Mining Propositional Simplification Proofs for Small Validating Clauses

Ian Wehrman   Aaron Stump

Dept. of Computer Science and Engineering
Washington University in Saint Louis
http://cl.cse.wustl.edu/

Third Workshop on Pragmatics of Decision Procedures in Automated Reasoning

Conflict Clauses and Validating Clauses

Small conflict clauses are often important for modern SAT and SMT tool performance: some $A' \subseteq A$ such that

$$A' \Rightarrow (\varphi \Leftrightarrow F)$$

When checking validity, called *validating clauses*: some $A' \subseteq A$ such that

$$A' \Rightarrow (\varphi \Leftrightarrow T)$$

Essential Operation of an SMT Tool

Tools like CVC Lite proceed as follows:

1. pick an atom a from the goal φ to split on
2. decide on its value, e.g. $a \Leftrightarrow T$
3. simplify goal based on this decision:

$$\varphi \overset{a \Leftrightarrow T}{\dashrightarrow} \varphi'$$

4. if $\varphi' = F$: halt,
   if $\varphi' = T$: record a validating clause and backtrack,
   else: goto step 1.

Example of Splitting and Simplification

Example of splitting and simplification of $\varphi := (a \vee b) \wedge c$:

$$(a \vee b) \wedge c \overset{a \Leftrightarrow F}{\dashrightarrow} b \wedge c \overset{b \Leftrightarrow T}{\dashrightarrow} c \overset{c \Leftrightarrow T}{\dashrightarrow} T$$

Assignment with domain $\{a, b, c\}$ is a validating clause

Example of Splitting and Simplification (cont.)

But, assignment with domain $\{b, c\}$ is also a validating clause:

$$(a \vee b) \wedge c \overset{b \Leftrightarrow T}{\dashrightarrow} c \overset{c \Leftrightarrow T}{\dashrightarrow} T$$

Decision $a \Leftrightarrow F$ is redundant

Proofs of Propositional Simplification

SMT tools such as CVC Lite generate *proofs* of simplification

Proofs correspond to step-by-step simplification of the goal to T

Main observation: these proofs can be transformed after generation to find small validating clauses

Rewriting Proofs of Simplification

Given a goal φ and proof of $\varphi \Leftrightarrow T$, reduce the proof with a *term rewriting system* (TRS) to one using fewer decisions

- Proof p of simplification

$$(a \vee b) \wedge c \stackrel{a \Leftrightarrow F}{\dashrightarrow} b \wedge c \stackrel{b \Leftrightarrow T}{\dashrightarrow} c \stackrel{c \Leftrightarrow T}{\dashrightarrow} T$$

... is rewritten to proof p' of simplification

$$(a \vee b) \wedge c \stackrel{b \Leftrightarrow T}{\dashrightarrow} c \stackrel{c \Leftrightarrow T}{\dashrightarrow} T$$

Algebraic Proof Mining

Proofs viewed as first-order terms

Sound equational theory between proofs is defined

Information extracted from algebraically equivalent proof

Here:

- Equations are completed to a convergent TRS
- Proofs are rewritten, then information extracted

More sophisticated mining techniques are future work

Propositional Equivalence Formulas

Goal formulas:

$$\mathcal{S} ::= \ \mathcal{A} \mid (\mathcal{S} \vee \mathcal{S}) \mid (\mathcal{S} \wedge \mathcal{S}) \mid \neg \mathcal{S}$$

Boolean-valued equivalence formulas:

$$\mathcal{E} ::= \mathcal{S} \Leftrightarrow \mathcal{S} \mid \mathcal{S} \Leftrightarrow V$$

$\mathcal{A}$ the set of propositional variables, $V = \{T, F\}$.

First-order Proof Terms

Equivalence proofs:

$$\mathcal{P} ::= \mathcal{U} \mid \text{Refl} \mid \text{Trans}(\mathcal{P}, \mathcal{P}) \mid \text{NotFalse} \mid \text{NotTrue} \mid \text{OrTrue1} \mid \text{OrTrue2} \mid \text{OrFalse1} \mid \text{OrFalse2} \mid \text{CongrNot}(\mathcal{P}) \mid \text{CongrOr1}(\mathcal{P}) \mid \text{CongrOr2}(\mathcal{P})$$

$\mathcal{U}$ a set of atomic proofs (corresponding to decisions)

Meaning of the Proof Terms

Define a binary relation $\vdash$ between formulas and proofs:

Refl	$\vdash$	$c \Leftrightarrow c$	
Trans(p_1, p_2)	$\vdash$	$c \Leftrightarrow c''$	if $p_1 \vdash c \Leftrightarrow c', p_2 \vdash c' \Leftrightarrow c''$
NotTrue	$\vdash$	$\neg T \Leftrightarrow F$	
NotFalse	$\vdash$	$\neg F \Leftrightarrow T$	
OrTrue1	$\vdash$	$T \vee c \Leftrightarrow T$	
OrTrue2	$\vdash$	$c \vee T \Leftrightarrow T$	
OrFalse1	$\vdash$	$F \vee c \Leftrightarrow c$	
OrFalse2	$\vdash$	$c \vee F \Leftrightarrow c$	
CongrOr1(p_1)	$\vdash$	$c \vee b \Leftrightarrow c' \vee b$	if $p_1 \vdash c \Leftrightarrow c'$
CongrOr2(p_1)	$\vdash$	$c \vee b \Leftrightarrow c \vee b'$	if $p_1 \vdash b \Leftrightarrow b'$
CongrNot(p_1)	$\vdash$	$\neg c \Leftrightarrow \neg c'$	if $p_1 \vdash c \Leftrightarrow c'$

