BrazilInstance6	0 / 160
FinlandElementarySchool	0 / 3
FinlandSecondarySchool2	0 / 0
Aigio1stHighSchool2010-2011	0 / 0
Italy_Instance4	0 / 61
KosovaInstance1	0 / 3
Kottenpark2003	0 / 5355
Kottenpark2005A	24 / 13930
Kottenpark2008	8 / 27909
Kottenpark2009	19 / 5565
Woodlands2009	0 / 441
Spanishschool	0 / 12
WesternGreeceUniversity3	0 / 5
WesternGreeceUniversity4	0 / 8
WesternGreeceUniversity5	0 / 0

executed our solver 4,440 times, for ten minutes each run, adjusted according to the provided benchmark.

Besides searching for sets of fixed parameters, we also invested time trying to determine parameters that could be defined according to some instance characteristic. T_0, which is a critical Simulated Annealing parameter, was the chosen one for these tests. In the third column of Table 9, cells with negative values indicate that T_0 was decided in run-time. This negative value, $-x$,

Table 8 Results of third round of ITC 2011

Instance	**GOAL**	**HFT**	**HySST**	**Lectio**	**VAGOS**
BrazilInstance2	-	4	3	2	1
BrazilInstance3	-	4	3	2	1
BrazilInstance4	-	4	3	2	1
BrazilInstance6	-	4	3	1	2
FinlandElementarySchool	2.5	2.5	2.5	2.5	-
FinlandSecondarySchool2	2	4	2	2	-
Aigio1stHighSchool2010-2011	1	4	3	2	-
Italy_Instance4	2	4	1	3	-
KosovaInstance1	1	3	2	4	-
Kottenpark2003	3	4	2	1	-
Kottenpark2005A	2	3	1	4	-
Kottenpark2008	1	4	2	3	-
Kottenpark2009	2	4	1	3	-
Woodlands2009	1	4	2	3	-
Spanishschool	1	4	3	2	-
WesternGreeceUniversity3	1.5	4	3	5	1.5
WesternGreeceUniversity4	1	3	2	4	-
WesternGreeceUniversity5	2	4	2	5	2
Average	1.64	3.75	2.25	2.75	3.86

indicates that T_0 was set to be $x \times f(\hat{s})$, where $f(\hat{s})$ is the cost of the worst neighbor in a sampling of the neighborhood $N(s)$[9] of the initial solution s. Results with these tests where not encouraging, as the method behaved better with fixed parameters than with dynamic ones.

The final evaluation of each parameter setting is displayed in "column av. rank" (column average rank). No parameter setting dominated every other, and so the best parameter setting was not ranked with value 1. One interesting result of this test is that we managed to obtain good results with very different parameter configurations. In fact, among the three best parameter settings, one had $\alpha = 0.5$ while another had $\alpha = 0.85$. We have observed that even though our solver has a large number of parameters, it is not hard to find parameter configurations that perform well.

5.4 Discussion of Results

In the first round of ITC 2011 we were able to improve almost all of the best known solutions. In instance *SouthAfricaLewitt2009* we found a solution with same cost as the best solution generated in this round, but we were not the first team to produce it. Thus, VAGOS got the prize in this instance. In this round, our performance was below average as our solver was still in development and some improvements came too late to improve the solutions sent.

The second round was the most important in the competition, since all solvers had their performance fairly compared in a controlled computational environment. In this round our solver was the best ranked in twelve out of

[9] 100 neighbors were considered in our tests.

