A Simulated Annealing Based Approach to the High School Timetabling Problem

George H.G. Fonseca, Samuel S. Brito, and Haroldo G. Santos

Graduate Program in Computer Science
Federal University of Ouro Preto
Diogo de Vasconcelos st. 150, 35930-000, Ouro Preto, Brazil
george@decea.ufop.br, {samuelsouza,haroldo}@iceb.ufop.br

Abstract. The High School Timetabling Problem remains subject of many research in Artificial Intelligence and Operational Research fields because of its hardness to solve and practical importance. A solution for this problem basically consists in the schedule of lessons to timeslots and the assignment of resources for these lessons. This work considers the solution of the problem of the ongoing Third International Timetabling Competition (ITC), which includes a diverse set of instances from many educational institutions around the world. We proposed an approach based on Simulated Annealing. One important structural feature of our approach is the use of the KHE engine to generate initial solutions combined with a multi-neighborhood search approach. The achieved results were encouraging: nine out of seventeen feasible solutions were found and five out of twenty one all-time best solutions were improved or matched in the processing limit stipulated in the ITC.

Keywords: Simulated Annealing, High School Timetabling Problem, Third International Timetabling Competition.

1 Introduction

The High School Timetabling Problem, denoted as Class×Teacher Timetabling Problem in the early works of [6], consists in the production of a schedule in a such way that no teacher or class attend more than one lesson at same time, respecting among other constraints the availability of teachers and prealocations.

The automated school timetabling has been the subject of much research in the fields of Artificial Intelligence and Operational Research. In [2] were presented some reasons to this interest:

- Hardness to solve. Find a timetabling that satisfies the interest of all elements involved is a hard task. Moreover, often the simple construction of a valid timetabling is a complicated task;
- Practical importance. A good timetabling can improve the staff satisfaction and allow the school to be more efficient in managing their resources. Moreover a good schedule can improve the students performance;

H. Yin et al. (Eds.): IDEAL 2012, LNCS 7435, pp. 540–549, 2012.

– Theoretical importance. The problem addressed is classified as $\mathcal{NP}$-Hard [4] and progress in solving such problems is a major goal of current research in Computer Science and Mathematics.

Methods based in Integer Programming were proposed to the problem [15], but they can only solve exactly a subset of instances of the problem in a viable processing time. Nowadays, metaheuristic approaches were commonly applied to the problem [12] [1] [11].

In this sense, the main goal of the presented work is to propose a Simulated Annealing based approach to the model of High School Timetabling Problem proposed by the ongoing Third International Timetabling Competition 2011 (ITC 2011). For this event, a set of diverse instances from many institutions around the world were collected and specified the XHSTT format [14], allowing for the first time the extensive experimental evaluation of solvers in a really rich set of instances [9].

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

2 High School Timetabling Problem

Used in the ITC 2011, the addressed model of High School Timetabling Problem came up with the goal of providing a generic model capable to address the various features of the High School Timetabling Problem around the world [7] [17] [13] [16] [3] [15] [14]. The model is split in three main entities:

2.1 Times and Resources

The time entity consists of a timeslot or a set of timeslots (time group). The resources, in turn, are divided in three main categories: students, teachers and rooms [14]:

students: a group of students attends to an event (lesson). Important constraints to the students are control their idle times and the number of lesson by day;

teachers: a teacher can be pre-assigned to attend an event. In some cases it is not pre-assigned and should be assigned according to their qualifications and workload limits;

rooms: most of events take place on a room. One room has a certain capacity and a set of features.

2.2 Events

An event is the basic unit of allocation, representing a simple lesson or a set of lessons (event group). A timeslot assignment to a event is called meet and a

resource assignment to a event is called task. The term course is used to designate a group of students who attend the same events. Other kinds of events, like meetings are allowed by the model [14]. An event has the following attributes:

duration that represents the number of timeslots which have to be allocated to the event;
course related to the event;
pre-assigned resources to attend the event (optional);
workload that will be added to the total workload of resources assigned to the event (optional);
pre-assigned timeslots some events yet present the timeslot in which will be allocated (optional).

