SAT 2013
Theory and Applications of Satisfiability Testing
16th International Conference July 8-12, 2013, Helsinki, Finland

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

MUS Extraction with Interactive Choice of Candidates

Johannes Dellert, Christian Zielke, Michael Kaufmann
University of Tuebingen, Germany

Introduction

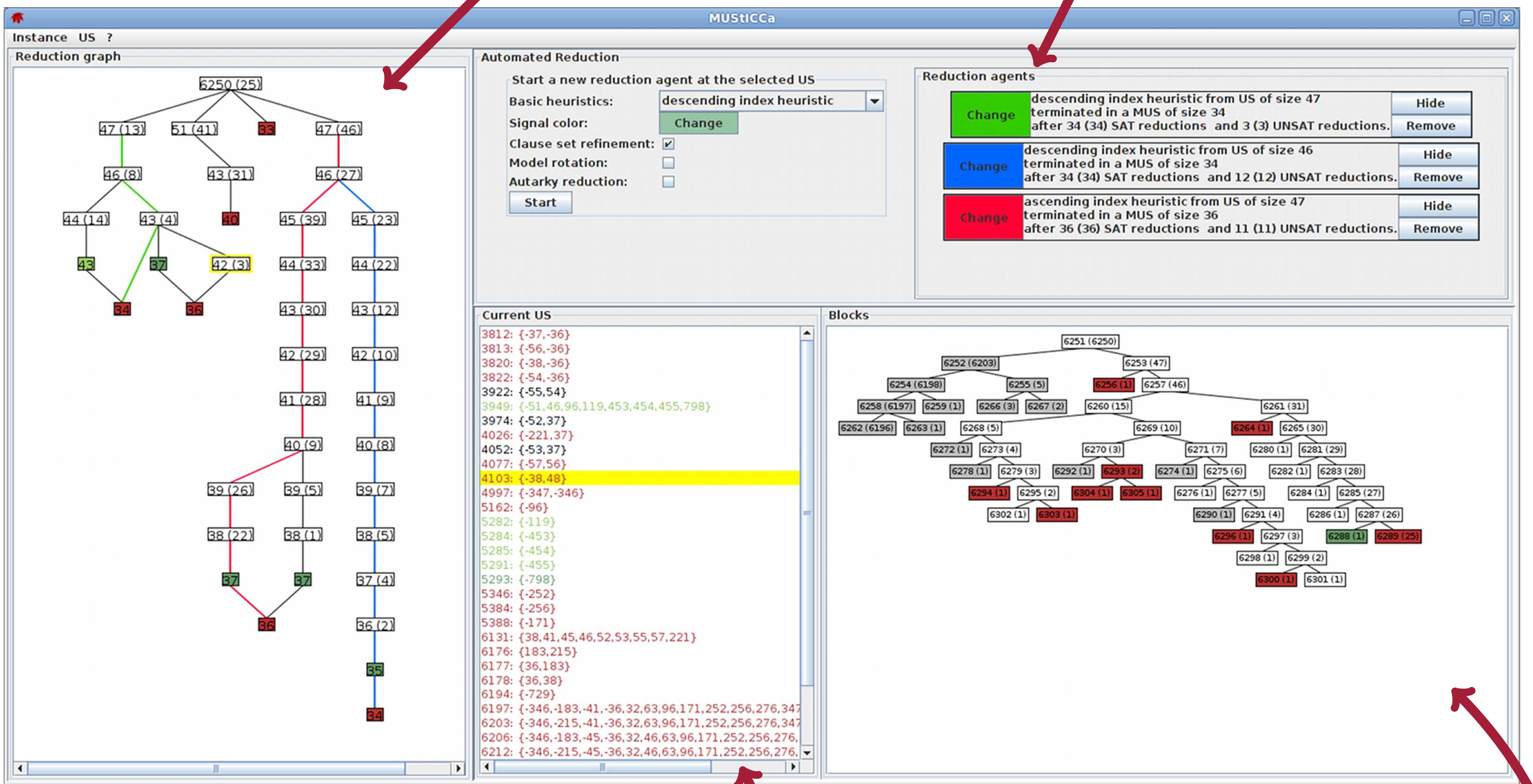
With a remarkable amount of recent work (e.g. [3, 1, 2]) and a broad range of applications (e.g. [4, 5]), the field of minimal unsatisfiable subset (MUS) extraction has become an emerging research field in the SAT community.
We introduce **interactive MUS extraction**, a completely new approach to guiding the destructive MUS extraction algorithm by Nadel [6]. The user receives full control over the individual reduction steps, moving into smaller unsatisfiable subsets F' until all subsets $F'' \subset F'$ are satisfiable. We explicitly visualize the search space, and provide an interface for interactively focusing on or eliminating subspaces by reverting to intermediate results, and exploring new parts of the search space by choosing alternative deletion candidates.

Reduction Graph

The interface is based on the open-source Kahina framework [9] for graphical debugging. In the visualization of the reduction graph, the number on each node gives the size of the corresponding US, followed in brackets by the number of clauses of unknown criticality. MUSes are marked in red, dark green marks non-MUSes where all reduction options have been explored, and light green colour marks USes where all clauses are known to be either critical or unnecessary, but some unexplored reduction options remain.
Whenever a node in the reduction graph view is selected, MUStICCa uses unit-propagation to derive which clauses in it are implied to be critical, and the US corresponding to the node is displayed in the US view.

