Reformation: A Domain-Independent Algorithm for Theory Repair

Alan Bundy Joint work with Boris Mitrovic

School of Informatics, University of Edinburgh

ESSENCE Lecture, Autumn School, 28th October 2014



・ロト ・ 日 ・ ・ 日 ・ ・ 日 ・

Outline









Repairing Faulty Theories

KnowItAll Ontology:

cap_of (Tokyo, Japan)

cap_of(Kvoto, Japan)

Proof of inconsistency:

		$cap_of(Tokyo, Japan), cap_of(x, z) \land cap_of(y, z) \implies x = y$
	cap_of(Kyoto, Japan),	$cap_of(y, Japan) \implies Tokyo = y$
$\textit{Tokyo} \neq \textit{Kyoto},$		Tokyo = Kyoto

Reformation repair:

Block unification of *cap_of(Kyoto, Japan*) and *cap_of(y, Japan*), e.g., change *cap_of(Kyoto, Japan*) to *was_cap_of(Kyoto, Japan*), or add time argument to *cap_of*, e.g., *present*, *past*.



The Need for Language Repair

The Reformation Algorithm

Discussion

Repairing Planning Failures

Plan failure in ORS: Mismatch of

Money(PA, £200) and Money(PA, £200, Credit_Card)

Reformation repair: Unblock failed unification.

Change planning agent's Money/2 to Money/3.

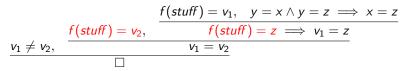


Repairing Physics Theories

Where's My Stuff Trigger:

$$O_1 \vdash f(stuff) = v_1$$
$$O_2 \vdash f(stuff) = v_2$$
$$O_{arith} \vdash v_1 \neq v_2$$

Proof of Inconsistency:



Reformation Repair:

Block unification of $f(stuff) = v_2$ and f(stuff) = z. e.g., rename two occurrences of stuff apart.



Example: Repairing a Faulty Proof of Cauchy's

Faulty Theorem: The limit of a convergent series of continuous functions is itself continuous [Cauchy]. **Counter-Example:** Square wave (discontinuous) is convergent sum of sine waves (continuous) [Fourier]. **Failed unification:**

$$y \ge y$$
 and $n \ge m(\epsilon/3, x + b(\delta(\epsilon/3, x, n)))$

due to an occurs check failure,

where m, δ and b are Skolem function.

Repair: Change 'convergent' to 'uniformly convergent'.

Convergent: $\forall \mathbf{x}, \forall \epsilon$

$$x > 0. \exists m. \forall n \geq m. |\sum_{i=m}^{n} f_i(x)| < \epsilon$$

Uniformly Convergent: $\forall \epsilon > 0. \exists m. \forall x. \forall n \ge m. |\sum_{i=m}^{n} f_i(x)| < \epsilon$

Note that $\forall x$ is moved to after $\exists m$.



э

A D > A D > A D > A D >

The Standard Unification Algorithm

Case	Before	Condition	After
Base	Τ;σ		Terminates
Trivial	$s \equiv s \wedge u; \sigma$		и; σ
Decomp	$f(\vec{s}^n) \equiv f(\vec{t}^n) \wedge u; \sigma$		$\bigwedge_{i=1}^{n} s_i \equiv t_i \wedge u; \sigma$
Clash	$f(\vec{s}^m) \equiv g(\vec{t}^n) \wedge u; \sigma$	$f \neq g \lor m \neq n$	fail
Orient	$t \equiv x \wedge u; \sigma$		$x \equiv t \wedge u; \sigma$
Occurs	$x \equiv s \wedge u; \sigma$	$x \in \mathcal{V}(s) \land x \neq s$	fail
Var Elim	$x \equiv s \wedge u; \sigma$	$x \not\in \mathcal{V}(s)$	$u\{x/s\}; \sigma \oplus \{x/s\}$

- Adapted from [Baader & Snyder, 2001][p455].
- Returns unique most-general unifier.



The Modified Unification Algorithm

Case	Before	Condition	After
Base	Τ;σ		Terminates
CCs	$f(\vec{s}^m) \equiv g(\vec{t}^n)$	$f = g \land n = m$	$\bigwedge_{i=1}^{n} s_i \equiv t_i \wedge u; \sigma$
CC _f	$\wedge u; \sigma$	$f \neq g \lor n \neq m$	Fail
VC _f	$x \equiv t \wedge u; \sigma$	$x \in \mathcal{V}(t)$	Fail
VC_s	or $t \equiv x \wedge u$; σ	$x \not\in \mathcal{V}(t)$	$u\{x/t\}; \sigma \oplus \{x/t\}$
$VV_{=}$	$x \equiv x \wedge u; \sigma$		и; σ
VV_{\neq}	$x \equiv y \wedge u; \sigma$	$x \neq y$	$u\{x/y\}; \sigma \oplus \{x/y\}$

- Equivalent to standard unification algorithm.
- Groups compound/compound and variable/compound cases into success/fail.