2.3 Constraints

Post [14] groups the constraints in three categories: basic constraints of scheduling, constraints to the events and constraints to the resources. The objective function $f(.)$ is calculated in terms of violations to each constraint penalized according to their weight (so this is a minimization problem). They were also divided in hard constraints, whose attendance is mandatory; and soft whose attendance is desirable but not mandatory. Each instance can define whether a constraint is hard or soft. For more details, see [14].

Basic Constraints of Scheduling

1. AssignTimeConstraint: assign timeslots to each event;
2. AssignResourceConstraint: assign the resources to each event;
3. PreferTimesConstraint: indicates that some event have preference for a particular timeslot(s);
4. PreferResourcesConstraint: Indicates that some event have preference for a particular resource(s).

Constraints to Events

1. LinkEventsConstraint: schedule a set of events to the same starting time;
2. SpreadEventsConstraint: specify 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. AvoidSplitAssignmentsConstraint: for each event, assign a particular resource to all of his meets;
4. DistributeSplitEventsConstraint: for each event, assign between a minimum and a maximum meets of a given duration;
5. SplitEventsConstraint: limits the number of non-consecutive meets that an event should be scheduled and his duration.

Constraints to Resources

1. AvoidClashesConstraint: assign the resources without clashes (i.e. without assign the same resource to more than one event at a timeslot);
2. AvoidUnavailableTimesConstraint: avoid assigning resources on the times that they are not available;
3. LimitWorkloadConstraint: schedule the workload of the resources between a minimum and a maximum bound;
4. LimitIdleTimes: the number of idle times in each time group should 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. LimitBusyTimesConstraint: the number of busy times in each time group should lie between a minimum and a maximum bound for each resource;
6. ClusterBusyTimesConstraint: the number of time groups with a timeslot assigned to a resource should lie between a minimum and a maximum limit; this can be used, for example, to concentrate teacher's activities in a as few days as possible.

3 Solution Approach

Our approach uses the KHE school timetabling engine[9] to generate initial solutions and the metaheuristic Simulated Annealing perform local search around this solution. These two elements will be explained in the following subsections.

3.1 Build Method

The KHE is a platform for handling instances of the addressed problem which also provides a solver, used to build initial solutions in the presented approach. The KHE's solver was chosen to generate the initial solutions since it is able to find good initial solutions efficiently (see Table 2).

The incorporated solver is based on the concept of Hierarchical Timetabling [8], where smaller allocations are joint to generate bigger blocks of allocation until a full representation of the solution is developed.

The Hierarchical Timetabling is supported by Layer Tree data structure [8], consisting of nodes that represent the required meet and task allocation. An allocation may appear in at most one node. A Layer is a subset of nodes within the propriety that none of them can be overlapped in time. Commonly, nodes are grouped in a Layer when share resources.

The hard constraints of the problem are modeled to this data structure and then a Matching problem is solved to find the times/resources allocation. The Matching is done by connecting each node to a timeslot or resource respecting the property of Layer. For full details, see [8,9].

3.2 Neighborhood Structure

Five neighborhood structures were used:

- ES (EventSwap): two events e_1 and e_2 have their timeslots t_1 and t_2 swapped;
- EM (EventMove): an event e_1 is moved from t_1 to another timeslot t_2;
- EBM (EventBlockMove): like es, swaps the timeslot of two events e_1 and e_2, but if the events have different duration, e_1 is moved to the following the last timeslot occupied by e_2.
- RS (ResourceSwap): two events e_1 and e_2 have their assigned resources r_1 and r_2 swapped. Resources r_1 and r_2 should play the same role (e.g. both have to be teachers).
- RM (ResourceMove): an event e_1 have his assigned resource r_1 replaced to a new resource r_2.

