flea

bit(e)s and pieces

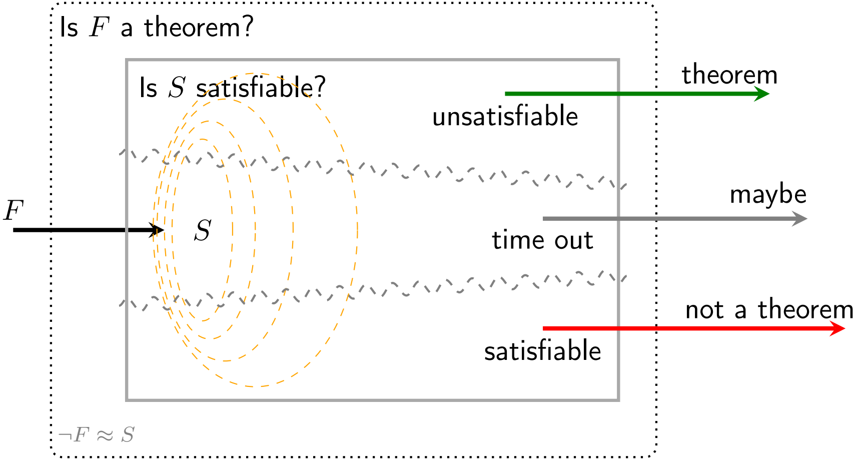
Alexander Maringele

June 15th, 2016

References

Douglas R. Hofstadter, *Gödel, Escher, Bach: An Eternal Golden Braid*, Basic Books, Inc., New York, NY, USA, 1979.

Robert Nieuwenhuis, Thomas Hillenbrand, Alexandre Riazanov, and Andrei Voronkov, *On the evaluation of indexing techniques for theorem proving*, Automated Reasoning (Rajeev Goré, Alexander Leitsch, and Tobias Nipkow, eds.), Lecture Notes in Computer Science, vol. 2083, Springer Berlin Heidelberg, 2001, pp. 257–271.


Is F a theorem?
Is S satisfiable?
theorem
unsatisfiable
maybe
F
S
time out
not a theorem
satisfiable
$\neg F \approx S$

Definition (Ordered Resolution)

$$\frac{L \vee C \quad \neg L' \vee D}{(C \vee D)\sigma}$$

where

$L\sigma$ strictly maximal in $C\sigma$, $\neg L'\sigma$ maximal in $D\sigma$, $\sigma = \mathrm{mgu}(L, L')$.

Definition (Inst-Gen)

$$\frac{L \vee C \quad \neg L' \vee D}{(L \vee C)\sigma \quad (\neg L' \vee D)\sigma}$$

where

$$\mathrm{sel}(L \vee C) = L \qquad \mathrm{sel}(\neg L' \vee D) = \neg L' \qquad \sigma = \mathrm{mgu}(L, L')$$

Example (Resolution)

$$S = \{\mathsf{P}(x) \vee \neg\mathsf{P}(y), \neg\mathsf{P}(\mathsf{a}), \mathsf{P}(\mathsf{b})\}$$

$$\cfrac{\cfrac{\mathsf{P}(x) \quad \neg\mathsf{P}(y)}{\mathsf{P}(y)}\ x \mapsto \mathsf{a} \qquad \mathsf{P}(\mathsf{b})}{\square}\ y \mapsto \mathsf{b}$$

Example (Inst-Gen)

$$S_0\bot = \{\mathsf{P}(\bot) \vee \neg\mathsf{P}(\bot), \neg\mathsf{P}(\mathsf{a}), \mathsf{P}(\mathsf{b})\} \qquad \text{satisfiable}$$

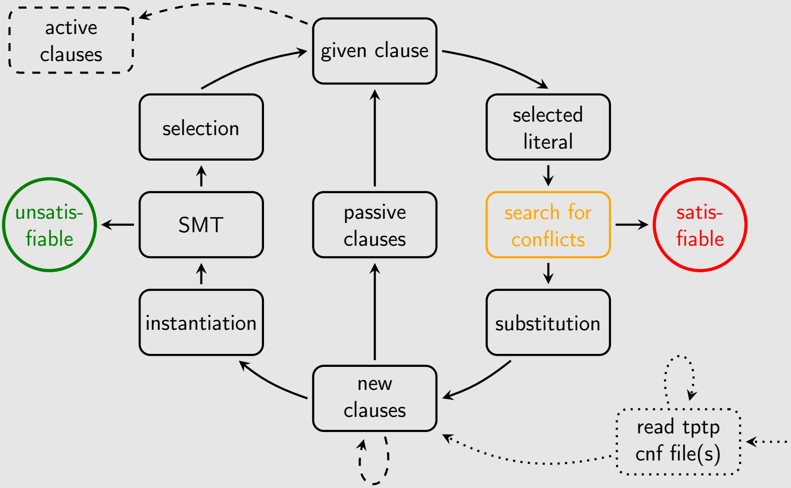
$$\frac{\mathsf{P}(x) \vee \neg\mathsf{P}(y)) \quad \neg\mathsf{P}(\mathsf{a})}{\mathsf{P}(\mathsf{a}) \vee \neg\mathsf{P}(y)}\ x \mapsto \mathsf{a}$$

