Towards a Theory of Diagnosis of Faulty Ontologies

Towards a Theory of Diagnosis of Faulty Ontologies *

Alan Bundy
School of Informatics, University of Edinburgh
Edinburgh, Scotland, UK
A.Bundy@ed.ac.uk

Abstract

We initiate research into a generic theory of diagnosis of faulty ontologies. The proposals are based on, but generalise, our experience with the GALILEO and ORS systems. We make some initial simplifying assumptions, which we hope will not restrict the application of the proposals to new areas. In particular, we look at repairing faulty ontologies where the fault is revealed by an inference failure and the repair is implemented by a signature and/or theory morphism. More concretely, we focus on situations where a false conjecture has been proved. Diagnosis consists of constructing a morphism by analysis of failed inference. It is assumed that an oracle is available that can answer (and sometimes ask) questions, but whose own ontology is otherwise inscrutable.

1 Introduction

This paper builds on our experience of two systems for ontology evolution: ORS and GALILEO. Both systems start with a potentially faulty logical theory representing some knowledge. We will use the word *ontology* to describe such logical theories. As a result of inference failure, these ontologies are shown to be faulty. The fault is then automatically diagnosed and the faulty ontology is then repaired. We will try to generalise from the experience of these two systems to construct a generic theory of diagnosis of faulty ontologies.

ORS (Ontology Repair System) evolves planning ontologies [McNeill and Bundy, 2007]. In its domain a planning agent (PA) constructs plans from the services provided by service providing agents (SPAs). The PA represents these services with STRIPS-like operators. If the plan fails on execution then this indicates that the PA's representation of the SPA's services is faulty. ORS diagnoses the fault and repairs the PA's ontology. Repairs can be either to the theory or to the language. An example of a theory repair is inserting a missing precondition in a STRIPS operator e.g., an SPA issuing travel visas might require the PA to possess some additional documentation that it wasn't initially aware that it needed. An example of a language repair is adding an additional argument to a predicate, e.g., refining a proposition asserting that the PA must provide a photograph by specifying its size with the additional argument. The PA then replans with the repaired ontology. Several repairs may be required to form a plan that will execute successfully.

GALILEO (Guided Analysis of Logical Inconsistencies Leads to Evolved Ontologies) evolves ontologies representing physics theories [Bundy and Chan, 2008]. In its domain the predicted value of a physical function may be inferred to differ from its observed value. GALILEO represents the world with multiple ontologies, e.g., one for the theory and another for an experiment. These ontologies are locally consistent, but can be globally inconsistent. Different patterns of divergence and repair are represented in *ontology repair plans* (ORP)s. If an ORP's trigger pattern is matched then an appropriate repair is executed. Repairs include: splitting one concept into many, merging several concepts into one, adding a dependence on a hidden variable, or making a concept independent of a variable. For instance, an anomaly in the orbital velocity of spiral galaxies might suggest splitting the concept of matter into regular visible matter, dark matter and total matter.

An *ontology* is a pair $\langle \Sigma, A \rangle$, where:

- Σ is the ontology's *signature*, which defines its language. It consists of a set of type declarations for the concepts in the ontology.
- A is the ontology's *axioms*, which define its theory. It consists of a set of formulae asserted to be true.

The type declarations and formulae are expressed in a logic $\mathcal{L}$. We will represent the repair of a source ontology O as the target ontology $\nu(O)$, defined as:

$$\nu(O) \quad ::= \quad \langle \nu_\sigma(\Sigma), \nu_\alpha(A) \rangle$$

where ν_σ is a signature morphism and ν_α is a theory morphism. These morphisms are meta-level functions that describe how to repair the source signature/axioms to form the target ones.

The logic of ORS is KIF, an extended, typed, classical first-order logic. Its signature consists of type declarations for its

*Thanks to Liwei Deng and an anonymous referee for feedback on an earlier draft and to my research group for feedback during a seminar.

predicates and constants. Its axioms are (a) the STRIPS operators and (b) propositions describing its beliefs about its environment.

The logic of GALILEO is simply-typed lambda calculus, a form of higher-order logic. Its ontology's signature's consists of type declarations of functions describing the physical world and its attributes. Note that higher-order functions are required to represent force fields, calculus operations, astronomical orbits, etc. Its ontology's axioms include mathematical theorems, physical laws, experimental observations, etc.

