# Compression of Propositional Resolution Proofs by Lowering Subproofs

Joseph Boudou[1] *and Bruno Woltzenlogel Paleo[2] †

[1] Université Paul Sabatier, Toulouse
joseph.boudou@matabio.net
[2] Vienna University of Technology
bruno@logic.at

Sat-solvers are among the most efficient automated deduction tools available today. Leveraging this efficiency, sat-solvers have been embedded into various other automated deduction tools, such as SMT-solvers, interactive proof assistants and automated first-order and even higher-order theorem provers. In such a scenario, proofs of unsastifiability produced by the sat-solver must be analysed by the frontend tools, and therefore the quality and size of proofs has become of critical importance.

Techniques to compress propositional resolution proofs in a post-processing stage have recently been proposed. `RecyclePivotsWithIntersection` and `LowerUnits` [**?**] are two examples of such proof compression algorithms. They both achieve good compression ratios in linear time w.r.t. the number of resolution steps. The former reduces irregularity, as defined by Tseitin [**?**]. The latter extends the concept of irregularity to redundancies across branches (*horizontal irregularity*) and reduces it by lowering subproofs of units (i.e. clauses with a single literal). Experiments [**?**] showed that sequentialy composing these two algorithms combines their proof compression power.

In this talk, we will describe a new proof compression algorithm, called `LowerUnivalents` [1], which extends `LowerUnits`. It achieves two goals. Firstly, by lowering more subproofs, it is able to compress more than `LowerUnits`. Secondly, it makes it possible to implement a non-sequential combination of `LowerUnivalents` after `RecyclePivotsWithIntersection`, which is both faster and more compressing than the sequential composition of `LowerUnits` after `RecyclePivotsWithIntersection`.

The principle of `LowerUnivalents` is to take already lowered subproofs into account in order to allow more proofs to be lowered. For instance, if a subproof of $p$ has already been selected for lowering, a subproof of $\neg p, q$ may be lowered above the previous subproofs, provided its lowering would not introduce unwanted literals in the conclusion. In the talk, the formal conditions for such lowering will be exposed along with practical optimizations that allow the algorithm to reduce both traditional and horizontal irregularities.

*Supported by the Google Summer of Code 2012 program.
†Supported by the Austrian Science Fund, project P24300.
[1] A full article describing this algorithm is available at `http://www.matabio.net/univalent.pdf` and has just been submitted to a conference.