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

A knowledge-based approach to detecting and explaining conflicts in timetabling problems

Kylian Van Dessel · Joost Vennekens

Received: date / Accepted: date

Keywords timetabling · conflict detection · declarative programming · knowledge representation and reasoning

1 Introduction

Traditional research on combinatorial optimization problems, such as scheduling, focuses on computational efficiency. The typical goal is to compute better solutions within given time constraints, or to compute optimal solutions faster. Over the years, the field of Operational Research (OR), and more recently also the field of Artificial Intelligence (AI) put numerous techniques forward that can solve combinatorial optimization problems in a highly efficient manner. The benchmarks used in this research are often artificially generated. Moreover they typically consist of well-defined problem definitions over consistent problem instances, i.e. there are no contradicting hard constraints and instances are not constrained to the point that there is no longer a solution. This has lead to a mismatch between academic result and real-life applications, in which problems are most often ill-defined.

To overcome this gap, both research fields have put a substantial effort in solving optimization problems for real-world benchmarks. E.g., the biannual international timetabling competition gathers a large number of real-world applications for its tracks [7] and more and more conferences in the field include

K. Van Dessel
Jan Pieter De Nayerlaan 5, 2860 Sint-Katelijne-Waver
Tel.: +32-15-688191
E-mail: kylian.vandessel@kuleuven.be

J. Vennekens
Jan Pieter De Nayerlaan 5, 2860 Sint-Katelijne-Waver
Tel.: +32-15-688235
E-mail: joost.vennekens@kuleuven.be

talks on real-world applications or even have dedicated tracks for them, such as PATAT or ICAPS.

In this work we aim to identify if and, more importantly, why a problem specification is ill-defined. If we can explain in a comprehensible manner why a conflict occurs, then the inconsistency can be reported to the user. The user can then use this information to construct a consistent specification that can be handled by the system. We focus on inconsistencies in the problem instance, rather than in the problem definition itself. Existing systems typically try to detect or prevent such inconsistencies in a rather ad hoc manner, e.g., by means of a user-friendly interface or by performing input data checks. These techniques are only usable for shallow inconsistencies such as typing errors and miscounts. For the detection of more profound conflicts, more complex inference is required.

In this paper, we propose a knowledge-based approach for detecting and explaining conflicts in instances of the School Timetabling Problem (STP) in the context of Belgian secondary schools. We do this in an effort to tackle the problem from a Knowledge Representation and Reasoning (KRR) point of view as opposed to the more traditional research from the field of OR. We believe that the declarative aspect of our proposed system will allow conflicts to be explained in a more comprehensible manner.

An additional contribution will be the compilation of a data set with realistic conflicts based on actual STP instances, which will also be used for the validation of our work.

2 The IDP knowledge base system

In order to explain inconsistencies in a comprehensible manner, we start from a purely declarative knowledge base. In this approach, a problem specification consists of a set of hard and soft constraints much like in OR techniques. However, we focus on writing down the knowledge base in an intuitive language, which is understandable for non-experts, as opposed to the mathematical models often used in OR techniques.

The STP has been studied in such a context of declarative programming before, e.g., in ASP [1]. State-of-the-art answer set programming (ASP) systems typically use a two-step ground-and-solve process to unite a declarative specification language with an efficient solver. A traditional ASP system first grounds the problem specification to a propositional program. This step reduces complex language constructs and shorthands, added to the language for expressiveness and ease of modeling, to their most basic, variable-free form. In a second step an ASP solver is then used to generate answer sets (solutions) for the ground program [3].

The knowledge-based system we propose for studying conflict detection and explanation in the context of the STP, is the IDP system [2]. This system is similar to ASP systems, but uses a language based on First Order Logic (FO), extended with additional language features such as types and aggregates (as opposed to the non-monotonic reasoning ASP-language used by traditional ASP systems).

