Towards a better understanding of SAT translations

Matthew Gwynne\textsuperscript{1} Oliver Kullmann\textsuperscript{2}

Computer Science, Swansea University, UK

\textsuperscript{1}http://cs.swan.ac.uk/~csmg

\textsuperscript{2}http://cs.swan.ac.uk/~csoliver

LCC 2011
Toronto, June 25, 2011
Towards a better understanding of SAT translations

Oliver Kullmann

Introduction

Hardness

Representations

SAT

Representation

Hypothesis

Finding good representations

AES and DES

SAT translations: case studies and theory

Some remarks on the genesis of this research:

1. We started by translating AES into SAT.
2. Trying to develop good translations, we came up with some general ideas.
3. Since AES yields very hard SAT problems, we investigate currently mainly small-scale AES.
4. And for a better comparison with the SAT-literature, we also considered DES.
5. We spent quite some time on the experiments, but we are still at an early stage (it takes a lot of time).

All is available in the OKlibrary (http://www.ok-sat-library.org).
Towards a better understanding of SAT translations

Oliver Kullmann

Introduction

Hardness

Representations

SAT Representation Hypothesis

Finding good representations

AES and DES

Understanding — but what?!?

We want to understand “what SAT solvers are doing”:

- But do we really want to understand these solvers?
- The solvers behave very erratic.
- I consider it as likely that there are many artefacts.
- And I consider it as likely that especially for our problems there exist much better solvers.
- So the main point of empirical studies, at least at the current stage, should be to point into new directions.

This talk concentrates mostly on the theory side, and the conceptual framework.

See the forthcoming technical report [Gwynne and Kullmann, 2011], where we will then also present extensive experimental data (and their analysis).
Towards a better understanding of SAT translations

Oliver Kullmann

Outline

1. Introduction
2. Hardness
3. Representations
4. SAT Representation Hypothesis
5. Finding good representations
6. AES and DES
Generalised UCP

In [Kullmann, 1999, Kullmann, 2004] the following hierarchy of reductions $r_k : CLS \rightarrow CLS$ has been investigated:

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

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

- $r_1$ is unit-clause propagation (UCP)
- $r_2$ is failed-literal reduction
Hardness for unsatisfiable clause-sets

For unsatisfiable $F$ the **hardness** is defined as

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

We call $F$ **$k$-soft** if $hd(F) \leq k$.

For the tree-resolution complexity $\text{Comp}_{tR}(F)$ (minimum number of leaves in a tree representing a resolution refutation of $F$) we have

$$
hd(F) \leq \log_2 \text{Comp}_{tR}(F) \leq \log_2(n(F) + 1) \cdot hd(F).
$$

$hd(F) \leq k$ holds iff there is a tree-resolution refutation of $F$ where the tree has pebbling complexity at most $k + 1$ (allowing shifting of pebbles).
Generalisation for all clause-sets

In [Kullmann, 1999, Kullmann, 2004] also an algorithmically motivated extension of $\text{hd}(F)$ for all clause-sets $F$ has been introduced and discussed.

Here now we introduce an extension which shall measure how good $F$ is as a representation of some underlying boolean function:

For clause-set $F$ the **hardness** $\text{hd}(F)$ is the smallest $k \in \mathbb{N}_0$ such that for all clauses $C$ with $F \models C$ we have

$$\text{hd}(\langle x \rightarrow 0 : x \in C \rangle * F) \leq k$$

(i.e., using $F \models C \iff F \land \neg C \models \bot$, where $C \sim \bigvee_{x \in C} x$, and thus $\neg C \sim \bigwedge_{x \in C} \neg x$).
Representing boolean functions by CNFs

A boolean function is a map $f : \{0, 1\}^V \rightarrow \{0, 1\}$ for some (finite) set $V$ of variables.

A clause-set $F$ represents $f$ if

- $\text{var}(f) \subseteq \text{var}(F)$
- taking the set of satisfying total assignments for $F$ and restricting it to $V$, we obtain exactly the set of satisfying assignments for $f$.

If $F$ has exactly the same number of satisfying total assignments as $f$, then the representation has the unique extension property (uep).

Remark: In practice all representations seem to have uep — could there be a proof that we need only to consider representations with uep “without loss of power”? 
A different point of view

For a clause-set $F$, the boolean functions represented by $F$ are obtained as follows:

1. Let $f_0$ be the boolean function underlying $F$ (with $\text{var}(f_0) = \text{var}(F)$).
2. Now the boolean functions represented by $F$ are exactly the “1-projections” of $f_0$ to $V \subseteq \text{var}(F)$.
3. Such a projection for an assignment to $V$ yields 1 iff there exists an extension to a satisfying assignment of $F$.
4. So $F$ represents $(0)_{x \in \{0,1\}^\emptyset}$ iff $F$ is unsatisfiable.
5. And $F$ represents $(1)_{x \in \{0,1\}^\emptyset}$ iff $F$ is satisfiable.
The SAT Representation Hypothesis (SRH)

SRH is the following hypothesis under development:

A representation $F$ of a boolean function $f$ is “good” for SAT solving if and only if $F$ has low hardness (and $F$ is not too large).

