Refutation of abductive repair in ontology engineering

© Copyright 2019 by Colin James III  All rights reserved.

Abstract: We evaluate the stated example of complete-debug problem (CDP) in formulas framing the definitions, oracles, and repairs. None is tautologous. This forms a non tautologous fragment of the universal logic $VŁ4$.

We assume the method and apparatus of Meth8/$VŁ4$ with Tautology as the designated proof value, $F$ as contradiction, $N$ as truthity (non-contingency), and $C$ as falsity (contingency). The 16-valued truth table is row-major and horizontal, or repeating fragments of 128-tables, sometimes with table counts, for more variables. (See ersatz-systems.com.)

LET $\neg$ Not, $\neg$; $+$ Or, $\lor$, $\cup$; $-$ Not Or; $\&$ And, $\land$, $\cap$, $\cdot$, $\otimes$; $\setminus$ Not And;

> Imply, greater than, $\rightarrow$, $\Rightarrow$, $\mapsto$, $\succ$; $<$ Not Imply, less than, $\in$, $\subset$, $\not\in$, $\not\subset$;

$\equiv$ Equivalent, $\equiv$, $\equiv$, $\equiv$, $\equiv$; $\not\equiv$ Not Equivalent, $\not\equiv$, $\not\equiv$;

$\%$ possibility, for one or some, $\exists$, $\forall$, $\mathcal{M}$; $\#$ necessity, for every or all, $\forall$, $\Box$, $L$;

$(z=z)$ $T$ as tautology, $T$, ordinal 3; $(z@z)$ $F$ as contradiction, $\emptyset$, Null, $\perp$;

$(\%z>\#z)$ $N$ as non-contingency, $\Delta$, ordinal 1; $(\%z<\#z)$ $C$ as contingency, $\nabla$, ordinal 2;

$\sim(x < y)$ $(x \leq y)$, $(x \subseteq y)$; $(A=B)$ $(A\sim B)$.

Note for clarity, we usually distribute quantifiers onto each designated variable.


Abstract As semantically-enabled applications require high-quality ontologies, developing and maintaining as correct and complete as possible ontologies is an important, although difficult task in ontology engineering. A key step is ontology debugging and completion. In general, there are two steps: detecting defects and repairing defects. In this paper we formalize the repairing step as an abduction problem and situate the state of the art with respect to this framework. ...

2.1 Formalization

2.1.1 Repair As an example, consider the CDP [complete-debug problem] in Fig. 1. ... Then R1,R2, R3, R4 and R5 are all repairs of the CDP

Figure 1: Example complete-debug problem

\[
T: \{ax1: p1 \subseteq p2, ax2: p1 \subseteq p3, ax3: p1 \subseteq \neg p4, ax4: p2 \subseteq p4, ax5: p2 \subseteq p5, ax6: p3 \subseteq p5, ax7: p3 \subseteq p6, ax8: p4 \subseteq p7, ax9: p5 \subseteq \forall s.p8, ax10: p6 \subseteq \exists s.\neg p8\} \\

\ldots
\]

\[
\text{Or}(X) = \text{true for } X = ax2, ax3, ax4, ax5, ax7, ax8, ax9, p7 \subseteq p3; \\
\text{Or}(X) = \text{false for } X = ax1, ax6, ax10, p7 \subseteq p5, p3 \subseteq p8
\]

Remark 2.1.1.0: We test $(T>((\text{true for } X)\&(\text{false for } X)))$ as Eqs. 2.1.1.1 > (2.1.1.3.1)&(2.1.1.4.1)).
(((~(q<p)&~(r<p))&~(s<p)&~(s<q)))&((~(t<q)&~(t<r))&~((#x&w)<t)&~((%x&w)<u)))>
(((((~(r<p)&~(~s<p))&~(s<q)&~(t<q)))&((~(u<r)&~(v<s))&~((#x&w)<t)&
~((r<v)))))=(z=z))&
(((~(q<p)&~(t<r))&~((%x&w)<u))&~(t<v)&~(w<r)))=(z@z)) ;


~(t<s)&~(r<v) ;

Remark 2.1.3.1.0: We test the CDP to imply the repairs:

Eqs. 2.1.1.5.1 implies 2.1.2.1.1.

(((((~(q<p)&~(r<p))&~(s<p)&~(s<q)))&((~(t<q)&~(t<r))&~((u<r)&~(v<s))))&
~((#x&w)<t)&~((%x&w)<u)))>(((((~(r<p)&~(~s<p))&~(s<q)&~(t<q)))&
((~(u<r)&~(v<s))&~((#x&w)<t)&~(r<v))))=(z=z))&(((~(q<p)&~(t<r))&
~((%x&w)<u))&~(t<v)&~((w<r)))=(z@z)) ;

Remark 2.1.2.1.0: We map the unique relations from repairs of R1, R2, R3, R4, R5 of
Eqs. 2.1.1.5.2 and 2.1.3.1.2 as rendered are *not* tautologous. This means the example given as 2.1.1.1 is *not* tautologous, the oracles in 2.1.1.3.1 and 2.1.1.4.1 are *not* truthful, and the repairs in 2.1.2.1.1 are incorrect. This refutes the conjecture of ontology engineering.