

On SAT representations of XOR constraints (towards a theory of good SAT representations)

Oliver Kullmann

Computer Science Department
Swansea University

<http://cs.swan.ac.uk/~csoliver/papers.html>

Horn formulas, directed hypergraphs, lattices and closure systems
May 13, 2014

Good representations

A fundamental problem is to integrate
special reasoning of the problem domain
into SAT solving.

- Constraint solving (CSP) — that didn't work (for problems with a combinatorial core).
- Integration of such constraint reasoning into SAT solvers — doesn't seem to work currently.
- The SMT (SAT modulo Theory) framework: works, but only in an “industrial setting” (where the problems are big, but not really hard).

Our approach is to start a
Theory of “good” SAT representations
 (“good” representations integrate the special reasoning).

Upper and lower bounds for XOR constraints

Since CNF-representations are a restricted class of representations, we need

- upper and lower bounds on the sizes of “good” representations,
- where “good” can mean many things.

We consider the “case study” (interesting in its own rights) of

systems of XOR constraints
(parity constraints, systems of linear equations over \mathbb{Z}_2).

We show

- There is no good poly-size representation for the general case.
- But when considering parameters (like the number of equations), parts of a complex landscape with many non-trivial possibilities unfold (resp. we hope so).

Other approaches at intelligent XOR-translations

- While we show fpt in the number m of XOR-constraints, the weaker parameter n , the number of variables, was show fpt in [Laitinen, Junttila, and Niemelä \[27\]](#).
- Practical results (SAT benchmarks) for translating XOR-clause-sets into CNF-clause-sets are in [Laitinen, Junttila, and Niemelä \[26\]](#).
- These authors also introduced the DPLL(XOR) framework, for integrating dedicated XOR-reasoning into SAT solving ([Laitinen, Junttila, and Niemelä \[24, 25\]](#)).

The project: a theory of SAT representations

- These “hardness measures” for proof complexity: [Beyersdorff and Kullmann \[5, 6\]](#) (SAT 2014; [click](#)).
- The LATA 2014 paper [Gwynne and Kullmann \[20\]](#) on XOR ([click](#)), with underlying (arXiv) report [Gwynne and Kullmann \[18\]](#) ([click](#)).
- SOFSEM 2013 ([click](#)) and JAR ([click](#)) for the basic “hardness measures”, measuring the “quality” of a representation: [Gwynne and Kullmann \[16, 19\]](#)
- Trading quality for size, showing that the various hardness measures yield hierarchies for the representation of boolean functions: [Gwynne and Kullmann \[17\]](#) (arXiv; [click](#))

Outline

- 1 Introduction
- 2 Basics of XOR
- 3 Hardness measures
 - Hardness via Horton-Strahler
 - Generalised unit-clause propagation
 - Hardness via GUCP
- 4 No short good representations
- 5 FPT results
 - Forced assignments and p-hardness
 - Positive results
- 6 Conclusion

XOR-constraints I

We want to construct a “SAT representation” of some problem, which includes an XOR-constraint

$$x_1 \oplus \cdots \oplus x_n = \varepsilon, \quad x_i \in \mathcal{LIT}, \quad \varepsilon \in \{0, 1\}.$$

To make life easier, we assume $\varepsilon = 0$, and we handle that XOR-constraint simply as an

$$\mathbf{XOR-clause} \ C := \{x_1, \dots, x_n\} \in \mathcal{CL}.$$

And a system of XOR-constraint is simple handled as an

$$\mathbf{XOR-clause-set} \ F \in \mathcal{CLS}.$$

The trivial representation of XOR-constraints I

There is precisely one CNF-clause-set, which is equivalent to an XOR-clause C , and we denote it by $X_0(C) \in \mathcal{CLS}$.

$$X_0(\{a\}) = \{\{\bar{a}\}\}$$

$$X_0(\{a, b\}) = \{\{\bar{a}, b\}, \{a, \bar{b}\}\}$$

$$X_0(\{a, b, c\}) = \{\{\bar{a}, b, c\}, \{a, \bar{b}, c\}, \{a, b, \bar{c}\}, \{\bar{a}, \bar{b}, \bar{c}\}\}$$

- $X_0(C)$ has 2^{n-1} clauses of length n .
- $X_0(C)$ is perfect for small n .