Two features:

1. A representation $F$ of $f$ with low hardness must allow to derive all clauses which follow from $F$ — not just those which follow from $f$.

2. There is a tradeoff between hardness and the size of the representation.
Towards a better understanding of SAT translations

Oliver Kullmann

Introduction

Hardness

Representations

SAT

Representation Hypothesis

Finding good representations

AES and DES

Low hardness is “knowing the truth-table”

What is the meaning of having low hardness?

• “Knowing” a boolean function means “knowing the truth-table”.

• Similarly, “knowing” a constraint means knowing the satisfying (and falsifying!) assignments.

• In the same vein, now “knowing” means “falsification can be detected by \( r_k \)-reduction”.

So having a representation \( F \) of \( f \) with “low hardness” can be interpreted as a parameterised version of

“\( F \) acting as a constraint”. 
Hardness 1 versus “hyperarc consistency”

In the literature one finds the related notion of “(hyper)arc consistency”:

- This (seems) to mean that for every partial assignment *in the original variables* (that is, \( \text{var}(f) \)) one can find all forced assignments by UCP.
- In contrast, our approach also takes the new variables into account (i.e., \( \text{var}(F) \)).
- Instead of UCP (i.e., \( r_1 \)) we now consider \( r_k \).
- We treat as the central category the detection of mere falsification, not forced assignments.
- The term “(hyper)arc consistency” is not appropriate, since the notion of “constraint” is very fuzzy here.

So we propose to consider our notion of hardness as a good replacement of “hyperarc consistency” (of course, only for SAT translations).
Remarks on Extended Resolution (ER)

- The SRH says: The whole business of Extended Resolution is to construct some (poly-size) $k$-soft representation for appropriate (fixed!) $k \geq 1$.
- We hope that we can demonstrate this for the PHP.
- SRH needs only to consider tree-like resolution, since w.r.t. ER full resolution and tree-like resolution have the same power.

Two natural questions here:

- Can we make the application of our framework more powerful by looking at smaller boolean functions inside the “big” constant-0 function?
- Is splitting on the new variables of importance, or is the sole purpose of the new variables to enable compression of prime implicates via $r_k$-reduction?
Remarks on “too big” boolean functions

- We don’t know the truth-table of DES or AES.
- So we have to decompose the big function into small functions.
- We do not understand how to make a “good decomposition”.
- For this first phase of our investigations, we only considered the obvious decomposition, and apply SRH to the small functions.
A prime implicate of a boolean function $f$ is a clause $C$ with $f \models C$ and $\forall C' \subset C : f \not\models C'$.

And a prime implicate of a clause-set $F$ is a prime implicate of the underlying boolean function.

By $\text{prc}_0(f)$ resp. $\text{prc}_0(F)$ we denote the set of all prime implicates (“0” for unsat – falsifying assignments).

$\text{prc}_0(f)$ is the prototypical representation of $f$ with hardness 0 — in the light of SRH, “all what remains” is to find suitable abbreviations for this set (which is mostly too large for SAT solving).
Prime implicates II

“Smurfs” ([Franco et al., 2004]) yield representations of boolean functions comprising all prime implicates and all prime implicants via a BDD-like approach.

We on the other hand “believe in CNF”.

CNF offer the potential of breaking up the barriers between “constraints”.

And representations by CNFs offer the potential of splitting on new variables.

That is, we break up the black box.
Bases

A basic systematic approach for finding a $k$-soft representation of $f$ is

1. Start with $F := \text{prc}_0(f)$.
2. Repeatedly remove clauses $C \in F$ such that $F$ remains $k$-soft.

A completed such computation yields a $k$-base.

- We have developed some heuristic improvements of this basic algorithm.
- Given the truth-table of $f$ (which we always assume), decision of “$F$ is $k$-base for $f$” is in polytime.
- So finding a $k$-base is a search problem in NP.
- The optimisation problem seems very tough, even for boolean functions with just, say, 8 variables.
The canonical translation: The idea

A class of alternative approaches for finding 1-soft representations of $f$ is based on the following idea:

1. Consider the canonical DNF $\text{DNF}(f)$, consisting of all prime implicants of $f$ (i.e., all satisfying total assignments, as DNF-clauses).

2. Apply the Tseitin translation to $\text{DNF}(f)$.

This yields a 1-soft representation of $f$.

There is more to it than just “Tseitin translation applied to DNF”, and we present a more systematic development.