2 Types of Ontological Fault

ORS and GALILEO have the following common operations:

- identification of a problem with an ontology;
- diagnosis of this problem;
- repair of the problem.

Although each system incorporates mechanisms for diagnosis and repair, these are domain specific. We lack a theory to underpin them. In particular, we lack *a theory of diagnosis*. In this paper we will initiate the development of such a diagnostic theory.

To aid formalisation, we will make the following simplifying assumptions.

1. *An attempt to prove a conjecture can one of the following outcomes:*
   - The conjecture may be *proved*;
   - The search space may be exhausted without a proof being found, i.e., the conjecture is *unprovable*; or
   - Resource limits may be encountered before a proof is found or the search space is exhausted, i.e., the conjecture is *undetermined*.
2. *By a* fault *in an ontology we will mean some kind of reasoning failure.* Examples of such failures include:
   (a) A conjecture is true but unprovable.
   (b) A conjecture is false but *is* proved.
   (c) Inference is too inefficient, e.g., the search space is infeasibly large. The evidence may be that too many conjectures are left undetermined.
3. *By an ontology* repair *we will mean the application of a morphism (signature, theory or some combination) to the source ontology to produce a new target one.* The fault being repaired should not hold in the target ontology.
4. *By a fault* diagnosis *we will mean constructing a morphism that can be used to repair the fault.*
5. *The* diagnostic process *is a procedure that, given a faulty source ontology, can produce a diagnosis. For both type 2a and type 2b faults the diagnosis process can be represented as the analysis of a failed or successful proof attempt.*
6. *Proof attempts can be represented as a search space containing a partial or complete proof. Without loss of generality, we can represent the search space as an* OR*-tree and a proof as an* AND*-tree.*

3 Justification of these Simplifying Assumptions

In both GALILEO and ORS, ontology evolution is driven by inference failure, in particular, type 2b inference failure. The derivation of false theorems can arise in two ways: (i) because the ontology is inconsistent, so all formulae are provable; (ii) because a particular theorem is false in the preferred model. Below we will focus on type 2b (ii) inference failure.

- In ORS, a plan is derived that then fails on execution. The derivation of the plan is also a formal verification that the plan will achieve the goals of the PA. The failure of the plan shows that the plan will *not* achieve these goals, so the verification has proved a false conjecture. Note that this is a type 2b (ii) inference failure, where the preferred model is the real world, and truth and falsity in this world are revealed by the success or failure of plan execution.
- In GALILEO, a contradiction is derived from the combination of two ontologies: a theoretical one and an experimental one. So, by merging these two ontologies into a single inconsistent one, we could regard GALILEO's type 2b inference failure as of the (i) kind. However, it will be more convenient to regard the experimental ontology as describing experiments performed in the real world. Under this interpretation, GALILEO is similar to ORS, in that the preferred model is the real world and truth and falsity in that world is revealed by experimental results that are described in the experimental ontology merely for implementational convenience.

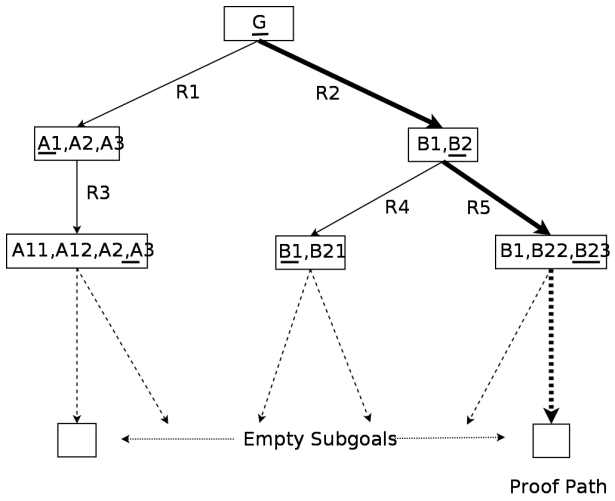
We will, therefore, concentrate on the diagnosis of type 2b (ii) faults. We leave type 2a faults for future work, but expect the diagnostic process for these faults to be essentially dual to the type 2b ones. Type 2c faults are inherently rather different and we expect the diagnostic process also to be very different. Early work on this problem includes McCarthy's mutilated checkers board problem [McCarthy, 1964] and Amarel's work on the evolution of representations of the missionaries and cannibals problem [Amarel, 1968].