3.3 Simulated Annealing Implementation

Proposed by [10], the metaheuristic Simulated Annealing is a probabilistic method on an analogy to thermodynamics simulating the cooling of a set of heated atoms. This technique starts its search from any initial solution. The main procedure consists of a loop that randomly generates, at each iteration, one neighbor $s^{'}$ of the current solution s.

Let Δ be the variation of the objective function value incurred from moving to the candidate neighbor ($\Delta = f(s^{'}) - f(s)$). The method accepts the move and the neighbor becomes the new current solution if $\Delta < 0$. If $\Delta \geq 0$ the neighbor can also be accepted, but in this case, with a probability $e^{-\Delta/T}$, where T is a method parameter, called temperature and regulates the probability to accept solutions with higher cost.

The temperature can assume, initially, a high value T_0. After a fixed number of iterations $SAmax$ (that represents the number of iterations needed to the system reaches the thermic equilibrium at a given temperature), the temperature is gradually lowered by a cooling rate α, such that $T_k \leftarrow \alpha \times T_{k-1}$, and $0 < \alpha < 1$. With this procedure, a greater chance to avoid local minimum occurs at the initial iteration and as T approaches zero, the algorithm behave like a descent method as it reduces the likelihood of accepting worsen movements[5].

The developed implementation of Simulated Annealing will be described in Algorithm 1, where $f(.)$ denotes the objective function and $N(.)$, the neighborhood structure. The considered parameters were $\alpha = 0.5$ and $T_0 = 5$. The parameter $SAmax$ was defined according to the number of events (nE) for each instance set by a multiplier. If the initial solution is feasible (i.e. there is no hard constraint violation), $SAmax = nE \times 10$, otherwise, $SAmax = nE \times 100$.

At each iteration the selected movement can be from any of the proposed neighborhoods. If the instance requires the resource assignment (i.e. have a constraint of kind *AssignResourceConstraint*), the kind of neighborhood is chosen based on the following probabilities: $es = 0.2$, $em = 0.4$, $ebs = 0.1$, $rs = 0.2$ and $rm = 0.1$, otherwise, the neighborhood es and er and the odds are: $es = 0.5$, $em = 0.3$ and $ebs = 0.2$. These values were empirically adjusted.

Algorithm 1. Developed implementation of Simulated Annealing

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

4 Computational Experiments

All experiments ran on a Intel ® i5 2.4 Ghz computer with 4GB of RAM computer under Ubuntu 11.10 operating system. The programming language used on software development was C++ compiled by GCC 4.6.1. The processing time was adjusted according to the benchmark available from Third International Timetabling Competition 2011. The timeout was set at 1000 seconds (normalized), like required in the second phase of ITC 2011. All of our results was validated by HSEval validator `http://sydney.edu.au/engineering/it/~jeff/hseval.cgi`.

4.1 Dataset Characterization

The set of instances available from ITC 2011 `http://www.utwente.nl/ctit/hstt/archives/XHSTT-2012` was originated from many countries and ranges from small instances to huge challenging ones. The Table 1 presents the main features of the instances.

4.2 Obtained Results

Table 2 presents the results obtained by the solution approach. The results are expressed by pair x/y, where x contains the infeasibility measure and y the

Table 1. Features of instances from ITC2011

Instance	Timeslots	Teachers	Rooms	Class	Events	Duration
AustraliaBGHS98	40	56	45	30	387	1564
AustraliaSAHS96	60	43	36	20	296	1876
AustraliaTES99	30	37	26	13	308	806
BrazilInstance1	25	8		3	21	75
BrazilInstance4	25	23		12	127	300
BrazilInstance5	25	31		13	119	325
BrazilInstance6	25	30		14	140	350
BrazilInstance7	25	33		20	205	500
EnglandStPaul	27	68	67	67	1227	1227
FinlandArtificialSchool	20	22	12	13	169	200
FinlandCollege	40	46	34	31	387	854
FinlandHighSchool	35	18	13	10	172	297
SecondarySchool	35	25	25	14	280	306
GreeceHighSchool1	35	29		66	372	372
GreecePatras3rdHS2010	35	29		84	178	340
GreecePreveza3rdHS2008	35	29		68	164	340
ItalyInstance1	36	13		3	42	133
NetherlandsGEPRO	44	132	80	44	2675	2675
NetherlandsKottenpark2003	38	75	41	18	1156	1203
NetherlandsKottenpark2005	37	78	42	26	1235	1272
SouthAfricaLewitt2009	148	19	2	16	185	838

quality measure. The column ITC 2011 $f(s^*)$ presents the best known solution to the problem instance.