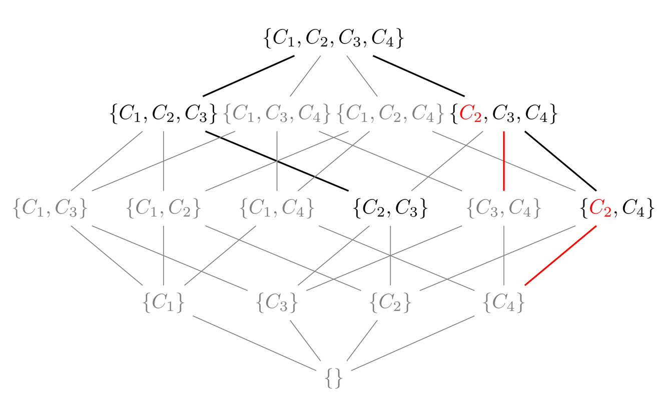
Reduction Agents

MUStICCa includes an automated reduction mechanism in the form of **reduction agents**, comparable to autonomous additional users that are given sets of simple instructions. This is useful for testing new reduction heuristics.
In the dialog for creating and starting new reduction agents the user can select one of the predefined heuristics from a drop-down menu. Clause set refinement [6], model rotation [11] and autarky pruning [10] can be activated or deactivated independently. The new reduction agent starts at the US that is currently selected in the reduction graph, and runs until it has determined a MUS. The downward path of an agent through the powerset lattice is visualized in the form of an **agent trace** highlighted in its randomly chosen signal colour.



Internal Representation

Consider the powerset lattice of $\{C_1, C_2, C_3, C_4\}$. Assume a reduction graph that spans the edges coloured in **black**, where the grey edges and subsets are still unexplored. In the subset $\{C_2, C_3, C_4\}$, the clause C_2 is critical, which we symbolize by colouring critical clauses and the corresponding transition edges in red. We can propagate the criticality of C_2 downwards to the subset $\{C_2, C_4\}$, but we lose the information, that C_2 is critical in other subsets as well, e.g. in the already explored US $\{C_2, C_3\}$.



To overcome this we use boolean **constraints over selector variables** in a so-called **meta-instance**. The criticality information of C_2 depends on all previously deleted clauses, leading to the meta-constraint $\overline{s_1} \rightarrow s_2$ in our example. This expression can directly be written as a single clause $(s_1 \vee s_2)$, representing the fact that either C_1 or C_2 have to be present in each MUS.
Solving the meta-instance with $\overline{s_1} \wedge \overline{s_4}$ as assumptions - denoting both incoming edges of the node $\{C_2, C_3\}$ in the powerset lattice - will require s_2 to be *true* or in other words C_2 to be critical in $\{C_2, C_3\}$.

US View

The clauses in the US view are colour-coded: critical clauses are displayed in red, explicitly reduced clauses in a dark green, other unnecessary clauses in a lighter green, and clauses of unknown status in **black**. The colour codes make it easy to spot interesting deletion candidates for reduction steps, which are started by double-clicking on clauses in the US view.
For advanced interactions, a set of clauses in the US view can be selected and refined via a hierarchy of context menus, to either initiate a series of deletion attempts or a single attempt to delete all selected clauses at once.

Block View

The **meta-instance** is compressed using an inferred **block structure** over the input clauses. Assume that the clauses $(s_1, \ldots s_i, s_k)$ and $(s_1, \ldots s_i, s_l)$ were added. To save space, we insert a fresh variable b_i representing a **block** of variables, and obtain the following clauses: $(s_1, \ldots s_i, b_i)$, $(\overline{b_i}, s_k)$ and $(\overline{b_i}, s_l)$. The block structure visualizes the overlaps between encountered USes, often revealing interesting structural features of the input instance.

References

[1] Belov, A., Lynce, I., Marques-Silva, J.: Towards efficient MUS extraction. AI Commun. 25(2), 97–116 (2012)
[2] Wieringa, S.: Understanding, Improving and Parallelizing MUS Finding Using Model Rotation. In: CP. 672–687 (2012)
[3] Ryvchin, V., Strichman, O.: Faster Extraction of High-Level Minimal Unsatisfiable Cores. In: SAT. 174–187 (2011)
[4] Sinz, C., Kaiser, A., Küchlin, W.: Formal Methods for the Validation of Automotive Product Configuration Data. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 17, 75–97 (2003)
[5] Oh, Y., Mneimneh, M.N., Andraus, Z.S., Sakallah, K.A., Markov, I.L.: AMUSE: a minimally-unsatisfiable subformula extractor. In: DAC. 518–523 (2004)
[6] Nadel, A.: Boosting minimal unsatisfiable core extraction. In: FMCAD. pp. 221–229 (2010)
[7] Bakker, R.R., Dikker, F., Tempelman, F., Wognum, P.: Diagnosing and Solving Over-Determined Constraint Satisfaction Problems. In: Proceedings of IJCAI-93. pp. 276–281.
[8] Chinneck, J.W., Dravnieks, E.W.: Locating Minimal Infeasible Constraint Sets in Linear Programs. INFORMS Journal on Computing 3(2), 157–168 (1991)
[9] Evang, K., Dellert, J.: Kahina - Trac. Web (2013), http://www.kahina.org/trac
[10] Kullmann, O.: On the use of autarkies for satisfiability decision. Electronic Notes in Discrete Mathematics 9, 231–253 (2001)
[11] Joao Marques-Silva and Ines Lynce. On improving MUS extraction algorithms. In SAT, 159–173, (2011)

http://algo.inf.uni-tuebingen.de/?site=forschung/sat/MUStICCa | This work was supported by DFG-SPP 1307, project "Structure-based Algorithm Engineering for SAT-Solving"