In previous work it has been shown how various kinds of ontology repair operations can be represented as signature and/or theory morphisms, adapting ideas from Category Theory[1] For the purposes of the current paper, it is enough to envisage these morphisms as meta-level functions applied to the two components of ontologies. Typically, both theory and signature morphisms will be required in the overall repair of an ontology. Thus, *diagnosis* can be regard as finding these morphisms and *repair* as applying them. Since repair consists only of morphism application, the more interesting problem, and the focus of this paper, is diagnosis.

It will be convenient to represent a search space as an OR-tree (see Figure 1). Each node of the tree will consist of a set of sub-goals, where the root is the singleton set of the initial conjecture. The children of a node are the result of applying a rule of inference to one of the sub-goals and replacing it with the resulting sub-sub-goals. OR-branching occurs when there are multiple possible rule applications to a node. The

[1] Alan Smaill, personal communication (BBN) 1682.

sub-goals in a leaf node are unproven conjectures. A proof will be a path of the tree from the root to a leaf, in which the leaf subset is empty. illustrates these two kinds of tree.



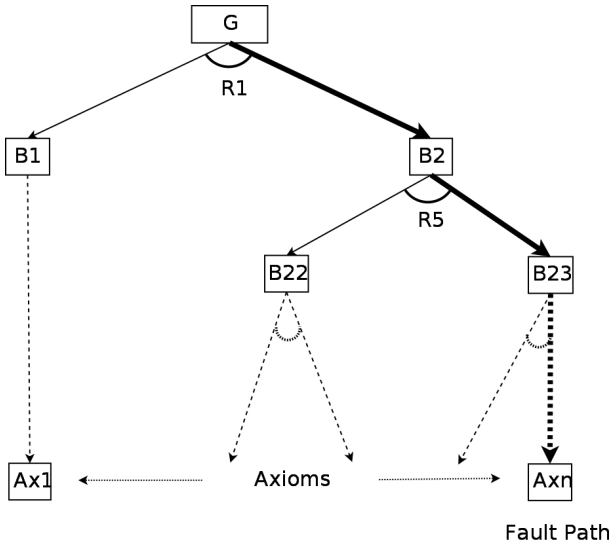
Each rectangle represents a set of AND *sub-goals, each of which must be proved. The arrows represent rules applied to the underlined sub-goal. Each path is a potential proof attempt, which completes when its leaf contains no sub-goals. The bold face path is a completed proof.*

Figure 1: A Search Space as an OR Tree

If we want to consider a proof alone, it will be convenient to consider it as an AND tree, in which each node consists of only one sub-goal (see Figure 2). The children of a node will be the sub-sub-goals arising from an application of a rule of inference to its sub-goal. The leaves will be instances of axioms.

An earlier attempt at diagnosis via proof analysis is Shapiro's work on logic programming debugging [Shapiro, 1983]. Suppose a Prolog program has unexpectedly failed. This is akin to a type 2a fault. Shapiro shows how to step through the failed proof. For each sub-goal, an oracle (the user) is asked whether the sub-goal is true or false. At each node of the OR-tree search space there will be at least one true sub-goal whose proof has failed, otherwise, the search space will contain a proof. A path of true/failed subgoals are traced through the search space until a leaf is reached. At this point the true/failed leaf sub-goal represents a missing axiom. Adding this sub-goal as a new axiom will debug the logic program.

A Shapiro-like technique is used in ORS but for type 2b faults, i.e., it is used to identify false 'facts' in the ontology, where the oracle is derivation in the ontology. This suggests that this kind of analysis can be adapted for faults of both type 2a and type 2b.



Each rectangle contains only one sub-goal. Each rule labels a family of arrows: one for each sub-goals arising from the rule's application. A proof is complete when all the leaves of the tree are instances of axioms. The bold face path represents a fault path, i.e., one in which all the sub-goals are false, including the final axiom instance.

Figure 2: A Proof as an AND Tree

4 How to Block Illegitimate Success

By 'illegitimate success' we mean that something false can be inferred in the ontology, i.e., the ontology has a type 2b (ii) fault. We now discuss the use of Shapiro-like, proof analysis to partially diagnose the fault by locating the source of the problem. Note that, in the case of type 2b faults, we don't have to explore the whole search space, as we have a proof of something false, so can restrict ourselves just to this proof.

