

Term-Indexing

Alexander Maringele

January 27th, 2016

References

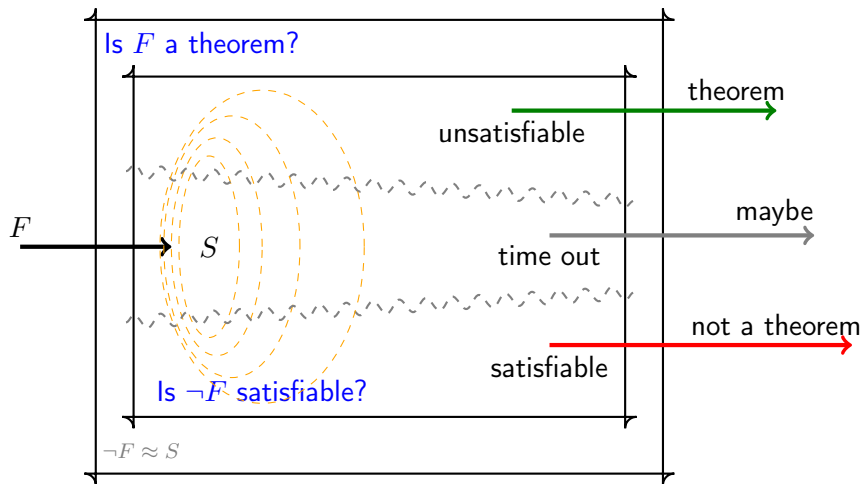


R. Sekar, I. V. Ramakrishnan, and Andrei Voronkov, *Term indexing*, Handbook of Automated Reasoning (Alan Robinson and Andrei Voronkov, eds.), Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 2001, pp. 1853–1964.

Outline

- 1 Motivation
- 2 Term Structure
- 3 Path Indexing
- 4 Discrimination Trees
- 5 Substitution Trees
- 6 Experiment

Refutation



Notation

Clausal form

$$\begin{aligned} & \{ P(f(x)) \vee f(x) \not\approx a, \ g(x, y) \approx a \vee \neg Q(x, y), \ C_3 \} \\ & \quad \equiv \\ & \quad \forall x (P(f(x)) \vee f(x) \not\approx a) \\ & \quad \quad \wedge \\ & \quad \forall xy (g(x, y) \approx a \vee \neg Q(x, y)) \\ & \quad \quad \wedge \\ & \quad \forall \text{Var}(C_3) (C_3) \end{aligned}$$

Goal

A sound and refutation complete calculus.

Resolution (without equality)

Resolve and factor all clauses and literals in an unsatisfiable set

$$\frac{A \vee \mathcal{C} \quad \neg B \vee \mathcal{D}}{(C \vee D)\sigma} (\sigma) \text{ resolution} \qquad \frac{A \vee B \vee \mathcal{C}}{(A \vee C)\sigma} (\sigma) \text{ factoring}$$

$$\sigma = \text{mgu}(A, B)$$

and the empty clause will be derived eventually.

Observation

Usually the set grows too fast to obtain a result.

Goal

A sound, refutation complete, and *effective* calculus.

- 1 *Reduce* search space
 - Ordered Resolution, Strategies, ...
 - ... with selection functions for clauses and literals
- 2 *Reduce* redundancy
 - e.g. discard clauses that are subsumed by other clauses
 - ... depending on the calculus

Example (forward subsumption)

$$S = \{^1P(x, y), ^2\neg P(a, z)\} \cup \{^3P(a, z')\}$$

t_1 subsumes t_3

$$\frac{P(x, y) \quad \neg P(a, z)}{\square} \quad \{x \mapsto a, y \mapsto z\}$$

Resolution

$$S \perp = \{P(\perp, \perp), \neg P(a, \perp), P(a, \perp)\}$$

InstGen / SMT

Goal

A sound, refutation complete, and effective calculus.

3 Quickly find

- *variants*
- *instances*
- *generalizations*
- *unifiable terms*

variant removal

backward subsumption

forward subsumption

resolution, demodulation

of a query term in a given set of terms.

Observation

Deduction rate drops quickly with sequential search.

Term Indexing

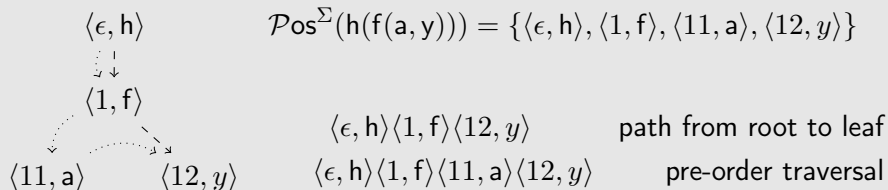
Data structures and algorithms for fast retrieval of matching terms.