Simulated Annealing				Iterated Local Search			av. rank
$SAmax$	α	T_0	$SAreheats$	$ILSlsMax$	p_0	p_{max}	
10,000	0.85	5.0	1	14,000	1	10	8.44
10,000	0.85	5.0	1	15,000	1	10	9.78
10,000	0.50	5.0	5	14,000	1	10	9.85
10,000	0.85	5.0	1	13,000	1	10	9.90
10,000	0.50	5.0	5	11,000	1	10	9.92
10,000	0.90	5.0	1	14,000	1	10	9.92
10,000	0.50	5.0	5	12,000	1	10	10.03
11,000	0.85	5.0	1	14,000	1	10	10.14
10,000	0.95	5.0	1	14,000	1	10	10.18
10,000	0.80	5.0	1	14,000	1	10	10.26
17,000	0.50	5.0	3	11,000	1	10	10.47
11,000	0.50	5.0	5	12,000	1	10	10.52
10,000	0.80	5.0	3	14,000	1	9	10.62
10,000	0.50	5.0	5	15,000	1	10	10.65
10,000	0.50	5.0	4	14,000	1	10	10.66
10,000	0.70	5.0	5	10,000	1	10	10.74
10,000	0.80	5.0	1	10,000	1	10	11.06
10,000	0.50	5.0	4	10,000	1	10	11.36
10,000	0.50	5.0	5	13,000	1	10	11.36
10,000	0.60	5.0	5	10,000	1	10	11.38
10,000	0.50	5.0	5	12,000	1	9	11.43
10,000	0.70	5.0	2	10,000	1	10	11.72
10,000	0.50	5.0	5	9,000	1	10	11.82
10,000	0.50	4.0	5	10,000	1	10	11.97
11,000	0.50	5.0	5	10,000	1	10	12.12
9,000	0.50	5.0	5	12,000	1	10	12.22
10,000	0.50	5.0	6	10,000	1	10	12.32
10,000	0.50	6.0	5	10,000	1	10	12.32
10,000	0.40	5.0	5	10,000	1	10	12.35
7,200	0.50	5.0	7	10,000	1	10	12.68
10,000	0.50	5.0	5	12,000	1	11	12.82
10,000	0.50	-1.0	2	10,000	1	10	12.86
10,000	0.70	-0.7	4	10,000	1	10	12.88
10,000	0.50	5.0	5	12,000	2	10	12.91
10,000	0.50	5.0	5	10,000	1	11	13.03
10,000	0.50	-0.3	5	10,000	1	10	13.38
10,000	0.50	5.0	5	10,000	1	9	13.73

Table 9 Different parameter configurations and their average ranks considering XHSTT-2013

fourteen[10] instances. We were also highly competitive in the two instances for which we did not find the best results. This way, we won this round by a wide margin. The success of GOAL solver in this round is certainly linked to the effectiveness of the neighborhood exploration, since the local search procedures were able to find good solutions, even to huge instances, in a short amount of time. One can observe that the improvement compared to KHE initial solution and to the competitors results is really significant.

[10] Excluding the Brazilian instances, in which we could not compete.

In the third round of ITC 2011, the GOAL solver was the best ranked in ten out of fourteen instances and was the winner of the round. Again our solver was also competitive in the other instances. In this round the final version of our solver was complete, thus we got much better results than in the first round.

For some instances, even the production of feasible solutions is a hard task. These instances commonly define most of constraints as hard constraints. Therefore, it is not expected for a solver to always find feasible solutions for the whole set of instances. Therefore, the use of the pair *infeasibility/quality* for describing results was encouraged. Even in the restricted timeout, our solver was able to reach eleven out of eighteen [11] feasible solutions (see Table 5). Without the time limit constraint we were able to find thirteen out of eighteen feasible solutions (see Table 7).

We believe that the success of our solver in ITC 2011 was due to the following factors: (1) the fast generation of initial solutions by KHE (see Table 5); (2) the diversity of local search moves, which allowed a comprehensive exploration of the search space and (3) the controlled application of these moves inside our SA (Algorithm 1) and ILS (Algorithm 2) implementations.

6 Concluding Remarks

Our local search approach was able to find feasible solutions for almost all instances and won the third International Timetabling Competition. This result, coupled with the result of the second ITC, where another local search approach won (Muller, 2009), indicates that local search methods may be nowadays the best heuristic approach for timetabling problems.

Nevertheless, we believe that there is still room for improvement in our approach. Some possible future works are (1) develop new additional neighborhoods in order to make more significant structural changes on the solution in a single move; (2) develop automatic parameter tuning and (3) explore the implementation of other metaheuristics using the already implemented local search procedures.

Another future work would be the development of a graphical user interface to allow schools and universities from all around the world to produce their instances and solve them with our solver.

Acknowledgements The authors acknowledge FAPEMIG (grant APQ-04611-10) and CNPq (grant 552289/2011-6) for supporting the development of this research.

[11] We would like to highlight that for some of these instance there isn't a known feasible solution yet.