Consider the proof as an AND-tree. The root of the tree is the false 'theorem'. The children of each node are the sub-goals that collectively imply it. We will assume that the derivation of a sub-goal by its children is via a logical rule of inference that is beyond reproach, i.e., the fault lies in the axioms of the ontology and not its underlying logic[2]. The leaves of the tree are instances of axioms.

Note that free variables in the sub-goals will arise from dual skolemisation of existentially quantified variables, and their instantiation during the proof will correspond to witnesses being discovered for these variables. Usually, all such free variables will be instantiated at some point during the proof. If any free variables are left uninstantiated then this indicates that the theorem has been proved for all their possible instantiations. Any instantiations of free variables made during the proof can be inherited back up the tree, so proof trees will usually be totally ground.

[2]If required, this assumption could be relaxed by treating the rule of inference as an additional child node.

For each false node, one of its children is also false, otherwise the node would be true. Therefore, there exists at least one path from root to leaf in which all the nodes are false. We call such a path a *fault path*.

We assume that an oracle exists that can identify false nodes, provided they are in its ontology's language. In §5 we discuss where these oracles might come from. For each false node, the oracle is asked for the truth of each of the child nodes. By recursing this process, all the fault paths can be identified. The leaves of these fault paths are all false instances of axioms. Therefore, the axioms are also false and must be repaired.

Suppose that, during this process, a sub-goal is found that is not in the oracle's language. The source ontology's signature should be repaired so that the sub-goal *is* in the oracle's ontology. If this newly repaired sub-goal is false, then the search for fault paths should resume at this sub-goal.

5 Where do the Oracles come from?

The proposals in §4 depend crucially on the existence of an oracle to identify proven, but false, subgoals. Is it realistic to assume the existence of such oracles? The answer depends on the application. We illustrate this with the two applications of ORS and GALILEO.

5.1 ORS Oracle

In ORS the oracle is provided by the service-providing agents. The assumptions about these SPAs in ORS are:

- An SPA does not have the functionality, nor would its owner usually consider it desirable, to reveal its ontology.
- An SPA is regarded as the ultimate authority of the conditions under which it is prepared to perform a service. As such, its ontology is not subject to change[3].
- An SPA will, however, answer specific questions. These are posed as KIF formulae, which the SPA will try to prove and to which it will return an answer.
- These answers can take the form of $true$, if the formula can be proven without instantiation, $false$, if the formula cannot be proven, or a substitution, indicating the instantiations necessary to prove the formula.
- An SPA may also ask questions of the planning agent (PA) during the plan formation process. These follow the same process as when the PA asks a question of the SPA.

Note that ORS oracles are able to classify non-ground subgoals and can sometimes prove them without any instantiation, i.e., for all possible values of the free variables. When instantiation is necessary to prove the subgoals, then this may prove useful in the next phase of axiom modification (see §6).

[3]However, in a current UG4 project by Agnieszka Bomersbach, we are relaxing this assumption.

5.2 GALILEO Oracle

In GALILEO the real world can be regarded as the oracle. This enforces rather different assumptions about the oracle than in ORS.

- The real world does not have an inherent ontology, but only one imposed by those trying to understand it. Consequently, its ontology is also subject to evolution as such understanding deepens.
- We can use it only to make measurements and observations of particular phenomena. These can be modelled either as ground atoms, which will be classified as $true$ or $false$, or ground function calls, for which the output will be returned. For instance, a ground predicate might be $At(Ball_1, Posn_1, Time_1)$ and a ground function call might be $Vel(Ball_1, Time_1)$, to which the answer might be $10\ metres/sec.$
- The real world cannot itself ask questions.

6 Modifying Axioms

If an instance of a faulty axiom is found, then we need to delete or modify this axiom. The simplest thing is to delete it, but that may not be optimal. In particular, the source axiom might be successfully used in proofs of true formulae. An unwanted side effect of deleting it would be that these true formulae would cease to be theorems — at least, by any proofs using the source axiom. The alternative is a modification that makes it inapplicable in the faulty proof, but retains its applicability elsewhere. In ORS, for instance, a precondition was often added to the STRIPS operators that differentiated the good uses from the bad ones.

The problem can be seen as a classification task, of the kind to which the HR system is suited [Colton *et al.*, 1999]. We form two sets of applications of the source axiom: Good and Bad. Good applications are those where it is used to prove true formulae and Bad ones where it is used to prove false formulae. In particular, its application in the faulty proof is in the Bad set. We now learn a classification that differentiates these two sets. It might, for instance, be an extra condition on the axiom or it might be an instantiation of the axiom to restrict it to Good applications. We can test our modified axiom by asking the oracle whether it is true. If not, then further or different modifications are needed.