Position Strings

Definition

$$\mathcal{Pos}^{\Sigma}(t) = \begin{cases} \{\langle \epsilon, x \rangle\} & \text{if } t = x \in \mathcal{V} \\ \{\langle \epsilon, f \rangle\} \cup \{\langle ip, s \rangle \mid \langle p, s \rangle \in \mathcal{Pos}^{\Sigma}(t_i)\} & \text{if } t = f(t_1, \dots, t_n) \end{cases}$$

Term traversals



Variables

Variants of terms generate the same position strings

- if variable names are ignored
- or normalized

$$f(y, z) \Rightarrow \langle \epsilon, f \rangle \langle 1, * \rangle \langle 2, * \rangle$$

$$f(y, z) \Rightarrow \langle \epsilon, f \rangle \langle 1, x_1 \rangle \langle 2, x_2 \rangle$$

$$f(y, y) \Rightarrow \langle \epsilon, f \rangle \langle 1, x_1 \rangle \langle 2, x_1 \rangle$$

In the first case even non-variants of terms generate the same strings.

Notation

We abbreviate

- path strings $\langle \epsilon, h \rangle \langle 1, f \rangle \langle 12, * \rangle$ h.1.f.2.*
- and pre-order traversal strings $\langle \epsilon, h \rangle \langle 1, f \rangle \langle 11, * \rangle \langle 12, * \rangle$ h.f.a.*
when the arities of function symbols are fixed.

Path Indexing

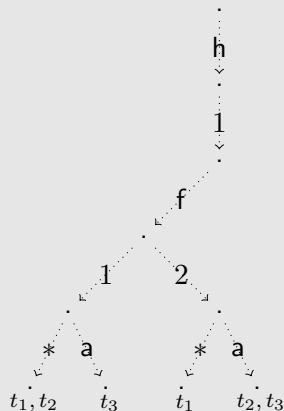
Build

$t_1: h(f(x, y)), t_2: h(f(x, a)), t_3: h(f(a, a))$

$t_1 \Rightarrow \{h.1.f.1.*, h.1.f.2.*\}$

$t_2 \Rightarrow \{h.1.f.1.*, h.1.f.2.a\}$

$t_3 \Rightarrow \{h.1.f.1.a, h.1.f.2.a\}$



Retrieve

$$^{t_1}h(f(x, y)), ^{t_2}h(f(x, a)), ^{t_3}h(f(a, a))$$

$$h(f(z, b)) \Rightarrow \{h.1.f.1.*, h.1.f.2.b\}$$

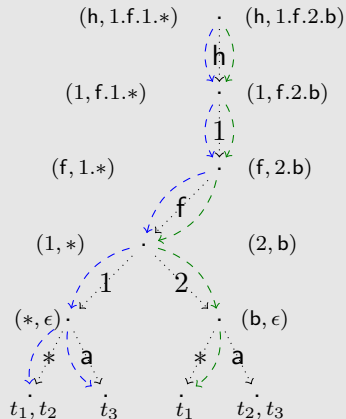
$$u : h(f(z, b)) \mapsto \{t_1, t_2, t_3\} \cap \{t_1\}$$

$$i : h(f(z, b)) \mapsto \{t_1, t_2, t_3\} \cap \{\}$$

$$g : h(f(z, b)) \mapsto \{t_1, t_2\} \cap \{t_1\}$$

$$v : h(f(z, b)) \mapsto \{t_1, t_2\} \cap \{\}$$

$$v : h(f(z, z)) \mapsto \{t_1, t_2\} \cap \{t_1\}$$



Unit Superposition Inference Rules

$$\frac{s \approx t \quad L[s']}{(L[t]) \cdot \sigma} \quad \begin{array}{l} \text{unit} \\ \text{paramodulation} \end{array}$$

where $\sigma = \text{mgu}(s, s')$, $s' \notin \mathcal{V}$, $t\sigma \neq s\sigma$

$$\frac{s \approx t \quad u[s'] \not\approx v}{(u[t] \not\approx v) \cdot \sigma} \quad \begin{array}{l} \text{unit} \\ \text{superposition} \end{array} \quad \frac{s \approx t \quad u[s'] \approx v}{(u[t] \approx v) \cdot \sigma}$$

where $\sigma = \text{mgu}(s, s')$, $s' \notin \mathcal{V}$, $t\sigma \neq s\sigma$, $v\sigma \neq u[s']\sigma$

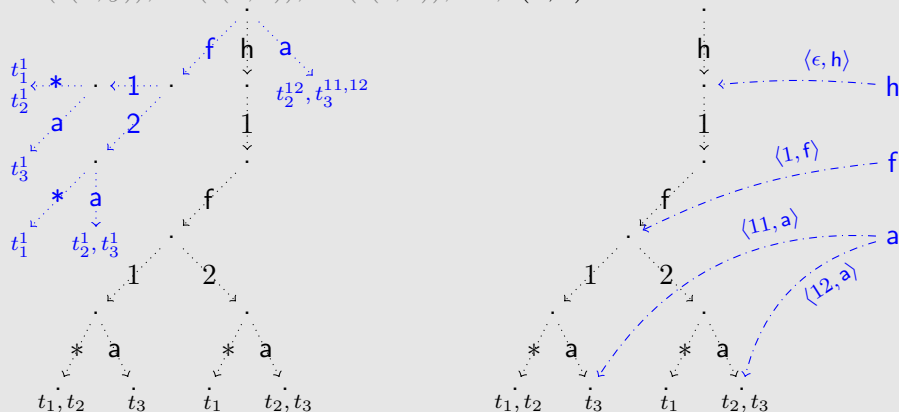
$$\frac{s \not\approx t}{\square} \quad \begin{array}{l} \text{unit equality} \\ \text{resolution} \end{array}$$