We can use X_0 piecewise, obtaining the first general translation:

$$X_0 : \mathcal{CLS} \rightarrow \mathcal{CLS}$$

$$X_0(F) := \bigcup_{C \in F} X_0(C).$$

The trivial representation of XOR-constraints II

$X_0(C)$ is actually the unique equivalent CNF —
what to do for large n !?!

The standard representation of XOR-constraints I

To obtain a **small** translation for arbitrary XOR-clauses C , we use **auxiliary variables**. We split up C , using new variables y_i for partial sums, e.g. for $C = \{x_1, \dots, x_4\}$:

$$x_1 \oplus x_2 = y_2, \quad y_2 \oplus x_3 = y_3, \quad y_3 \oplus x_4 = 0.$$

In general C is split into an XOR-clause-set F' with $n - 1$ XOR-clauses, and we obtain the representation

$$\mathbf{X}_1(\mathbf{C}) := X_0(F') \in 3\text{-CLS}.$$

We got

$$\mathbf{X}_1 : \mathcal{CL} \rightarrow 3\text{-CLS}.$$

The standard representation of XOR-constraints II

So we can represent a single XOR-constraint. If we have many of them, we apply the translation piecewise, obtaining

$$\mathbf{X}_1 : \mathcal{CLS} \rightarrow 3\text{-}\mathcal{CLS}.$$

That is, for a general XOR-clause-set $F \in \mathcal{CLS}$ we get the representation

$$\mathbf{X}_1(F) := \bigcup_{C \in F} \mathbf{X}_1(C) \in 3\text{-}\mathcal{CLS},$$

where new variables are used for the different XOR-clauses in F .

How good is this representation?

We now have a representation $X_1(F) \in 3\text{-}\mathcal{CLS}$ for arbitrary sets F of XOR-clauses.

- This is the default representation, used nearly everywhere.
- But is it “good” ?
- And can we do “better” ?!

Isn't it trivial?

Why can't we just solve F (Gaussian elimination), and that's it?

The point is that there are other “constraints” in the complete SAT problem (besides the XOR-constraints)

— and thus we need to represent *precisely* the solutions of F .

A **CNF-representation** F of a boolean function f means:

- $\text{var}(f) \subseteq \text{var}(F)$,
- the satisfying assignments of F projected to $\text{var}(f)$ are precisely the satisfying assignments of f .

The variables in $\text{var}(F) \setminus \text{var}(f)$ are the **auxiliary variables**.

f is the existential quantification of F over the auxiliary variables.

(See [Bubeck and Büning \[8\]](#) for general results on auxiliary variables.)

Basic linear algebra, or?

Alright, then:

Given an XOR-clause-set F —
just compute a basis of the solution space!

HOWEVER, we want to *recognise* a solution when given to the representation, and for that a basis is just in the way.

But then $X_1(F)$ does already the job ?!

- Unit-clause propagation r_1 is a basic inference mechanism (explained later), computable in linear time.
- r_1 suffices for a total assignment to the variables of F to evaluate $X_1(F)$ to TRUE or FALSE.

Partial instantiation

A SAT solver does not work with *total assignments*, but with
partial assignments.

So the representation $X_1(F)$ will become partially instantiated.

- It might become unsatisfiable.
- It is known, starting with the seminal work of Tseitin [32], that these unsatisfiable problems in general are hard for resolution, and thus for SAT solvers.

The task is thus:

Making the representation fully *explicit*,
so that every unsatisfiable partial instantiation is recognised
by a simple mechanism like unit-clause propagation.

So then online Gaussian elimination?

Alright, we then run Gaussian elimination again and again, after every update to the partial assignment.

