

# Enhancing model finding techniques for function-free logics

Juan Antonio Navarro-Pérez

Supervisor: Andrei Voronkov

Adviser: Renate A. Schmidt

Formal Methods Group

## INTRODUCTION

Automated model finding is often used in *problem solving* approaches where some given problem is encoded as a mathematical logic formula. This is done in such a way that solutions to the original problem can be recovered from the mathematical models of the resulting formula (or the proof that no models actually exist). Theorem proving is a closely related task with a stronger emphasis on obtaining *proofs* rather than *models*.

My research topic is mainly focused on automated model finding in a fragment of first-order logic known as *effectively propositional* or, alternatively, the *Bernays-Schönfinkel class*. This fragment is characterised by the fact that no function symbols are allowed<sup>1</sup> and thus enabling, in particular, a solution method known as *grounding*.

The grounding procedure consists on generating many new copies of the original formula, where variables are replaced by every possible combination of constant symbols. The conjunction of the resulting formulae is *satisfiable*, i.e. it has solutions, if and only if the original formula also was; and can be tackled then with standard *propositional satisfiability solving* techniques.

There are several aims and objectives of my proposed research in this context. First, and perhaps more importantly, to describe several applications or kinds of problems that can be expressed in this particular fragment of logic. And, secondly, to review approaches to solve this kind of problems while bringing together methods from the propositional and first-order domains. One of the objectives of my research was, in particular, to study the feasibility of solving methods that do not necessarily

---

<sup>1</sup>Neither explicit nor implicit as Skolem functions which may be introduced to remove existentials.

rely on the generation of a fully grounded formula (which could, in general, be prohibitively large).

The contributions expected out of my research would include a cross-fertilisation of different techniques and methodologies in areas such as propositional satisfiability, first-order theorem proving and model finding. A better understanding of the relations between solving methods and particular classes of problems would also provide, ideally, new insights on how to improve existing approaches with respect to specific application domains.

In other words, my research work is strongly aimed at the study of kinds of problems that do arise in practical applications and, taking advantage of particular properties that they might have, be able to solve them more efficiently.

## SUMMARY OF PROGRESS

Following my original research plan, and in order to reach the desired objectives, several different tasks were carried out during my second year of studies.

Firstly, I continued the study and experimentation of propositional solving and implementation techniques that were started at the end of my first year. This involved a brief statistical analysis of the runtime behaviour of solvers while they are searching for solutions. Although no major discoveries were produced from it, this analysis served to clarify the relationship between particular characteristics of problems, some metrics of the search process, and the impact that they have in the actual running time of the implemented algorithms.

It followed a study on how problems from different application domains, including some in bounded model checking and circuit verification, are encoded in terms of propositional satisfiability. This led to the proposal of a few *minor* improvements on the encoding process and the exploration of an idea to

enhance an existing solver [3] with capabilities to handle a special kind of *parity clauses*. This would, in principle, considerably reduce the size of the generated satisfiability problems (and thus saving some memory). Unfortunately, neither the search space nor the total runtime was reduced by this feature. Moreover, it also turned out, large parity clauses do not appear very often in typical problems from the applications considered. These results were submitted for publication and presented at the Workshop on Automated Reasoning 2006 in Bristol [5].

Until this point, my research was mostly focused on plain *propositional* problems. I shifted then to consider more closely the *function-free fragment* of first-order logic which lies at the core of my proposed research. This included a study on existing methods to solve this class of formulae (e.g. forms of grounding, first-order resolution) and experimentation using a larger base of benchmark problems: more bounded model checking problems and a variety of problems from areas such as puzzles, group theory, natural language processing, and modal logics from the TPTP problem library [7].

From this, new ideas emerged on how to enhance existing grounding techniques [e.g. 6] by incorporating lessons learnt from the propositional case. To measure the impact of such improvements, a first-order model finder [2] was modified and additional experimentation was carried out. While such optimisations, which have a similar effect to *pure literal deletion*, allowed the generation of much smaller grounded formulae, the total solving time was not affected significantly. Although these results were not considered appropriate for publication, a detailed report will be included in my PhD thesis.

## FUTURE WORK

Recently I have also started exploring a new application domain which could provide an additional source of problems. Hustadt et al. [4] proposed a way to solve a number of description logic problems, in the context of the logic  $\mathcal{SHIQ}^-$ , by encoding them into a *Datalog* program. We observed that the resulting program also lies in the fragment of effectively propositional logic formulae and, therefore, could be tackled with the sort of techniques that I have been studying so far.

Moreover, the encoding required for such description logic problems suggested that an extension of model finders, with the ability to handle a special

kind of ‘at-most-one’ constraints, could significantly improve on existing approaches. Such sort of constraints are often needed in applications to describe some *functional* behaviour. One of my plans for the next year is to pursue this line of research.

A more in depth study of alternative (non-grounding) methods, such as those proposed by Baumgartner and Tinelli [1], is also scheduled in my research plan. A couple of ideas that are also considered for further exploration include the use of model generation techniques in the context of databases, and the encoding of planning problems in terms of effectively propositional formulae.

A Gantt chart with a plan for my remaining research, Figure 1, and an outline of a tentative structure for my thesis are included in the following page.

## CONCLUSIONS

In this report we have presented a short summary of the work performed during my second year of research studies. Some evaluation of the progress made and of the work still remaining to do was also presented. A plan to carry out during my final year of PhD studies was finally laid down.

