TOP (2015) 23:350–354
DOI 10.1007/s11750-015-0363-2

DISCUSSION

Comments on: An overview of curriculum-based course timetabling

Roberto Asín

Published online: 19 March 2015

Abstract The paper presents an updated revision of the state of the art in the curriculum-based course timetabling problem (CB-CTT). CB-CTT has been proposed for one track of the Second International Timetabling Competition (ITC-2007). Since then, several solution methods have been proposed and their results could have been tracked thanks to the web platform hosted at http://satt.diegm.uniud.it/ctt. In the following lines, some thoughts and remarks will be given, mainly from the SAT-solving point of view. First, we review some facts about CB-CTT and remark some characteristics that, to our understanding, may have helped the community to do better research. Later, from the different cited approaches, we observe two things: (1) how the problem has been represented (the modeling language) and (2) the solving algorithms. Finally, we close this comment pointing out some possible future work in CB-CTT.

1 About the problem

The problem has been proposed for the third track of the Second International Timetabling Competition (ITC-2007) and it abstracts the timetabling problem of universities where students' courses are predefined according to a specific curricula. In practice, this modeling may be useful in cases where most students belong to well-defined groups that take, exactly, the same courses, in contrast with the post-enrollment-based course timetabling problem (PE-CTT), which was proposed for another track of the competition.

During the competition, and after, the efforts made by the track organizers have succeeded at encouraging several researchers to work in the problem and several papers

This comment refers to the invited paper available at doi:10.1007/s11750-015-0366-z.

R. Asín (✉)
Universidad Católica de la Santísima Concepción, Concepción, Chile
e-mail: rasin@ucsc.cl

and solving proposals have been produced. We think that the following facts were key for such success:

A well-defined problem: One inconvenient for research in timetabling, before the International Timetabling Competitions, was that almost every related publication presented the problem in a new particular way and the sets of considered constraints were also non-uniform. This made difficult to compare the solving methods and also to judge the contribution of each of them. We know that timetabling problems may differ from institution to institution and that the CB-CTT formulation may lack of being an oversimplification of many institutions' real cases. Nevertheless, from an academic point of view, to our understanding, it is important to count with a well-defined and uniform formulation to share a common framework, compare, judge and analyze each proposal. Once a baseline is settled, as it is being done with CB-CTT, the formulation can be extended, taking into account more constraints and thus, approaching the results to industrial production.
Public datasets: Another important characteristic about CB-CTT research has been the fact that several public datasets were generated. In line with the above, having common datasets allows to objectively compare results and, again, judge the relevance of each new contribution. This contribution may not only come from better results, but also from new solution methods. Sharing the instances may help researchers to know if such methods are competitive or may constitute a contribution to the state of the art. Also, we think of importance that these datasets correspond to real-world examples. The fact that the datasets come from real-world instances ensure that improving the results on community may also contribute to solving the problem in reality and that the solution methods can be adapted to run on commercial applications.
Results tracking: Besides the above characteristics, the existence of the web platform, allowing researchers to stay up to date on the current best results, the methods used, current lower bounds and being able to use applications to validate solutions or even to generate more instances, has helped to speed up their research in several ways. For instance, keeping track of the solutions may have ease researchers the task of following the state of the art on the problem. Also, it has permitted to broaden the proposed methods to "non-traditional" ones, for which the specific competition rules were not suitable, but, even though, clearly constitute a contribution to the state of the art. Lower bound tracking also helps non-exact methods to evaluate the quality of the solutions their techniques throw.

In general, all these features may have made the problem attractive to work in and may have encouraged research in this field. All these characteristics were possible thanks to the work of several people and we use this space to thank their excellent hard work and dedication.

2 Modeling languages

While reviewing the different approaches, we think of importance to remark two different aspects in each approach. First, to notice that several methods represent

the problem as an instance of a more general problem, which we call a modeling language. Second, the proper solving algorithm. Regarding the modeling languages, we comment about the most important ones cited in the paper:

Direct representation: Several authors used a direct representation of the problem. This has the advantage that several operations inside the algorithms may be directly related with the nature of the problem. Examples of this are operations such as: given two events assigned to the same time slot, interchange the rooms in which they are scheduled. Direct representations of the problems benefit from the knowledge on the specific domain of the problem and so, are better suited to adapt several heuristics, used by persons that make the timetables in their institutions. Nevertheless, some drawbacks on this are that solvers have to be generated almost from scratch and may not benefit from previous research in more general problems like MIPS or SAT. From this point of view, several implementation improvements and sophisticated methods that have been proven useful in other related problems should be re-implemented for the new framework or just avoided.
MIP/ILP: Maybe the most used modeling language for combinatorial optimization problems is representing them as mixed integer problems (MIPS). The main advantage of doing this is that there exist several fine-tuned solvers to handle MIPs and it is natural to take advantage of all the development that has been done. Unfortunately, for pure integer linear programming instances, it is not clear that such fine-tuned solvers constitute the state of the art. Such has been the case with CB-CTT, where the MIPS approaches have contributed mainly with the calculation of lower bounds, with limited success in improving the best solutions. We think that an important observation on the proposed ILP models for CB-CTT is to notice that all of them mainly involve 0–1 variables and cardinality constraints.
CSP: Other frequently used modeling languages for combinatorial optimization problems are those related with constraint satisfaction problems (CSP). From a more logical nature, these languages have predefined constraints that usually appear in combinatorial optimization problems. Specialized solvers for problems encoded in such languages implement very efficient algorithms to treat this kind of constraints and, for certain kind of problems, may constitute state-of-the-art solving methods.
SAT: The satisfiability problem can be seen as the most basic CSP language. In such language, all the variables have the domain $\{0, 1\}$ and the constraints are represented as clauses. In the case of CB-CTT, SAT has proven to be a good base language, since the problem's nature involves only binary variables and the constraints can be encoded as relatively small clauses plus the encoding of cardinality constraints that can be efficiently encoded into SAT (see Abío et al. 2013). At least for the public instances, the encodings are polynomial in the size of the problem. Nevertheless, it is not clear if much bigger instances may be tractable for current state-of-the-art SAT solvers.