The column SA contains the results obtained by solution approach. The results were collected considering five executions of the method. Thus $f(s^*)$ presents the best solution found, $f(\overline{s})$ presents the average, σ presents the standard deviation of results and t_s informs the average time (in seconds) and also the standard deviation. The bold results were better to or equal to the best known solution.

The column SA_{free} presents the results obtained by Simulated Annealing applied to the best known solution to the instance without time limitation. This aims at the first phase of ITC 2011 where neither does not matter time spent nor technologies used to obtains all-time best solutions. To this column were considered only one execution. The bold results highlight improvements that our method was able to make over the best known solution.

4.3 Discussion of Results

Our heuristic approache used of KHE solver as build procedure and the a diverse set of neighborhood kinds. The KHE solver was able to quickly find good solutions and the proposed neighborhood structures were able to consistently explore the solutions space and perform significant improvements in te initial solution.

Table 2. Obtained Results

Instance	ITC 2011 $f(s^*)$	KHE Solver		Simulated Annealing				SA$_{free}$
		$f(s)$	t_s	$f(s^*)$	$f(\overline{s})$	σ	$\overline{t_s}$	
AustraliaBGHS98	7.0 / 433.0	11.0 / 415.0	16.7	11.0 / 375.0	11.0 / 381.6	±0.0 / ±5.7	147.5±3.5	**4.0 / 367.0**
AustraliaSAHS96	23.0 / 44.0	11.0 / 15.0	61.3	**11.0 / 11.0**	11.0 / 11.8	±0.0/ ±1.3	167.3±1.6	**22.0 / 44.0**
AustraliaTES99	26.0 / 134.0	6.0 / 418.0	4.7	**6.0 / 148.0**	6.0 / 148.0	±0.0 / ±0.0	72.1±0.4	**25.0 / 124.0**
BrazilInstance1	0.0 / 24.0	0.0 / 81.0	0.0	0.0 / 36.0	0.0 / 39.6	±0.0 / ±6.5	0.7±0.0	**0.0 / 15.0**
BrazilInstance4	0.0 / 112.0	17.0 / 225.0	0.7	12.0 / 150.0	12.0 / 159.0	±0.0 / ±12.0	4.8±0.4	**0.0 / 103.0**
BrazilInstance5	0.0 / 225.0	12.0 / 258.0	0.0	6.0 / 149.0	7.0 / 167.4	±1.0 / ±20.7	4.7±1.2	**0.0 / 210.0**
BrazilInstance6	0.0 / 209.0	11.0 / 339.0	1.0	5.0 / 258.0	5.6 / 262.8	±0.5 / ±10.3	5.3±0.0	**0.0 / 173.0**
BrazilInstance7	0.0 / 330.0	20.0 / 429.0	1.3	13.0 / 285.0	14.6 / 281.4	±0.9 / ±11.5	8.1±0.4	**0.0 / 318.0**
EnglandStPaul	0.0 / 18444.0	0.0 / 49756.0	77.3	**0.0 / 15208.0**	0.0 / 17916.8	±0.0 / ±1991.0	592.1±3.7	**0.0 / 11732.0**
FinlandArtificialSchool	0.0 / 0.0	16.0 / 18.0	1.3	10.0 / 8.0	13.0 / 8.0	±1.7 / ±2.5	53.7±2.3	0.0 / 0.0
FinlandCollege	0.0 / 0.0	20.0 / 747.0	2.7	4.0 / 109.0	5.6 / 114.2	±1.1 / ±15.1	157.3±2.4	0.0 / 0.0
FinlandHighSchool	0.0 / 1.0	7.0 / 446.0	1.3	0.0 / 74.0	1.4 / 89.4	±1.1 / ±17.8	59.2±1.8	0.0 / 0.1
FinlandSecondarySchool	0.0 / 106.0	35.0 / 301.0	17.3	3.0 / 114.0	3.8 / 121.2	±0.8 / ±28.4	164.5±1.9	0.0 / 106.0
GreeceHighSchool1	0.0 / 0.0	0.0 / 0.0	68.7	**0.0 / 0.0**	0.0 / 0.0	±0.0 / ±0.0	73.5±1.9	0.0 / 0.0
GreecePatras3rdHS2010	0.0 / 0.0	8.0 / 399.0	8.7	0.0 / 80.0	0.6 / 91.2	±0.9 / ±19.1	97.9±2.4	0.0 / 0.0
GreecePreveza3rdHS2008	0.0 / 0.0	6.0 / 684.0	8.0	0.0 / 189.0	1.4 / 165.4	±1.1 / ±20.5	87.4±0.8	0.0 / 0.0
ItalyInstance1	0.0 / 28.0	0.0 / 323.0	1.3	0.0 / 34.0	0.0 / 40.4	±0.0 / ±4.5	2.7±0.0	0.0 / 28.0
NetherlandsGEPRO	1.0 / 566	2.0 / 69118	926.0	2.0 / 6095.0	2.0 / 18511.0	±0.0 / ±3641.4	1000.0±0.0	**1.0 / 549.0**
NetherlandsKottenpark2003	0.0 / 1410.0	1.0 / 72413.0	264.7	0.0 / 5481.0	0.2 / 7224.0	±0.4 / ±1185.4	1000.0±0.0	**0.0 / 1189.0**
NetherlandsKottenpark2005	0.0 / 1078.0	18.0 / 22284.0	346.7	10.0 / 2783.0	11.6 / 3045.8	±0.9 / ±270.3	1000.0±0.0	**0.0 / 963.0**
SouthAfricaLewitt2009	0.0 / 58.0	361.0 / 0.0	16.0	**0.0 / 20.0**	0.4 / 28.4	±0.5 / ±8.9	51.6±0.9	**0.0 / 42.0**