An Equational Theory for Proof Reduction

Basic reduction steps are oriented rewrite rules on the proof terms

Rules transform proofs of simplification into canonical form with fewer unnecessary subproofs, decisions

Derivations on the same subformula gathered so large subproofs are dropped by “cut-off” rules

Basic Rewrite Rules

Right-Assoc

$\text{Trans}(\text{Trans}(x_1, x_2), x_3) \rightarrow \text{Trans}(x_1, \text{Trans}(x_2, x_3))$

Trans-Refl

$\text{Trans}(\text{Refl}, x_1) \rightarrow x_1$

$\text{Trans}(x_1, \text{Refl}) \rightarrow x_1$

Congr-Refl

$\text{CongrOr1}(\text{Refl}) \rightarrow \text{Refl}$

$\text{CongrOr2}(\text{Refl}) \rightarrow \text{Refl}$

$\text{CongrNot}(\text{Refl}) \rightarrow \text{Refl}$

Cut-Off

$\text{Trans}(\text{CongrOr1}(x_1), \text{OrTrue2}) \rightarrow \text{OrTrue2}$

$\text{Trans}(\text{CongrOr2}(x_1), \text{OrTrue1}) \rightarrow \text{OrTrue1}$

Congr-Drop

$\text{Trans}(\text{CongrOr2}(x_1), \text{OrFalse1}) \rightarrow \text{Trans}(\text{OrFalse1}, x_1)$

$\text{Trans}(\text{CongrOr1}(x_1), \text{OrFalse2}) \rightarrow \text{Trans}(\text{OrFalse2}, x_1)$

Congr-Pull

$\text{Trans}(\text{Trans}(\text{CongrOr1}(x_1), \text{CongrOr2}(x_2)), \text{Trans}(\text{CongrOr1}(x_3), \text{CongrOr2}(x_4)))$
$\quad \rightarrow \text{Trans}(\text{CongrOr1}(\text{Trans}(x_1, x_3)), \text{CongrOr2}(\text{Trans}(x_2, x_4)))$

$\text{Trans}(\text{CongrNot}(x_1), \text{CongrNot}(x_2)) \rightarrow \text{CongrNot}(\text{Trans}(x_1, x_2))$

Soundness of the TRS

A single proof proves multiple theorems

Write $p_1 \xrightarrow{*} p_2$ to denote any number of rewrite steps in the completed TRS.

Theorem (Soundness)

For all proofs p_1, p_2, if $p_1 \xrightarrow{*} p_2$ then p_2 is "more general" than p_1.

Proof Reduction Example

Rewrite rule in completed TRS:

$$\mathrm{Trans}(\mathrm{CongrOr1}(x_1), \mathrm{Trans}(\mathrm{CongrOr2}(x_2), \mathrm{OrTrue2})) \rightarrow \mathrm{Trans}(\mathrm{CongrOr2}(x_2), \mathrm{OrTrue2})$$

$$\dfrac{\dfrac{\dfrac{(p_1)}{a \Leftrightarrow a'}}{a \vee b \Leftrightarrow a' \vee b}\ \text{CongrOr1} \qquad \dfrac{\dfrac{\dfrac{(p_2)}{b \Leftrightarrow T}}{a' \vee b \Leftrightarrow a' \vee T}\ \text{CongrOr2} \qquad \dfrac{}{a' \vee T \Leftrightarrow T}\ \text{OrTrue2}}{a' \vee b \Leftrightarrow T}\ \text{Trans}}{a \vee b \Leftrightarrow T}\ \text{Trans}$$

$$\downarrow$$

$$\dfrac{\dfrac{\dfrac{(p_2)}{b \Leftrightarrow T}}{a \vee b \Leftrightarrow a \vee T}\ \text{CongrOr2} \qquad \dfrac{}{a \vee T \Leftrightarrow T}\ \text{OrTrue2}}{a \vee b \Leftrightarrow T}\ \text{Trans}$$

Canonical Form of Reduced Proofs

TRS is convergent, but different proofs of a theorem don't always have the same canonical form

$$\dfrac{\dfrac{\dfrac{(p_1)}{a \Leftrightarrow T}}{a \vee b \Leftrightarrow T \vee b}\,\text{CongrOr1} \qquad \dfrac{\dfrac{\dfrac{(p_2)}{b \Leftrightarrow T}}{T \vee b \Leftrightarrow T \vee T}\,\text{CongrOr2} \qquad \dfrac{}{T \vee T \Leftrightarrow T}\,\text{OrTrue2}}{T \vee b \Leftrightarrow T}\,\text{Trans}}{a \vee b \Leftrightarrow T}\,\text{Trans}$$

Only need one of p_1 or p_2, but which one?

Conclusion

- We have described a possible technique for finding small validating clauses
- We use proof mining: proofs are viewed as first-order terms and reduced by a TRS
- Of potential use to SMT tools that rely on clausal form to find small validating clauses
- Validating clauses are not of optimal size but decisions that are clearly unnecessary