The Interaction of Representation and Reasoning

Alan Bundy
Joint work with Boris Mitrovic

School of Informatics,
University of Edinburgh

Belgrade 16th May 2013

Outline

Repairing Faulty Theories

KnowItAll Ontology:

$$cap\_of(Tokyo, Japan) \qquad cap\_of(Kyoto, Japan)$$

Proof of inconsistency:

$$\dfrac{Tokyo \neq Kyoto, \quad \dfrac{cap\_of(Kyoto, Japan), \quad \dfrac{cap\_of(Tokyo, Japan), \quad cap\_of(x, z) \wedge cap\_of(y, z) \implies x = y}{cap\_of(y, Japan) \implies Tokyo = y}}{Tokyo = Kyoto}}{\square}$$

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., $present$, $past$.

Repairing Planning Failures

Plan failure in ORS: Mismatch of

$$\mathit{pay}(\mathit{pa}, \$200, \mathit{spa}) \quad \text{and} \quad \mathit{pay}(\mathit{pa}, \$200, \mathit{spa}, \mathit{credit\_card})$$

Reformation repair: Unblock failed unification.

Change planning agent's $\mathit{pay}/3$ to $\mathit{pay}/4$.

Repairing Physics Theories

Where's My Stuff Trigger:

$$\begin{array}{rcl} O_1 & \vdash & f(stuff) = v_1 \\ O_2 & \vdash & f(stuff) = v_2 \\ O_{arith} & \vdash & v_1 \neq v_2 \end{array}$$

Proof of Inconsistency:

$$\dfrac{v_1 \neq v_2, \quad \dfrac{f(stuff) = v_2, \quad \dfrac{f(stuff) = v_1, \quad y = x \wedge y = z \implies x = z}{f(stuff) = z \implies v_1 = z}}{v_1 = v_2}}{\square}$$

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:

$$y \geq y \text{ and } n \geq m(\epsilon/3, x + b(\delta(\epsilon/3, x, n)))$$

due to an occurs check failure,
where m, δ and b are Skolem function.

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

Convergent:

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

Uniformly Convergent:

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

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

The Standard Unification Algorithm

Case	**Before**	**Condition**	**After**
Trivial	$s \equiv s \wedge E; \sigma$		$E; \sigma$
Decomp	$f(s_1, \ldots, s_n) \equiv f(t_1, \ldots, t_n) \wedge E; \sigma$		$s_1 \equiv t_1 \wedge \ldots \wedge s_n \equiv t_n \wedge E; \sigma$
Clash	$f(s_1, \ldots, s_m) \equiv g(t_1, \ldots, t_n) \wedge E; \sigma$	$f \neq g \vee m \neq n$	fail
Orient	$t \equiv x \wedge E; \sigma$		$x \equiv t \wedge E; \sigma$
Occurs	$x \equiv s \wedge E; \sigma$	$x \in V(s) \wedge x \neq s$	fail
Var Elim	$x \equiv s \wedge E; \sigma$	$x \notin V(s)$	$E\{x/s\}; \sigma \oplus \{x/s\}$

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

The Modified Unification Algorithm

Case	Before	Condition	After
CC_s	$f(s_1, \ldots, s_m) \equiv$	$f = g \wedge n = m$	$\bigwedge_{i=1}^{n} s_i \equiv t_i \wedge E; \sigma$
CC_f	$g(t_1, \ldots, t_n) \wedge E; \sigma$	$f \neq g \vee n \neq m$	fail
VC_f	$x \equiv t \wedge E; \sigma$	$x \in \mathcal{V}(t)$	fail
VC_s	or $t \equiv x \wedge E; \sigma$	$x \notin \mathcal{V}(t)$	$E\{x/t\}; \sigma \oplus \{x/t\}$
$VV_=$	$x \equiv x \wedge E; \sigma$		$E; \sigma$
$VV_{\neq}$	$x \equiv y \wedge E; \sigma$	$x \neq y$	$E\{x/y\}; \sigma \oplus \{x/y\}$

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

The Reformation Algorithm

Case	Input Problem	Condition	Block	Unblock
CC_s	$f(s_1,\ldots,s_m) \equiv g(t_1,\ldots,t_n) \wedge E$	$f = g \wedge m = n$	Make $f \neq g \vee m \neq n$ $\bigvee_{i=1}^{n}$ Block $s_i \equiv t_i$ $\vee$ Block E	$\bigwedge_{i=1}^{n}$ Unblock $s_i \equiv t_i$ $\wedge$ Unblock E
CC_f		$f \neq g \vee m \neq n$	No action	Make $f = g \wedge m = n$ $\bigwedge_{i=1}^{n}$ Unblock $\nu(s_i) \equiv \nu(t_i)$ $\wedge$ Unblock $\nu(E)$
VC_f	$x \equiv t \wedge E$ or $t \equiv x \wedge E$	$x \in \mathcal{V}(t)$	No action	Make $x \notin \mathcal{V}(t)$ $\wedge$ Unblock $\nu(E\{x/t\})$
VC_s		$x \notin \mathcal{V}(t)$	Make $x \in \mathcal{V}(t)$ $\vee$ Block $E\{x/t\}$	Unblock $E\{x/t\}$

- 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.

Propagating Repairs

- Propagate changes to ancestor axioms.

$$\dfrac{\dfrac{Q, \quad \neg Q' \vee L_1}{L_1[e_1]\sigma_q,} \qquad \dfrac{R, \quad \neg R' \vee \neg L_2[e_2]}{\neg L_2[e_2]\sigma_r}}{\square}$$

$$\dfrac{\dfrac{Q, \quad \neg Q' \vee L_1}{L_1[e_1]\sigma_q,} \qquad \dfrac{R, \quad \neg R' \vee \neg L_2[\nu(e_2)]\sigma_r\overline{\sigma_r}}{\neg L_2[\nu(e_2)]\sigma_r}}{\text{Blocked}}$$

- Can we insist one parent is an axiom?
- Refinement requires decisions.
  - Which instances of functions/predicates are renamed, e.g., *cap_of* or *was_cap_of*?
  - What values are given to new arguments, e.g., $cap\_of(London, UK, \ldots)$?

Boris Mitrovic's UG4 Project

- Extended reformation of single-sorted and multi-sorted logics.
    - Repairs now include splitting and merging of sorts.
    - Plus reorganisation of sort hierarchy.
- Exhaustively generate theorems in theory:
    - Repair any inconsistencies found.
- Identified heuristics to prune search space,
    - e.g., some functions/predicates protected, such as $=$, $+$, etc.

Search Space Control

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

Conclusion

- Language repair essential in many applications.
- Reformation is general-purpose algorithm.
- Repairs must be propagated from chosen reformation suggestion to all axioms.
- Huge search space requires heuristic control.
- 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 I, chapter 8, pages 447–553. Elsevier.