## REFERENCES

- [1] P. Baumgartner and C. Tinelli. The model evolution calculus. In F. Baader, editor, *Proc. 19th CADE*, LNAI 2741, pages 350–364, 2003.
- [2] K. Claessen and N. Sörensson. New techniques that improve MACE-style model finding. In *Proc. Workshop Model Computation*, 2003.
- [3] N. Eén and N. Sörensson. An extensible SAT-solver. In *Proc. SAT’03*, LNCS 2919, pages 502–518, Jan. 2004.
- [4] U. Hustadt et al. Reducing  $\mathcal{SHIQ}^-$  description logic to disjunctive datalog programs. In D. Dubois et al., editors, *Proc. KR 2004*, pages 152–162, June 2004.
- [5] J. A. Navarro. Translations to propositional satisfiability. In *Proc. ARW’06*, Apr. 2006.
- [6] S. Schulz. A comparison of different techniques for grounding near-propositional CNF formulae. In *Proc. AAAI-02*, pages 72–76, 2002.
- [7] G. Sutcliffe and C. Suttner. The TPTP problem library: CNF release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.

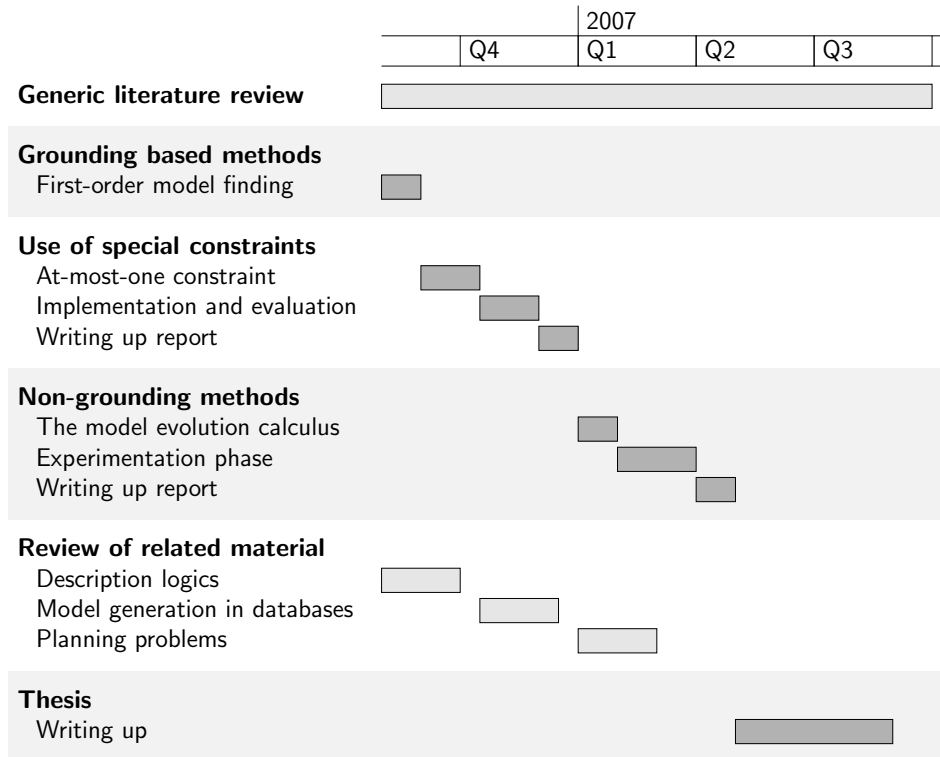


Figure 1: Research plan for third year

## ANTICIPATED THESIS STRUCTURE

- 1 Introduction
  - 1.1 Using logic to solve problems
  - 1.2 The effectively propositional fragment
  - 1.3 Scope and contributions
  - 1.4 Structure of the thesis
- 2 Propositional satisfiability
  - 2.1 Encoding the problem
  - 2.2 Forms of preprocessing
  - 2.3 Searching for solutions
    - 2.3.1 Systematic search with DPLL
    - 2.3.2 Stochastic search strategies
- 3 First-order model finding
  - 3.1 Encoding the problem
  - 3.2 Removing function symbols
  - 3.3 Forms of preprocessing
  - 3.4 Generating ground instances
    - 3.4.1 Incremental and complete grounding
    - 3.4.2 Refinements using sort inference
    - 3.4.3 Refinements using linking conditions
  - 3.5 Solving without grounding
- 4 Solving propositional problems
  - 4.1 What makes a problem difficult to solve?
    - 4.1.1 Generating hard syntactic problems
    - 4.1.2 Properties of *real life* problems
    - 4.1.3 The impact of problem encodings
  - 4.2 The use of special constraints
    - 4.2.1 Parity constraints
    - 4.2.2 At-most-one constraints
- 5 Solving problems with grounding
  - 5.1 Incremental vs. complete grounding
  - 5.2 Enhancing the linking conditions
  - 5.3 Using special constraints
- 6 Solving problems without grounding
  - 6.1 The model evolution calculus
  - 6.2 Integration with other approaches
  - 6.3 Evaluation and comparison
- 7 Case studies in applications
  - 7.1 Effectively propositional formulae
  - 7.2 Bounded model checking
  - 7.3 Description logics
- 8 Conclusions and future work
  - 8.1 Contributions
  - 8.2 Future work