$$\frac{A \quad \neg B}{\square} \quad \begin{array}{l} \text{unit} \\ \text{resolution} \end{array}$$

where s and t (A and B respectively) are unifiable

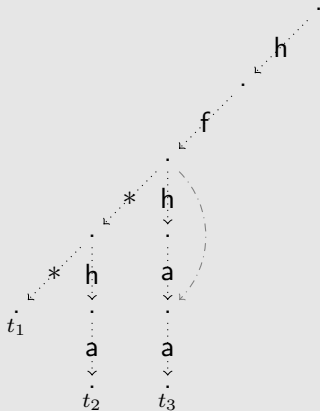
Demodulation

$$t_1: h(f(x, y)), t_2: h(f(x, a)), t_3: h(f(a, a)), \dots, f(x, a) \approx x$$



Discrimination Trees

Insert

$$^{t_1:}h(f(x, y)), ^{t_2:}h(f(x, h(a))), ^{t_3:}h(f(h(a), a))$$
$$t_1 \Rightarrow \text{h.f.*.*}$$
$$t_2 \Rightarrow \text{h.f.} * \text{h.a}$$
$$t_3 \Rightarrow \text{h.f.h.a.a}$$


Retrieve

$$^{t_1}h(f(x, y)), ^{t_2}h(f(x, h(a))), ^{t_3}h(f(h(a), a))$$

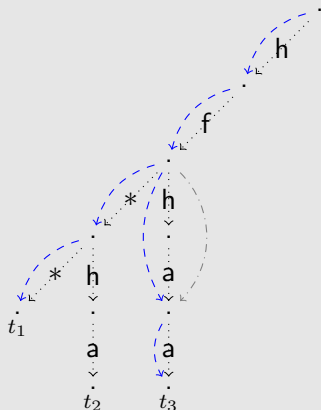
$$h(f(x', a)) \Rightarrow h.f.*.a$$

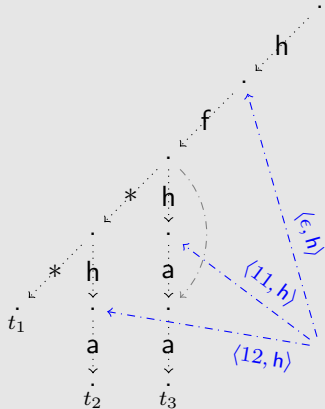
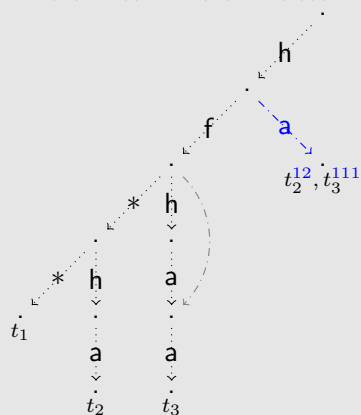
$$u : h(f(x', a)) \mapsto \{t_1, t_3\}$$

$$i : h(f(x', a)) \mapsto \{t_3\}$$

$$g : h(f(x', a)) \mapsto \{t_1\}$$

$$v : h(f(x', a)) \mapsto \{ \}$$



$$^{t_1}h(f(x, y)), ^{t_2}h(f(x, h(a))), ^{t_3}h(f(h(a), a))$$


Substitution Trees

Build

$$^{t_1}h(f(x, y)), ^{t_2}h(f(x, h(a))), ^{t_3}h(f(h(a), a)), ^{t_4}h(f(a, a)))$$

$$\downarrow$$

$$*0 \mapsto h(*1)$$

$$\downarrow$$

$$*1 \mapsto f(*2, *3)$$

$$*2 \mapsto x$$

$$*3 \mapsto a$$

$$*3 \mapsto y$$

$$t_1$$

$$*3 \mapsto h(a)$$

$$t_2$$

$$*2 \mapsto h(a)$$

$$t_3$$

$$*2 \mapsto a$$

$$t_4$$

TPTP/Problems/HWV/HWV134-1.p

2 332 428 formulae, 6 570 884 literals

checking afterwards	1000 new literals (ℓ_1, ℓ_2)	$A, \neg B$	sequential search	path index	speed up
1 000	500 000	761	726ms	70ms	10
2 000	1 500 000	812	2s	69ms	29
4 000	3 500 000	723	4s	75ms	53
8 000	7 500 000	433	9s	125ms	72
16 000	15 500 000	742	21s	221ms	95
32 000	31 500 000	592	40s	489ms	81
64 000	63 500 000	1 167	80s	697ms	114
128 000	127 500 000	1 479	160s	13s	12
256 000	255 500 000	1 097	320s	440s	<1
512 000	511 500 000	1 440	640s	348s	<2
1 024 000	1023 500 000	1 534	1280s	330s	<4