- That's precisely what CSP is about.
- One needs then to expand Gaussian elimination by update- and undo-operations (which has been done).

But we **want more!**

We want **one** CNF-representation which does the job (modulo a simple operation like unit-clause propagation)!

- In this way we can use a standard SAT solver.
- MORE IMPORTANTLY, the CNF-representation is fully instantiatable, and gives a meaningful result for **all partial instantiations** (also the satisfiable ones).

Resolution

First we need “resolution trees”:

- 1 The resolution rule, the fundamental proof (refutation) rule for clause-sets, is

$$\frac{C \cup \{x\} \quad D \cup \{\bar{x}\}}{C \cup D}$$

for clauses C, D and literals x with $x \notin C, \bar{x} \notin D$.

- 2 A resolution tree

$$T : F \vdash \perp,$$

derives the empty clause $\perp := \emptyset \in \mathcal{CL}$ from $F \in \mathcal{CLS}$ via resolution.

A clause-set F is unsatisfiable ($F \in \mathcal{USAT}$) iff there is $T : F \vdash \perp$.

Hardness of unsatisfiable clause-sets

As introduced in Kullmann [22, 23] and further expanded in Gwynne and Kullmann [16, 19]:

Definition

For $F \in \mathcal{USAT}$ the **(tree-)hardness** $\text{hd}(F) \in \mathbb{N}_0$ is the minimum of **Horton-Strahler numbers** $\text{hs}(T)$ for $T : F \vdash \perp$.

Special cases are:

- $\text{hd}(F) = 0 \Leftrightarrow \perp \in F$.
- $\text{hd}(F) \leq 1$ iff F can be refuted by **input resolution** (that's $T : F \vdash \perp$ with $\text{hs}(T) \leq 1$).

From Torán [31], Esteban and Torán [14] it follows that $\text{hd}(F) + 1$ is the **space complexity** of tree resolution for F .

Generalised Horn clause-sets

- “Hardness” has many sources (see [22, 5, 6] for the history).
- I came from hierarchies of generalised Horn clause-sets:
 - ① Yamasaki and Doshita [33],
 - ② Gallo and Scutellà [15],
 - ③ Dalal and Etherington [11],
 - ④ Kleine Büning [21],
 - ⑤ Eiter, Kilpeläinen, and Mannila [13],
 - ⑥ Pretolani [29].

This stream of thought was continued with

- ① Čepek and Kučera [9],
- ② Čepek, Kučera, and Vlček [10]).

Hardness on unsatisfiable(!) clause-sets
is the level of the underlying algorithms for deciding SAT
for generalised Horn clause-sets.

Hardness for satisfiable clause-sets

As studied in [16, 19] (first mentioned in Ansótegui, Bonet, Levy, and Manyà [1]):

Definition

For $F \in \mathcal{CLS}$ we define $\mathbf{hd}(F) \in \mathbb{N}_0$ as the maximum of $\mathbf{hd}(F')$ over all $F' \in \mathcal{USAT}$ obtained from F by partial instantiation of some variables.

A “good representation” of a boolean function f basically means a representation F with “small” $\mathbf{hd}(F)$.

SLUR

Most fundamental the class

$$UC := UC_1 := \{F \in \mathcal{CLS} : \text{hd}(F) \leq 1\}.$$

- The class UC has been introduced in del Val [12] for the purpose of Knowledge Compilation.
- Strengthening UC_0 , where $F \in UC_0$ iff F is the set of all *prime implicates* of some boolean function (mod subsumption).

My student Matthew Gwynne and me showed ([16, 19]):

$$UC = SLUR$$

(as introduced by Schlipf, Annexstein, Franco, and Swaminathan [30]; in Āeppek et al. [10] it was shown that membership decision for $SLUR$ is coNP-complete).

Relative hardness

Definition

For a set V of variables we define the relative hardness

$$\mathbf{hd}^V(\mathbf{F}) \in \mathbb{N}_0$$

by considering only assignments to variables in V (regarding the instantiations, yielding those unsatisfiable F').