As Table 2 shows, our approach was able to satisfactorily process even huge instances problems, improving sometimes the best known solutions, in the small provided timeout. From small to medium instances the approach was very quick. The standard deviation of the processing times also was low, showing consistence of method in terms of time spent.

For some instances, even the production of feasible solutions to configures a very hard task. These instances commonly define most of constraints as hard constraints. Therefore, ITC 2011 do not expect that a solver could find all feasible solutions and the use of pair $infeasibility/quality$ was encouraged. Our approach was able to reach nine out of seventeen [1] feasible solutions in the given timeout.

Our approach was able to improve or match the best known solution on five out of twenty one instances. It is noteworthy that there were no time or technology constraints to the original best known solutions. The proposed approach, even under the given timeout, was able to improve some best known results and get close to the others.

Taking the best known solution as initial solution without time limits, our local search technique has able to improve thirteen out of sixteen [2] solutions. Some of these results was the best achieved on the first phase of ITC 2011.

5 Concluding Remarks

Our Simulated Annealing based metaheuristic was able to improve or match the all-time best solution from ITC-2011 to five out of twenty one instances in a tiny processing time. Taking the best known solution as input of our method make him capable to improve thirteen out of sixteen instances. These results are encouraging due to the problem hardness and practical importance.

Nevertheless, we believe that there is still room for improvement in our approach. Some possible future works are (1) develop new additional neighborhood categories able to make more significant structural changes to a solution in restricted amount of movements, like kempe moves and ejection chains [11]; (2) develop and perform a computational study of other metaheuristics such as Variable Neighborhood Search to the problem, which could exploit available neighborhoods in different ways; and (3) develop a Mixed Integer Programing heuristic to the problem.