Example 1 The trivial constraint that each class should be taught by a teacher can be expressed by the following FO sentence:

$$\forall c[Class] : \exists t[Teacher] : Teaches(t, c).$$

This sentence is grounded to a disjunction of propositions over each element of the type $Teacher$, and this for each element of the type $Class$. Consider the following structure:

$$Teacher = \{Mary, John, Carl\}$$
$$Class = \{Maths, Physics\}$$

The constraint is grounded to the ground program:

$$Teaches(Mary, Maths) \vee Teaches(John, Maths) \vee Teaches(Carl, Maths).$$
$$Teaches(Mary, Physics) \vee Teaches(John, Physics) \vee Teaches(Carl, Physics).$$

A solution found by the solver would then look like this:

$$Teaches(John, Maths).\ Teaches(Mary, Physics).$$

For knowledge-based systems, several conflict detection techniques are known [4,6]. These techniques typically work by detecting an *unsatisfiable core*: a minimal subset of an inconsistent theory that is still inconsistent [5]. However, because existing techniques look for unsatisfiable cores in ground (i.e., propositional) theories, these cores tend to be very big and not comprehensible for a non-programmer. Because we aim to provide information to a timetabler, this makes these techniques unusable for us. We therefore develop a method which reduces an inconsistent theory to a smaller theory, that is still tractable in size and written in the same language as the original problem specification, i.e., the highly expressive FO(·)-language.

3 Diagnosis of an unsatisfiable problem

When representing a problem specification in a knowledge base, we first need to specify which symbols will be used for the representation (i.e. the vocabulary). The constraints then correspond to a theory over these symbols and a specific instance corresponds to a structure for part of the vocabulary. As such, the STP can be represented as a combination of a theory on all the hard and soft constraints of the problem (e.g., "Each class should be taught by a therefore qualified teacher") and a structure defining a specific STP instance (e.g., "Mary is a teacher", "teacher Mary is qualified to teach class Math").

For a given theory T and structure S, the problem of finding a solution is that of solving the model expansion problem $MX(S, T)$, which computes the set of all structures S' that extend S such that $S' \models T$.

We assume the problem to be unsatisfiable, thus $MX(S,T) = \{\}$, i.e., there are no solutions, and look for reductions S' of S for which $MX(S',T)$ is still empty. S' is considered a *reduction* of S if the domain of S' is a subset of the domain of S and, for each symbol P, the interpretation of P in S', denoted $P^{S'}$, is equal to the restriction of P^S to the domain of S'. If for each solution M to $MX(S,T)$, there exists a reduction M' of M such that M' is a solution of $MX(S',T)$, we call the reduction *T-safe*.

To help users understand why a model expansion problem $MX(S,T)$ is unsatisfiable, we might therefore look for reductions S' of S that are T-safe and for which $MX(S',T)$ is unsatisfiable. However, in the case that $MX(S,T)$ is empty, the notion of safeness is actually not very relevant, since each reduction is safe in that case.

Definition 1 *Let T be a theory and Σ a subset of the vocabulary of T. A* reduction policy *for T relative to Σ is a set of pairs $(P(x), \rho(\vec{y}, x))$ where P is a type in Σ and ρ is a Σ-formula. Given a Σ-structure S, we say that a reduction S' of S follows a reduction policy if it is the case that, for all pairs $(P(x), \rho(\vec{y}, x))$ in the policy and for each domain element d that is removed from type P, also all elements $e_1, e_2, \ldots, e_n$ are removed from their respective types for which $S \models \rho[x/d, \vec{y}/\vec{e}]$. A reduction policy is T-safe if, for each Σ-structure S, all reductions that follow the policy are T-safe.*

If we have a T-safe reduction policy, we can use the reductions that follow this policy to conclude that, if at least one of these is unsatisfiable, also the original model expansion problem is unsatisfiable.

Definition 2 *A diagnosis of an unsatisfiable problem $MX(S,T)$ is a T-safe reduction S' of S such that $MX(S',T)$ is unsatisfiable as well. Such a diagnosis is called minimal if S' itself is the only diagnosis of $MX(S',T)$.*

Example 2 Consider the following simple example. Each class must be taught, by a teacher who is qualified to teach it, but each teacher may teach only a single course:

$$\forall c[Class] : \exists t[Teacher] : Teaches(t, c).$$
$$\forall c[Class]\ t[Teacher] : Teaches(t, c) \Rightarrow Qualified(t, c).$$
$$\forall t[Teacher]\ c[Class]\ c'[Class] : Teaches(t, c) \land c \neq c' \Rightarrow \neg Teaches(t, c').$$

This theory is unsatisfiable in the following structure:

$$\begin{aligned} Teacher &= \{Mary, John, Carl\} \\ Class &= \{Maths, Physics, English\} \\ Qualified &= \{(Mary, Maths), (Mary, Physics), (John, English), (Carl, English)\} \end{aligned}$$