For a representation F of f we say:

- F is a representation of **absolute hardness** k iff $\mathbf{hd}(F) = k$,
- while F is a representation of **relative hardness** k iff $\mathbf{hd}^{\text{var}(f)}(F) = k$.

Unit-clause propagation

A basic mechanism in determining satisfiability is

unit-clause propagation (UCP).

- Detects and sets some forced assignments, repeatedly.
- Possible in linear time, and is confluent.
- Using the map $r_1 : \mathcal{CLS} \rightarrow \mathcal{CLS}$ for UCP we have

$$r_1(F) := \begin{cases} \{\perp\} & \text{if } \perp \in F \\ r_1(\langle x \rightarrow 1 \rangle * F) & \text{if } \exists x \in \text{lit}(F) : \perp \in \langle x \rightarrow 0 \rangle * F . \\ F & \text{otherwise} \end{cases}$$

Generalised unit-clause propagation

[22, 23] introduced the notion of

generalised unit-clause propagation

$$r_k : \mathcal{CLS} \rightarrow \mathcal{CLS}, k \in \mathbb{N}_0.$$

$$r_0(F) := \begin{cases} \{\perp\} & \text{if } \perp \in F \\ F & \text{otherwise} \end{cases}$$

$$r_k(F) := \begin{cases} r_k(\langle x \rightarrow 1 \rangle * F) & \text{if } \exists x \in \text{lit}(F) : r_{k-1}(\langle x \rightarrow 0 \rangle * F) = \{\perp\} \\ F & \text{otherwise} \end{cases}.$$

$r_k(F)$ can be computed in time $\ell(F) \cdot n(F)^{2k-2}$.

$r_2 : \mathcal{CLS} \rightarrow \mathcal{CLS}$ is (full) *failed literal elimination*.

Hardness via r_k

As shown in [22, 23]:

Theorem

For $F \in \text{USAT}$ we have

$$\text{hd}(F) = \min\{k \in \mathbb{N}_0 : r_k(F) = \{\perp\}\}.$$

Monotonisation of boolean functions

Consider a boolean function f .

We want partial assignments to f ,
handled by a boolean function \widehat{f} .

- Every variable is doubled.
- So we can encode “not assigned”.

Now

$\widehat{f} = 0$ iff
the corresponding partial assignment
makes f unsatisfiable.

Example: the monotonisation of the bijective PHP_m^m function is the matching function (essentially).

Monotone circuits

Theorem

Consider a boolean function f and a representation F with

$$\text{hd}^{\text{var}(f)}(F) \leq 1.$$

From F we can compute in time $O(\ell(F) \cdot n(F)^2)$ a monotone circuit computing \hat{f} .

Corollary

Boolean functions f_n have a CNF-representation F_n with $\text{hd}^{\text{var}(f_n)}(F_n) \leq 1$ and $\ell(F_n) = n^{O(1)}$ if and only if \hat{f}_n can be computed by monotone circuits of size polynomial in n .

(The predecessor of these results is [Bessiere, Katsirelos, Narodytska, and Walsh \[4\]](#) (with \hat{f} “hidden”).)

No polysize good representations for XOR's

Exploiting Babai, Gál, and Wigderson [2] (monotone span programs):

Theorem

The size of representations of systems of XOR-constraints with bounded relative hardness is super-polynomial in the number of constraints.

Forced assignments

After this basic negative result, we turn to positive results, where we want more than just unsatisfiability-detection:

An assignment $\langle x \rightarrow 1 \rangle$ for a literal x and $F \in \mathcal{CLS}$ is called **forced**, if $\langle x \rightarrow 0 \rangle * F \in \mathcal{USAT}$.

Thus $\langle x \rightarrow 1 \rangle * F$ is sat-equivalent to F .