The Reformation Algorithm

Case	Before	Condition	Block	Unblock
Base	Т		Failure	Success
CCs		$f = g \land m = n$	Make $f(\vec{s}^m) \neq f(\vec{t}^m)$	
			$\bigvee_{i=1}^{n}$ Block $s_i \equiv t_i$	$\bigwedge_{i=1}^{n}$ Unblock $s_i \equiv t_i$
	$f(\vec{s}^m) \equiv g(\vec{t}^n)$		∨ Block <i>u</i>	∧ Unblock <i>u</i>
CCf	$\wedge u$	$f \neq g \lor m \neq n$	Success	Make $f(\vec{s}^m) = g(\vec{t}^n)$
				$\bigwedge_{i=1}^{n}$ Unblock $\nu(s_i) \equiv \nu(t_i)$
				$\wedge Unblock \nu(u)$
VC _f		$x \in \mathcal{V}(t)$	Success	Make $x \not\in \mathcal{V}(t)$
	$x \equiv t \wedge u$			\wedge Unblock $\nu(u\{x/t\})$
VCs	or $t \equiv x \wedge u$	$x \not\in \mathcal{V}(t)$	Make $x \in \mathcal{V}(t)$	
			\vee Block $u\{x/t\}$	Unblock $u\{x/t\}$

- Adapts modified unification algorithm.
- Flips success and failure cases to block/unblock unification.
- Blocking is a disjunction; unblocking a conjunction.
- Implemented and evaluated in SWI Prolog.



э

・ロト ・ 四ト ・ ヨト ・ ヨト

Example: Family Relations

Unprovable truth:

 $\frac{Mother(Camilla, William), \neg Mother(p, c, Step) \lor StepMother(p, c)}{\times}$

1 of 2 repairs: Add 3rd argument to *Mother*(*Camilla*, *William*). Successful resolution:

 $\frac{Mother(Camilla, William, Step), \neg Mother(p, c, Step) \lor StepMother(p, c)}{StepMother(Camilla, William)}$



・ロット (雪) () () () ()

The Many-Sorted Reformation Algorithm

Case	Before	Condition	Block	Unblock
Base	Т		Failure	Success
CCs	$f(\vec{s}^m): au_f$ \equiv	f = g $\land m = n$	Make $f(\vec{s}^m) \neq f(\vec{t}^m)$ $\bigvee_{i=1}^n \text{Block } s_i \equiv t_i$ $\vee \text{Block } u$	$\bigwedge_{i=1}^{n} \text{Unblock } s_i \equiv t_i$ $\land \text{ Unblock } u$
CC _f	g(t ⁿ):τ _g ∧ u	$f \neq g \\ \lor m \neq n$	Success	Make $f(\overline{s}^m) = g(\overline{t}^n)$ $\bigwedge_{i=1}^n$ Unblock $s_i \equiv t_i$ \land Unblock u
VCs	$x:\tau_x \equiv t\tau_t \wedge u$	$\begin{array}{c} x \not\in V(t) \\ \wedge \tau_t \preceq^* \tau_x \end{array}$	$\begin{array}{c} Make \ x \in V(t) \\ \lor \ \tau_t \not\preceq^* \tau_x \\ \lor \ Block \ u\{x:\tau_x/t:\tau_t\} \end{array}$	Unblock $u\{x:\tau_x/t:\tau_t\}$
VCf	$t\tau_t \equiv x\tau_x \wedge u$	$x \in V(t) \\ \lor \tau_t \not\preceq^* \tau_x$	Success	$\begin{array}{c} Make x \not\in V(t) \\ \land \tau_t \preceq^* \tau_x \land \\ Unblock \ u\{x\tau_x/t\tau_t\} \end{array}$
VVs	$x\tau_x \equiv$	$D = glbs(\tau_x, \tau_y)$ $\land D \neq \emptyset$	$\begin{array}{l} Make glbs(\tau_x, \tau_y) = \emptyset \lor \\ & \bigwedge_{\tau_d \in D} Block \\ u\{x:\tau_x/y:\tau_d, y:\tau_y/y:\tau_d\} \end{array}$	$\bigvee_{\tau_d \in D} \text{Unblock} \\ u\{x:\tau_x/y:\tau_d, y:\tau_y/y:\tau_d\}$
VV _f	$y:\tau_y \wedge u$	$glbs(au_x, au_y) = \emptyset$	Success	Make $glbs(\tau_x, \tau_y) \neq \emptyset \land$ Unblock $u\{x:\tau_d/y:\tau_d\}$

- Extended reformation to many-sorted logics.
 - Repairs now include splitting and merging of sorts.
 - Plus reorganisation of sort hierarchy.

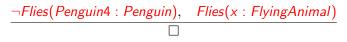


э

A D F A B F A B F A B F

Example: Flying Penguins

Contradiction:



where Penguin \prec Bird and Bird \prec FlyingAnimal.

1 of 3 repairs: Split Bird into FlyingBird and Bird.

- Replace *Bird* ≺ *FlyingAnimal* with *FlyingBird* ≺ *FlyingAnimal*.
- Add *FlyingBird* ≺ *Bird*.



イロト 不得 トイヨト イヨト

Search Space Control

- Huge search space: many possible repairs for every unwanted unification.
 - Each proof step requires unification.
 - Each unification step suggests multiple repairs.
- Need heuristics to prune and prioritise.
 - Protect some functions/predicates.
 - Keep repairs minimal.
 - Maximise blocked inconsistencies; minimise blocked truths.



A D F A B F A B F A B F

Conclusion

- Language repair essential in many applications.
- Reformation is general-purpose algorithm.
- Huge search space requires heuristic control.
- Need to define *minimality*.
- Explore extensions to other logics, e.g., DL.



Baader, F. and Snyder, W.

(2001).

Unification theory.

In Robinson, J. A. and Voronkov, A., (eds.), *Handbook of Automated Reasoning, Volume 1*, volume 1, chapter 8, pages 447–553. Elsevier.