For DES/AES, the main boolean functions are the “boxes”, which are permutations, and permutations have unique DNFs which are also small.
The semantics: 1-extensions

For a boolean function $f$ and $C \in \text{DNF}(f)$ we consider a new variable $\text{vct}_f(C)$.

The **canonical 1-extension** of $f$ is the boolean function

$$\text{ce}(f) := f \land \bigwedge_{C \in \text{DNF}(f)} \text{vct}_f(C) \leftrightarrow \bigwedge_{x \in C} x.$$ 

A **general canonical representation** of $f$ is a representation of $\text{ce}(f)$ without new variables.

- We believe that it is important to start with the semantical side, the boolean function.
- And not directly jumping to syntactical manipulations — like the Tseitin translation.
- The point here is that there are many general canonical representations!
- And we can apply the ideas underlying the notion of a $k$-base to $\text{ce}(f)$. 

General structure of AES / DES

DES and 128-bit AES are ciphers, encrypting a plaintext $P$ to a ciphertext $C$ using key $K$, where $P$, $C$, $K$ are bit-vectors of the same length:

- In DES $|K| = 64$ (only 56 bits are actually used).
- In AES $|K| = 128$.

So as boolean functions, we have 192-bit resp. 384-bit boolean function.

- DES has 16 rounds, each round involving 8 S-boxes, each a 10-bit boolean function.
- AES has 10 rounds, each involving
  - 16 S-boxes, each a 16-bit boolean function
  - 64 multiplication boxes (16-bit boolean functions)
  - another another 4 S-boxes for the key schedule.
We are exploring “the” complexity of the various boxes, considering the sizes and hardness of the various representations. Some appetisers:

- 8 DES S-boxes:
  1. From 1624 to 2047 prime implicates.
  2. The standard canonical translation has 705 clauses.
  3. The current best 1-bases have 123 to 152 clauses.
  4. The conjectured minimum-size representation have 66 to 69 clauses.
  5. For one box we found a (conjectured) minimum-size representation of hardness 2, otherwise hardness 3.

- AES S-box:
  1. 136253 prime implicates.
  2. The current best 1-base has 4398 clauses.
  3. The standard canonical translation has 4353 clauses.
  4. The (conjectured) minimum size representation has 294 clauses, with (apparent) minimum hardness 4.
Towards a better understanding of SAT translations

Oliver Kullmann

Introduction

Hardness

Representations

SAT

Representation

Hypothesis

Finding good representations

AES and DES

Experimental results

- des($m$): $m$ rounds of DES
- aes($m$, $r$, $c$, $e$):
  - $m$ rounds
  - $r$ rows, $c$ columns ($r$ more “important”)
  - $e$ bits in a “byte”.
- Key, plaintext and ciphertext have the same number of bits $r \cdot c \cdot e$.
- Standard 128-bit AES is aes(10, 4, 4, 8).

| Variant | $|K|$ | minisat-2.2.0 | OKsolver-2002 |
|---------|-----|---------------|---------------|
|         |     | can | 1-base | min | full | can | 1-base | min | full |
| des(3)  | 64  | 3987.84 | 1397.14 | 3220.98 | 5946.08 | 90.17 | 13477.65 | 408094.4 | - |
| aes(20, 1, 1, 4) | 4 | 27.84 | 96.53 | 472.35 | 3856.75 | 1.0 | 6.61 | 16.07 | 1521.34 |
| aes(20, 1, 2, 4) | 8 | 214.83 | 126.91 | 1816.14 | 15310.35 | 37.42 | 64.35 | 341.98 | - |
| aes(10, 1, 1, 8) | 8 | 267.53 | 2129.09 | 163743.8 | - | 1.0 | 200.75 | 1083.65 | - |
| aes(2, 2, 2, 4) | 16 | 10400.22 | 11061.98 | 8606.33 | 18593.56 | 179.16 | 4274.75 | 12013.04 | - |
| aes(1, 2, 1, 8) | 16 | 471.5 | 335.81 | 1578.48 | - | 1.0 | 620.29 | 12638.98 | - |

Figure: Conflicts/nodes used to solve DES/AES variants
Towards a better understanding of SAT translations

Oliver Kullmann

Introduction

Hardness

Representations

SAT

Representation

Hypothesis

Finding good representations

AES and DES

Summary

I We introduced a general notion of “hardness” for clause-sets.

II We sketched the SRH, that is, “good representation” means “low hardness”.

III We introduced two methods for constructing representations of low hardness.

IV We presented very first data on attacking DES and AES using these methods.
Bibliography I


Bibliography II

Investigating a general hierarchy of polynomially decidable classes of CNF’s based on short tree-like resolution proofs.

Upper and lower bounds on the complexity of generalised resolution and generalised constraint satisfaction problems.
End