- So we can (and should!) apply the partial assignment $\langle x \rightarrow 1 \rangle$.
- Detection of a forced assignment is coNP-complete.
- So special cases need to be considered.
- The r_k detect and eliminate some forced assignments.
- With $k = n(F)$ we get all forced assignments.

Propagation hardness

Let $r_\infty(F) := r_{n(F)}(F)$, that is, r_∞ applies all forced assignments.

Now $\mathbf{phd}(F) \in \mathbb{N}_0$ (“p-hardness”) for $F \in \mathcal{CLS}$ is the

smallest k such that for all partial assignments φ
we have $r_\infty(\varphi * F) = r_k(\varphi * F)$.

Let $\mathcal{PC}_k := \{F \in \mathcal{CLS} : \mathbf{phd}(F) \leq k\}$.

- $\mathcal{UC}_{k-1} \subset \mathcal{PC}_k \subset \mathcal{UC}_k$.
- $\mathcal{PC}_1 = \mathcal{PC}$ was introduced in Pipatsrisawat and Darwiche [28], Bordeaux and Marques-Silva [7] (**unit-propagation complete**).
- In Babka, Balyo, Čepek, Štefan Gurský, Kučera, and Vlček [3] it is shown that membership decision for \mathcal{PC} is coNP-complete.

Relative p-hardness

For the definition of $\text{phd}^V(F)$ for $F \in \mathcal{CLS}$ and a set V of variables one has to be a bit careful:

- 1 First the partial assignments φ are restricted to $\text{var}(\varphi) \subseteq V$.
- 2 Second also the forced literals x must be restricted to $\text{var}(x) \in V$.

That is, for **relative p-hardness** k we only say that forced literals *over the original variables* are realised via r_k .

Forced literals as closure system (via implications)

Added after the talk:

For a boolean function f we consider $\text{lit}(f) := \text{var}(f) \cup \overline{\text{var}(f)}$ (all possible literals), and the following *closure operator* on $\text{lit}(f)$, given via implications $P \rightarrow Q$ for $P, Q \subseteq \text{lit}(f)$:

- If $v, \bar{v} \in P$ for some $v \in \text{var}(f)$, then $Q := \text{lit}(f)$.
- Otherwise let φ set all literals of P to 1.
- If φ makes f unsatisfiable, then $Q := \text{lit}(f)$.
- Otherwise, Q is the set of literals which are forced to true in order to extend φ to a (total) satisfying assignment of f .

This closure system represents f (exactly).

$\text{phd}^{\text{var}(f)}(F)$ of a representation F
is the smallest k such that this closure operator
can be computed via r_k .

FPT in number of XOR-constraints I

Theorem

By adding all implied XOR-clauses to a system of m XOR-clauses, translating each via X_1 , we obtain a representation of relative p -hardness 1, with running time fixed-parameter tractable (fpt) in m (i.e., running-time 2^m times a polynomial in the length of the system).

We believe that this can be strengthened in two dimensions:

- Instead of the “relative condition”, we can obtain the “absolute condition” \mathcal{PC} .
- Instead of fpt in m , we can obtain fpt in the treewidth of the incidence graph.

FPT in number of XOR-constraints II

As a preliminary result in this direction, we can handle $m = 2$:

Lemma

By factoring out the common part of two XOR-clauses, we obtain a translation for $m = 2$ to \mathcal{PC} in linear time.

Summary and outlook

- I We believe there is a whole world to be discovered.
- II Hopefully a theory of “good SAT representations” will emerge which truly brings theory (boolean functions) and practice (SAT solving) together.
- III The translation of XOR-systems is a good first test-case: Despite the bad news “no poly-size good representation”, there seem to be a lot of opportunities for good representations (under various circumstances).

End

(references on the remaining slides).

For my papers see

<http://cs.swan.ac.uk/~csoliver/papers.html>.