7 Modifying Signatures

If fault is detected in the source ontology's signature, then we need to figure out what signature morphism, ν_σ, to construct. ν_σ must map the false sub-goal to a target sub-goal that *is* in the oracle's signature. There are many ways to do this: the target sub-goal can be any formulae in the oracle's signature. Ideally, however, we would like the source and target sub-goals to be similar, e.g., by minimising the edit difference between them. Another approach, that has been explored by Theodosia Togia in a recent MSc project, is the use of computational linguistics tools, such as Wordnet, to identify synonymous and similar connections between ontology signatures [Togia *et al.*, 2010]. In practice, ν_σ will typically be of the type listed in §3 or its dual, namely: naming apart/merging

of functions, permuting arguments and adding/removing arguments.

As in ORS we will make the following assumptions:

- We have no direct access to the oracle's ontology, i.e., we can't just browse its ontology for likely targets.
- We can, however, ask the oracle questions. So, we can construct a candidate target, send it to the oracle and expect an answer of one of the three forms: (i) in ontology's language and true, (ii) in ontology's language and false, (iii) not in ontology's language . If it is in the ontology's language then we can proceed, and if it is also false then we can use it to resume the search for a fault path.

As in ORS, we may already have some prior communication with the oracle, e.g., when discovering that the original 'theorem' is false. ORS has the notion of a *surprising question*, i.e., a question asked by the oracle that was not expected. Suppose that this question was not in the original ontology. Then it (or some instance of it) might be a candidate for the target. For instance, the PA might expect to be asked $Pay(PA, £100)$, since this instantiates a preconditions of one of its STRIPS operator. The SPA might, however, ask it $Pay(PA, £100, CreditCard)$ instead. This surprising question is a clue that it should modify the type of its Pay predicate from binary to ternary, where the extra argument defines the method of payment.

In the worst case, arbitrary modifications could be applied to the source, e.g., permuting and adding arguments, replacing names with synonyms, and then sent to the oracle for testing until one is in the oracle's signature. These might be explored breadth-first, e.g., first trying all one-step modifications, then all two-step, etc. This effectively uses minimisation of edit distance as a heuristic.

8 Conclusion

We have initiated a generic theory of diagnosis of faulty ontologies. The proposals here are based on, but generalise, our experience with the GALILEO and ORS systems. We have made some initial simplifying assumptions, which we hope will not restrict the application of the proposals to new areas. In particular, we are looking at repairing faulty ontologies where the fault is revealed by an inference failure and the repair is implemented by a signature and/or theory morphism. More concretely, in this note, we focus on situations where a false conjecture has been proved. Diagnosis consists of constructing a morphism by analysis of failed inference. It is assumed that an oracle is available that can answer (and sometimes ask) questions, but whose own ontology is otherwise inscrutable.

References

[Amarel, 1968] S. Amarel. On representations of problems of reasoning about actions. In D. Michie, editor, *Machine Intelligence 3*, pages 131–171. Edinburgh University Press, 1968.

[Bundy and Chan, 2008] Alan Bundy and Michael Chan. Towards ontology evolution in physics. In Wilfrid Hodges and Ruy de Queiroz, editors, *Logic, Language, Information and Computation*, volume 5110 of *Lecture Notes in Computer Science*, pages 98–110. Springer Berlin / Heidelberg, July 2008.

[Colton *et al.*, 1999] S Colton, A Bundy, and T Walsh. HR: Automatic concept formation in pure mathematics. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden*, pages 786–791, 1999.

[McCarthy, 1964] J. McCarthy. A tough nut for proof procedures. Stanford Artificial Intelligence Project Memo 16, Stanford University, 1964.

[McNeill and Bundy, 2007] F. McNeill and A. Bundy. Dynamic, automatic, first-order ontology repair by diagnosis of failed plan execution. *International Journal On Semantic Web and Information Systems*, 3(3):1–35, 2007. Special issue on ontology matching.

[Shapiro, 1983] E.Y. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.

[Togia *et al.*, 2010] Theodosia Togia, Fiona McNeill, and Alan Bundy. Harnessing the power of folksonomies for formal ontology matching on the fly. In *Proceedings of the ISWC workshop on Ontology Matching*, November 2010.