$$S_1\bot \supsetneq \{\neg\mathsf{P}(\mathsf{a}), \mathsf{P}(\mathsf{b}), \mathsf{P}(\mathsf{a}) \vee \neg\mathsf{P}(\bot)\} \qquad \text{satisfiable}$$

$$\frac{\mathsf{P}(\mathsf{b}) \quad \mathsf{P}(\mathsf{a}) \vee \neg\mathsf{P}(y)}{\mathsf{P}(\mathsf{a}) \vee \mathsf{P}(\mathsf{b})}\ y \mapsto \mathsf{b}$$

$$S_2\bot \supsetneq \{\neg\mathsf{P}(\mathsf{a}), \mathsf{P}(\mathsf{b}), \mathsf{P}(\mathsf{a}) \vee \neg\mathsf{P}(\mathsf{b})\} \qquad \text{unsatisfiable}$$

Run-Loop


active clauses
given clause
selection
selected literal
unsatis-fiable
SMT
passive clauses
search for conflicts
satis-fiable
instantiation
substitution
new clauses
read tptp cnf file(s)

Subsumption

$$S = \{C, D, \ldots\} \qquad \exists\theta\ C\theta \subseteq D$$ C subsumes D

$$S \text{ satisfiable } \overset{\checkmark}{\Longleftrightarrow} (S \setminus D) \text{ satisfiable}$$

$$\theta \text{ is proper, } S\bot \text{ satisfiable } \overset{\times}{\Longleftrightarrow} (S \setminus D)\bot \text{ satisfiable}$$

$$\theta \text{ is renaming, } S\bot \text{ satisfiable } \overset{\checkmark}{\Longleftrightarrow} (S \setminus D)\bot \text{ satisfiable}$$

Example

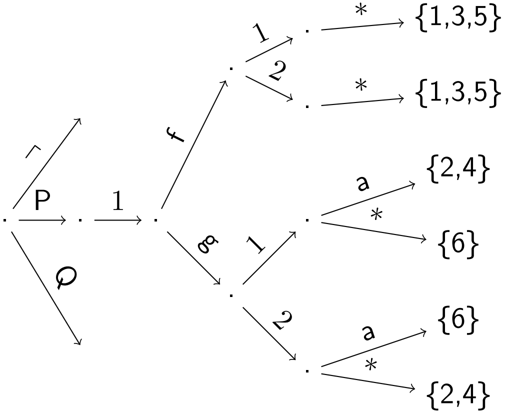
$$\{\mathsf{P}(x, y), \neg\mathsf{P}(\mathsf{a}, z)\} \qquad \{\mathsf{P}(x, y), \neg\mathsf{P}(\mathsf{a}, z), \mathsf{P}(\mathsf{a}, z)\}$$

$$\{\mathsf{P}(\bot, \bot), \neg\mathsf{P}(\mathsf{a}, \bot)\} \qquad \{\mathsf{P}(\bot, \bot), \neg\mathsf{P}(\mathsf{a}, \bot), \mathsf{P}(\mathsf{a}, \bot)\}$$

data structures

1. trees to represent clauses, literals and terms
2. path indexing for fast retrieval of clashing selected literals
3. discrimination trees for fast retrieval of clause variants

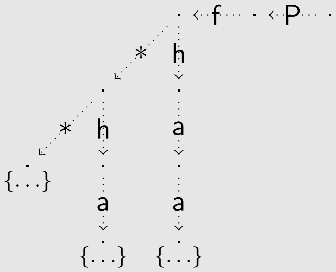
$$\{{}^{1:}\mathsf{P}(\mathsf{f}(x,x)), {}^{2:}\mathsf{P}(\mathsf{g}(\mathsf{a},x)), {}^{3:}\mathsf{P}(\mathsf{f}(y,z)), {}^{4:}\mathsf{P}(\mathsf{g}(\mathsf{a},y)), {}^{5:}\mathsf{P}(\mathsf{f}(y,x)), {}^{6:}\mathsf{P}(\mathsf{g}(y,a))\}$$



$$\neg\mathsf{P}(\mathsf{g}(\mathsf{b},z)) \mapsto \{\mathsf{P}.1.\mathsf{g}.1.\mathsf{b}, \mathsf{P}.1.\mathsf{g}.2.*\} \mapsto \{6\} \cap \{2,4,6\}$$

$^{\ell_1:}\mathsf{P}(\mathsf{f}(x,y)),{}^{\ell_2:}\mathsf{P}(\mathsf{f}(x,\mathsf{h}(\mathsf{a}))),{}^{\ell_3:}\mathsf{P}(\mathsf{f}(\mathsf{h}(\mathsf{a}),\mathsf{a}))$

$$\ell_1 \mapsto \mathsf{P.f.}{*}.{*}$$
$$\ell_2 \mapsto \mathsf{P.f.}{*}\mathsf{.h.a}$$
$$\ell_3 \mapsto \mathsf{P.f.h.a.a}$$