Bibliography I

- [1] Carlos Ansótegui, María Luisa Bonet, Jordi Levy, and Felip Manyà. Measuring the hardness of SAT instances. In Dieter Fox and Carla Gomes, editors, *Proceedings of the 23th AAAI Conference on Artificial Intelligence (AAAI-08)*, pages 222–228, 2008.
- [2] László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19 (3):301–319, March 1999.
- [3] Martin Babka, Tomáš Balyo, Ondřej Čepek, Štefan Gurský, Petr Kučera, and Václav Vlček. Complexity issues related to propagation completeness. *Artificial Intelligence*, 203:19–34, 2013.

Bibliography II

- [4] Christian Bessiere, George Katsirelos, Nina Narodytska, and Toby Walsh. Circuit complexity and decompositions of global constraints. In *Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 412–418, 2009.
- [5] Olaf Beyersdorff and Oliver Kullmann. Hardness measures and resolution lower bounds. Technical Report arXiv:1310.7627v2 [cs.CC], arXiv, February 2014.
- [6] Olaf Beyersdorff and Oliver Kullmann. Unified characterisations of resolution hardness measures. In Uwe Egly and Carsten Sinz, editors, *Theory and Applications of Satisfiability Testing - SAT 2014*, Lecture Notes in Computer Science. Springer, 2014.

Bibliography III

- [7] Lucas Bordeaux and Joao Marques-Silva. Knowledge compilation with empowerment. In Mária Bieliková, Gerhard Friedrich, Georg Gottlob, Stefan Katzenbeisser, and György Turán, editors, *SOFSEM 2012: Theory and Practice of Computer Science*, volume 7147 of *Lecture Notes in Computer Science*, pages 612–624. Springer, 2012.
- [8] Uwe Bubeck and Hans Kleine Büning. The power of auxiliary variables for propositional and quantified boolean formulas. *Studies in Logic*, 3(3):1–23, 2010.
- [9] Ondřej Čepek and Petr Kučera. Known and new classes of generalized Horn formulae with polynomial recognition and SAT testing. *Discrete Applied Mathematics*, 149:14–52, 2005.

Bibliography IV

- [10] Ondřej Čepek, Petr Kučera, and Václav Vlček. Properties of SLUR formulae. In Mária Bieliková, Gerhard Friedrich, Georg Gottlob, Stefan Katzenbeisser, and György Turán, editors, *SOFSEM 2012: Theory and Practice of Computer Science*, volume 7147 of *LNCS Lecture Notes in Computer Science*, pages 177–189. Springer, 2012.
- [11] Mukesh Dalal and David W. Etherington. A hierarchy of tractable satisfiability problems. *Information Processing Letters*, 44: 173–180, 1992.
- [12] Alvaro del Val. Tractable databases: How to make propositional unit resolution complete through compilation. In *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 551–561, 1994.

Bibliography V

- [13] Thomas Eiter, Pekka Kilpeläinen, and Heikki Mannila. Recognizing renamable generalized propositional Horn formulas is NP-complete. *Discrete Applied Mathematics*, 59:23–31, 1995.
- [14] Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, November 2001.
- [15] Giorgio Gallo and Maria Grazia Scutellà. Polynomially solvable satisfiability problems. *Information Processing Letters*, 29: 221–227, November 1988.

Bibliography VI

- [16] Matthew Gwynne and Oliver Kullmann. Generalising and unifying SLUR and unit-refutation completeness. In Peter van Emde Boas, Frans C. A. Groen, Giuseppe F. Italiano, Jerzy Nawrocki, and Harald Sack, editors, *SOFSEM 2013: Theory and Practice of Computer Science*, volume 7741 of *Lecture Notes in Computer Science (LNCS)*, pages 220–232. Springer, 2013. doi: 10.1007/978-3-642-35843-2_20.
- [17] Matthew Gwynne and Oliver Kullmann. Trading inference effort versus size in CNF knowledge compilation. Technical Report arXiv:1310.5746v2 [cs.CC], arXiv, November 2013.
- [18] Matthew Gwynne and Oliver Kullmann. On SAT representations of XOR constraints. Technical Report arXiv:1309.3060v4 [cs.CC], arXiv, December 2013.