References

Even S, Itai A, Shamir A (1976) On the complexity of timetable and multi-commodity flow problems. SIAM Jounal of Computing 5(4):691–703

Gendreau M, Potvin J (eds) (2010) Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol 146, 2nd edn. Springer

de Haan P, Landman R, Post G, Ruizenaar H (2007) A case study for timetabling in a dutch secondary school. In: Lecture Notes in Computer Science: VI Practice and Theory of Automated Timetabling. Berlin : Springer, pp 3867 : 267–279

Johnson DS, Aragon CR, McGeoch La, Schevon C (1991) Optimization by Simulated Annealing: An Experimental Evaluation Part II, Graph Coloring and Number Partitioning. Operations Research 39(3):378–406, DOI 10.1287/opre.39.3.378, URL `http://pubsonline.informs.org/doi/abs/10.1287/opre.39.3.378`

Kingston J (2013) Educational timetabling. In: Uyar AS, Ozcan E, Urquhart N (eds) Automated Scheduling and Planning, Studies in Computational Intelligence, vol 505, Springer Berlin Heidelberg, pp 91–108, DOI 10.1007/978-3-642-39304-4_4, URL `http://dx.doi.org/10.1007/978-3-642-39304-4_4`

Kingston JH (2005) A tiling algorithm for high school timetabling. In: Lecture notes in computer science: V Practice and theory of automated timetabling. Berlin: Springer, pp 3616 : 208–225

Kingston JH (2006) Hierarchical timetable construction. In: Problems, Proceedings of the First International Conference on the Practice and Theory of Automated Timetabling

Kingston JH (2012) A software library for school timetabling. URL `http://sydney.edu.au/engineering/it/~jeff/khe/`, retrieved April 2012

Kirkpatrick S, Gellat DC, Vecchi MP (1983) Otimization by simulated annealing. In: Science, pp 202, 671–680

Kristiansen S, Stidsen TR (2013) A comprehensive study of educational timetabling, a survey. Report 8.2013, DTU Management Engineering

Lourenco HR, Martin OC, Stutzle T (2003) Iterated local search. In: Glover F, Kochenberger G (eds) Handbook of Metaheuristics, Kluwer Academic Publishers, Boston, chap 11

Lú Z, Hao JK (2010) Adaptive Tabu Search for course timetabling. European Journal of Operational Research 200(1):235–244

Muller T (2009) ITC2007 solver description: a hybrid approach. Annals of Operations Research 172(1)

Nurmi K, Kyngas J (2007) A framework for school timetabling problem. In: Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications, Paris, pp 386–393

Pillay N (2013) A survey of school timetabling research. Annals of Operations Research DOI 10.1007/s10479-013-1321-8

Post G, Ahmadi S, Daskalaki S, Kingston JH, Kyngas J, Nurmi C, Ranson D (2010) An XML format for benchmarks in High School Timetabling. In: Annals of Operations Research, pp 3867 : 267–279

Post G, Kingston J, Ahmadi S, Daskalaki S, Gogos C, Kyngas J, Nurmi C, Musliu N, Pillay N, Santos H, Schaerf A (2011) XHSTT: an XML archive for high school timetabling problems in different countries. Annals of Operations Research pp 1–7

Post G, Gaspero L, Kingston J, McCollum B, Schaerf A (2013) The third international timetabling competition. Annals of Operations Research pp 1–7, DOI 10.1007/s10479-013-1340-5, URL `http://dx.doi.org/10.1007/s10479-013-1340-5`

Santos HG, Ochi LS, Souza MJF (2005) A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. In: ACM Journal of Experimental Algorithmics, pp 10 : 2–9

Santos HG, Uchoa E, Ochi L, Maculan N (2012) Strong bounds with cut and column generation forclass-teacher timetabling. Annals of Operations Research 194:399–412

Schaerf A (1999) A survey of automated timetabling. Artificial Intelligence Review 13(2):87–127

Souza M, Ochi L, Maculan N (2003) A GRASP-Tabu Search Algorithm for solving School Timetabling Problems, Kluwer Academic Publishers, pp 659–672

Valourix C, Housos E (2003) Constraint programming approach for school timetabling. In: Computers & Operations Research, pp 30 : 1555–1572

Wright M (1996) School timetabling using heuristic search. In: Journal of Operational Research Society, pp 47 : 347–357