Obviously, the problem here is the fact that Mary is the only teacher qualified to teach both Maths and Physics. In other words, this problem will already manifest itself in the following reduction S' of S:

$$\begin{aligned} Teacher &= \{Mary\} \\ Class &= \{Maths, Physics\} \\ Qualified &= \{(Mary, Maths), (Mary, Physics)\} \end{aligned}$$

This reduction is T-safe. To prove this, it suffices to show that the reduction policy consisting of the single pair $(Teacher(x), Qualified(x, y))$ is T-safe. Because of the first formula, it is clear that $Qualified(t, c)$ covers the first formula. In order to be allowed to remove the teachers $John$ and $Carl$, this covering formula asks that we also remove all classes that $John$ and $Carl$ are allowed to teach. We see that, indeed, $Class^{S'}$ no longer contains the class $English$. Therefore, $MX(S', T)$ has at least as many solutions as the original problem $MX(S, T)$. In other words, removing a set of teachers together with all the classes they may teach only makes the problem easier. Because $MX(S', T)$ is unsatisfiable, it is a diagnosis of $MX(S, T)$. Since removing either $Mary$ or one of her two classes would render the problem satisfiable, $MX(S', T)$ has no more diagnoses and hence it is a minimal diagnosis. On the other hand, removing only the teachers but not the classes is not safe, since it makes the problem harder.

We are not interested in just any diagnosis. We are looking for those we can explain to a user. To this end, we set up a frame that dictates which safe reductions we want to apply to which elements in order to detect a certain kind of conflict. If we apply such a frame to an inconsistent problem and the reduction is inconsistent as well, then we have a diagnosis of that kind of conflict. We try to define these frames in a way that the reductions that follow the frame are as small as possible to still explain the conflict, but note that a correct diagnosis is not necessarily minimal.

4 Discussion and Future work

We presented a knowledge-based approach for detecting and explaining conflicts in optimization problems. The problem specification is written down in an intuitive, understandable language, unlike the typical mathematical models used in OR techniques. Additionally, as opposed to existing KRR research on computing unsatisfiable cores from the ground program, we propose a reduction on the level of the problem specification itself. In this way, we not only detect the conflicts, but also have the ability of explaining them in a comprehensible manner to a non-programmer.

We will put the proposed concepts to practice and study their correctness and performance in the context of the STP. We focus on ill-defined problem specifications. To this end, we compile a data set with conflicts from a real-life context. For our experiments we will use the IDP knowledge base system, which provides an expressive declarative language (FO(·)) for representing the problem specification.

References

1. Banbara, M., Soh, T., Tamura, N., Inoue, K., Schaub, T.: Answer set programming as a modeling language for course timetabling. Theory and Practice of Logic Programming **13**(4-5), 783–798 (2013). DOI 10.1017/S1471068413000495
2. Bruynooghe, M., Blockeel, H., Bogaerts, B., Cat, B.D., Pooter, S.D., Jansen, J., Labarre, A., Ramon, J., Denecker, M., Verwer, S.: Predicate logic as a modeling language: Modeling and solving some machine learning and data mining problems with IDP3. CoRR **abs/1309.6883** (2013). URL http://arxiv.org/abs/1309.6883
3. Kaufmann, B., Leone, N., Perri, S., Schaub, T.: Grounding and solving in answer set programming. AI Magazine **37**(3), 25–32 (2016). DOI 10.1609/aimag.v37i3.2672. URL https://www.aaai.org/ojs/index.php/aimagazine/article/view/2672
4. Liffiton, M., Sakallah, K.: Algorithms for computing minimal unsatisfiable subsets of constraints. J. Autom. Reasoning **40**, 1–33 (2008). DOI 10.1007/s10817-007-9084-z
5. Lynce, I., Silva, J.: On computing minimum unsatisfiable cores (2004)
6. Torlak, E., Chang, F.S.H., Jackson, D.: Finding minimal unsatisfiable cores of declarative specifications. In: J. Cuellar, T. Maibaum, K. Sere (eds.) FM 2008: Formal Methods, pp. 326–341. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
7. Van Bulck, D., Goossens, D., Belien, J., Davari, M.: The fifth international timetabling competition (itc 2021): Sports timetabling. In: MathSport International 2021, pp. 117–122. University of Reading (2021)