3 Algorithms

The other point we want to discuss is the underlying algorithms that were used to solve the problem, with independence of the modeling language. For instance, we can classify the algorithmic approaches as follows:

Branch and cut based: This family of algorithms is closely related with solving MIP models. At the basis of commercial solvers like CPLEX or Gurobi, lays a Branch and cut algorithm that runs simplex at each node. The main contribution of such methods for CB-CTT may have been computing tight lower bounds to the problem. Unfortunately, it seems that the use of such commercial solvers is not, currently, state of the art for finding good solutions for CB-CTT instances. It may be interesting to study why this happens in this kind of problems.

There are also MaxSAT solvers based on branch and bound methods that, at each node, run a CDCL (see below) SAT solver. This kind of solvers have proven to be of use for CB-CTT and could achieve several good solutions.

Metaheuristic methods: This family of algorithms is the dominant in combinatorial optimization and may also be the most flexible. There exist metaheuristic-based solvers for virtually all the modeling languages used. Apparently, working over a direct representation of the problem, for defining neighborhoods and search strategies, is what gives this kind of approaches the best solutions for CB-CTT. One drawback of such approaches may be the impossibility of certifying optimality of the obtained solutions.

Backtracking based: A whole family of solvers, especially for CSP and SAT languages are based on backtracking-related search methods. For instance, modern SAT solvers implement what has been called the Conflict-Driven Clause-Learning (CDCL) algorithm (Biere et al. 2009). This algorithm lays at the solution to optimality of several CB-CTT instances. CDCL SAT solvers permit certifying unsatisfiability for a given input SAT formula. Furthermore, they can output a (hopefully small) subset of clauses that is unsatisfiable, called an unsatisfiable core. This fact, plus being able to encode cardinality constraints efficiently, permits UNSAT-core-based partial-MaxSATt solvers (see Ansótegui et al. 2009 for details) calculate lower bounds and, eventually, find optimal solutions by making successive calls to a SAT solver. Many of the best obtained results for CB-CTT have come from this approach and so, these techniques seem very well suited for this kind of problems. Nevertheless, some interesting facts arise. For instance, the underlying proof system for CDCL is resolution and it happens that some type of unsatisfiable problems have exponential resolution proofs for unsatisfiability. Such is the case of pigeonhole problems (see Haken 1985). Unfortunately, such kind of problems may be common in timetabling and, in case of unsatisfiability, CDCL-based SAT solvers may perform badly. Such is the case of instance comp01, which, even being relatively small, could not be decided, even in the basic feasibility only encoding. This fact, plus the fact that these techniques either find an optimal solution or no solution at all, may be the main drawbacks for the approach.

4 Future work

Looking at all the proposed approaches, we can find that no particular solving method outperforms all the others in all the benchmarks. It would be interesting to find out the reasons why certain approach is better in its strong instances and why it performs poorly in its weaker ones. This may help guiding future research and producing, for instance, portfolio solvers that could be more close to application than current ones. Also, this knowledge will help researchers in finding out new solution methods that can handle the newer and more complex constraints and evaluation function being considered for the problem. For instance, it may be a good idea to, for example, use SAT-based techniques for finding input feasible solutions for metaheuristic methods.

Also, each year, new solving methods appear for all the languages in which CB-CTT could have been represented and it would be worth testing them over the already generated encodings. Such may be the case of local-search-based SAT solvers like (Biere 2014) or new CDCL-based ILP solvers like (Nieuwenhuis 2014).

Acknowledgments Roberto Asín's research was funded by CONICYT, Chile, under FONDECYT project 11121220.

References

Abío I, Nieuwenhuis R, Oliveras A., Rodríguez-Carbonell E (2013) A parametric approach for smaller and better encodings of cardinality constraints. In: Principles and practice of constraint programming. Springer, Berlin, pp 80–96

Ansótegui C, Bonet ML, Levy J (2009) Solving (weighted) partial MaxSAT through satisfiability testing. In: Theory and applications of satisfiability testing-SAT 2009. Springer, Berlin, pp 427–440

Biere A (2014) Yet another local search solver and lingeling and friends entering the sat competition 2014. In: Proceedings of SAT competition 2014: solver and benchmark descriptions, vol B-2014-2, p 39

Biere A, Heule M, van Maaren H, Walsh T (2009) Handbook of satisfiability, vol 185. IOS Press, Amsterdam

Haken A (1985) The intractability of resolution. Theor Comput Sci 39:297–308

Nieuwenhuis R (2014) The intsat method for integer linear programming. In: Principles and practice of constraint programming. Springer, Berlin, pp 574–589