Bibliography VII

- [19] Matthew Gwynne and Oliver Kullmann. Generalising unit-refutation completeness and SLUR via nested input resolution. *Journal of Automated Reasoning*, 52(1):31–65, January 2014. doi: 10.1007/s10817-013-9275-8.
- [20] Matthew Gwynne and Oliver Kullmann. On SAT representations of XOR constraints. In Adrian-Horia Dediu, Carlos Martín-Vide, José-Luis Sierra, and Bianca Truthe, editors, *LATA 2014: Language and Automata Theory and Applications, 8th International Conference*, volume 8370 of *Lecture Notes in Computer Science (LNCS)*, pages 409–420. Springer, 2014. doi: 10.1007/978-3-319-04921-2_33.
- [21] Hans Kleine Büning. On generalized Horn formulas and k -resolution. *Theoretical Computer Science*, 116:405–413, 1993.

Bibliography VIII

- [22] Oliver Kullmann. Investigating a general hierarchy of polynomially decidable classes of CNF's based on short tree-like resolution proofs. Technical Report TR99-041, Electronic Colloquium on Computational Complexity (ECCC), October 1999.
<http://eccc.hpi-web.de/report/1999/041/>.
- [23] Oliver Kullmann. Upper and lower bounds on the complexity of generalised resolution and generalised constraint satisfaction problems. *Annals of Mathematics and Artificial Intelligence*, 40 (3-4):303–352, March 2004.
- [24] Tero Laitinen, Tommi Junttila, and Ilkka Niemelä. Extending clause learning DPLL with parity reasoning. In Helder Coelho, Rudi Studer, and Michael Wooldridge, editors, *ECAI 2010 – 19th European Conference on Artificial Intelligence*, pages 21–26. IOS Press, 2010.

Bibliography IX

- [25] Tero Laitinen, Tommi Junttila, and Ilkka Niemelä. Extending clause learning SAT solvers with complete parity reasoning. In *ICTAI 2012 – 24th International Conference on Tools with Artificial Intelligence*, pages 65–72, 2012.
- [26] Tero Laitinen, Tommi Junttila, and Ilkka Niemelä. Classifying and propagating parity constraints. In Michela Milano, editor, *Principles and Practice of Constraint Programming – CP 2012*, volume 7514 of *Lecture Notes in Computer Science*, pages 357–372. Springer, 2012.
- [27] Tero Laitinen, Tommi Junttila, and Ilkka Niemelä. Simulating parity reasoning. In Ken McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning – LPAR 2013*, volume 8312 of *Lecture Notes in Computer Science*, pages 568–583. Springer, 2013.

Bibliography X

- [28] Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artificial Intelligence*, 175(2):512–525, 2011.
- [29] Daniele Pretolani. Hierarchies of polynomially solvable satisfiability problems. *Annals of Mathematics and Artificial Intelligence*, 17(3-4):339–357, 1996.
- [30] John S. Schlipf, Fred S. Annexstein, John V. Franco, and R.P. Swaminathan. On finding solutions for extended Horn formulas. *Information Processing Letters*, 54:133–137, 1995.
- [31] Jacobo Torán. Lower bounds for space in resolution. In *Computer Science Logic, 13th International Workshop, CSL'99*, volume 1683 of *Lecture Notes in Computer Science*, pages 362–373. Springer, 1999.

Bibliography XI

- [32] G.S. Tseitin. On the complexity of derivation in propositional calculus. In *Seminars in Mathematics*, volume 8. V.A. Steklov Mathematical Institute, Leningrad, 1968. English translation: *Studies in mathematics and mathematical logic*, Part II (A.O. Slisenko, editor), 1970, pages 115-125.
- [33] Susumu Yamasaki and Shuji Doshita. The satisfiability problem for a class consisting of Horn sentences and some non-Horn sentences in propositional logic. *Information and Control*, 59: 1–12, 1983.