References

1. Barbosa, S.H., Souza, S.R.: Resolução do problema de programação de cursos universitários baseada em currículos via uma meta-heurística híbrida grasp-ils-relaxado. In: Proceedings of XLIII SBPO XLIII Simpósio Brasileiro de Pesquisa Operacional. SOBRAPO, Ubatuba, pp. 1:2827–1:2882 (2011)

[1] The remaining four instances were not taken into count because there is no guarantee that a feasible solution exists.

[2] The remaining five instances already had a zero cost known solution.

2. Bufé, M., Fischer, T., Gubbels, H., Häcker, C., Hasprich, O., Scheibel, C., Weicker, K., Weicker, N., Wenig, M., Wolfangel, C.: Automated Solution of a Highly Constrained School Timetabling Problem - Preliminary Results. In: Boers, E.J.W., Gottlieb, J., Lanzi, P.L., Smith, R.E., Cagnoni, S., Hart, E., Raidl, G.R., Tijink, H. (eds.) EvoIASP 2001, EvoWorkshops 2001, EvoFlight 2001, EvoSTIM 2001, EvoCOP 2001, and EvoLearn 2001. LNCS, vol. 2037, pp. 431–440. Springer, Heidelberg (2001)
3. de Haan, P., Landman, R., Post, G., Ruizenaar, H.: A Case Study for Timetabling in a Dutch Secondary School. In: Burke, E.K., Rudová, H. (eds.) PATAT 2007. LNCS, vol. 3867, pp. 267–279. Springer, Heidelberg (2007)
4. Garey, M.R., Jonhson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
5. Glover, F., Kochenberger, G.: Handbook of Metaheuristics. International Series in Operations Research & Management Science. Kluwer Academic Publishers (2003)
6. Gotlieb, C.C.: The construction of class-teacher time-tables. In: Proc. IFIP Congress, pp. 73–77. North Holland Pub. Co., Munich (1963)
7. Kingston, J.H.: A Tiling Algorithm for High School Timetabling. In: Burke, E.K., Trick, M.A. (eds.) PATAT 2004. LNCS, vol. 3616, pp. 208–225. Springer, Heidelberg (2005)
8. Kingston, J.H.: Hierarchical Timetable Construction. In: Burke, E.K., Rudová, H. (eds.) PATAT 2007. LNCS, vol. 3867, pp. 294–307. Springer, Heidelberg (2007)
9. Kingston, J.H.: A software library for school timetabling. Disponível em (2012), `http://sydney.edu.au/engineering/it/~jeff/khe/` (acessado em Abril de 2012)
10. Kirkpatrick, S., Gellat, D.C., Vecchi, M.P.: Otimization by simulated annealing. Science 202, 671–680 (1983)
11. Lú, Z., Hao, J.-K.: Adaptive tabu search for course timetabling. European Journal of Operational Research 200(1), 235–244 (2010)
12. Muller, T.: Itc2007 solver description: a hybrid approach. Annals OR 172(1), 429–446 (2009)
13. Nurmi, K., Kyngas, J.: A framework for school timetabling problem. In: Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications, Paris, pp. 386–393 (2007)
14. Post, G., Ahmadi, S., Daskalaki, S., Kingston, J.H., Kyngas, J., Nurmi, C., Ranson, D.: An Xml Format for Benchmarks in High School Timetabling. Annals of Operations Research 3867, 267–279 (2010)
15. Santos, H.G., Uchoa, E., Ochi, L.S., Maculan, N.: Strong bounds with cut and column generation for class-teacher timetabling. Annals OR 194(1), 399–412 (2012)
16. Valourix, C., Housos, E.: Constraint programming approach for school timetabling. Computers & Operations Research 30, 1555–1572 (2003)
17. Wright, M.: School timetabling using heuristic search. Journal of Operational Research Society 47, 347–357 (1996)