Implementation

```
Clause ↦ (Int, [term_t])
term_t ↦ [Int]
   Int ↦ Clause
```

```c
int32_t yices_assert_formula(context_t *ctx, term_t t);

term_t yices_or(uint32_t n, term_t arg[]);
term_t yices_not(term_t arg);
term_t yices_eq(term_t left, term_t right);

term_t yices_application(term_t fun, uint32_t n, const term_t arg[]);
term_t yices_new_uninterpreted_term(type_t tau);

type_t yices_function_type(uint32_t n, const type_t dom[], type_t range);
type_t yices_bool_type(void);
type_t yices_new_uninterpreted_type(void);

smt_status_t yices_check_context(context_t *ctx, const param_t *params);
model_t *yices_get_model(context_t *ctx, int32_t keep_subst);
int32_t yices_get_bool_value(model_t *mdl, term_t t, int32_t *val);
```

Example

$S = \{\mathsf{P}(\mathsf{a}), \neg\mathsf{P}(\mathsf{f}(\mathsf{a},\mathsf{b})), \mathsf{f}(x,\mathsf{b}) = x\}$	unsatisfiable
$S\bot = \{\mathsf{P}(\mathsf{a}), \neg\mathsf{P}(\mathsf{f}(\mathsf{a},\mathsf{b})), \mathsf{f}(\bot,\mathsf{b}) = \bot\}$	satisfiable
$\mathsf{a} \neq y \vee \neg\mathsf{P}(\mathsf{a}) \vee \mathsf{P}(y)$	$\mathsf{P}(\mathsf{a})$, congruence

Schemata

$x = x$	$s \neq s$	reflexivity
$x \neq y \vee y = x$	$s \neq t$	symmetry
$x \neq y \vee y \neq z \vee x = z$	$s \neq t$	transitivity
$x_1 \neq y_1 \vee x_2 \neq y_2 \vee \mathsf{f}(x_1, x_2) = \mathsf{f}(y_1, y_2)$	$\mathsf{f}(s_1, s_2) \neq \mathsf{f}(t_1, t_2)$	
$x \neq y \vee \neg\mathsf{P}(x) \vee \mathsf{P}(y)$	$\mathsf{P}(s)$	
$x \neq y \vee \neg\mathsf{P}(x) \vee \mathsf{P}(y)$	$\neg\mathsf{P}(s)$	congruence

Lemma

Symmetry and transitivity are consequences of reflexivity and congruence.

Symmetry.

$$\cfrac{\cfrac{}{x_1 = y_1 \wedge x_2 = y_2 \wedge x_1 = x_2 \rightarrow y_1 = y_2}\ \text{congruence}}{\cfrac{x = y \wedge x = x \wedge x = x \rightarrow y = x}{x = y \rightarrow y = x}\ \text{reflexivity}}\ x_1 \mapsto x, x_2 \mapsto x, y_1 \mapsto y, y_2 \mapsto x$$

□

Transitivity.

$$\cfrac{\cfrac{}{x_1 \neq y_1 \vee x_2 \neq y_2 \vee x_1 \neq x_2 \vee y_1 = y_2}\ \text{congruence}}{\cfrac{x \neq x \vee y \neq z \vee x \neq y \vee x = z}{x \neq y \vee y \neq z \vee x = z}\ \text{reflexivity}}\ x_1 \mapsto x, x_2 \mapsto y, y_1 \mapsto x, y_2 \mapsto z$$

□

The Bernays–Schönfinkel-Ramsey class of first-order formulae is a decidable fragment of first-order logic. Each formula in this class is equivalent to a satisfiable formula

$$\exists a_1 \ldots a_m \forall y_1 \ldots \; y_n \, F$$

where F is quantifier free and does not contain function symbols.

Example

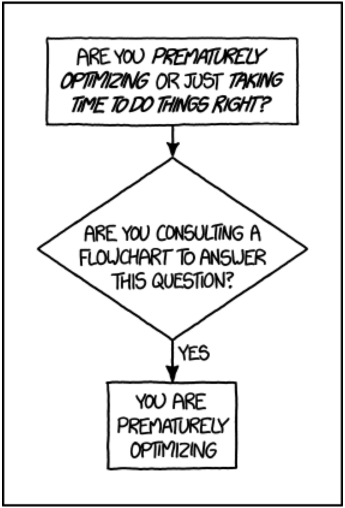
$$\mathsf{a} \neq x \vee \mathsf{Q}(\mathsf{a}, x), \mathsf{Q}(y, \mathsf{b})$$

Remark

In practice the additon of equality axioms reduces the success rates of (instantiation-based) automated theorem provers drastically.

- Done:
  - Scanning and parsing with `flex` and `bison`
  - Application logic with two simple strategies
  - SMT encoding to Yices 2
- Ongoing:
  - Integration of Unit Superposition calculus
  - Integration of Z3 for unsatisfiable core and maximal completion
  - Tests and experiments
- Missing:
  - Additional strategies
  - Combinations of strategies
  - Migration to Linux

Optimization



xkcd.com/1691/