

Supporting different inheritance mechanisms in ontology representations

Valentina A.M. Tamma and Trevor J.M. Bench-Capon,¹

1 Introduction

Ontologies have become increasingly important in sharing and reusing knowledge. In [17] an architecture of multiple shared ontologies for knowledge sharing was presented. In this architecture, resources no longer commit to one comprehensive ontology but instead are clustered together on the basis of the similarities they show in the way they conceptualise the common domain. This approach has the advantage of minimising the information loss when performing translations between the resources. Ontology clusters are then organised in a hierarchical fashion thus permitting concepts to be described at different levels of abstraction. Since different siblings can extend their parent cluster concepts in different ways the cluster hierarchy permits the co-existence of heterogeneous (sibling) ontologies. The proposed structure of multiple shared ontologies is based on inheritance mechanisms. Anomalies can arise from the use of inheritance, as illustrated in the literature ([1] and [15]). This paper analyses how inheritance problems can affect ontologies and proposes a methodology to deal, in a semi-automatic fashion, with the conflicts caused by the use of inheritance mechanisms, following the approach proposed by Goldszmidt and Pearl [6].

The proposal is to represent ontologies by an "enriched" frame-based language where the set of the slot's facets has been extended to encompass additional information required for a full understanding of a concept. Understanding a concept involves a number of things. First it involves knowing what can sensibly be said of a thing falling under that concept. This can be represented by associating *attributes* with the concept, and possible values that these attributes can take when applied to things of that type. Thus it is important to know that some birds fly and others do not. A full understanding of a concept involves more than this, however: it is important to know also what is true of a *prototypical* instance of a concept [13], to know that the prototypical bird flies. There are, however differences in how confident we can be that an arbitrary instance of a concept conforms to the prototype: it is a very rare mammal that lays eggs, whereas many types of well known birds do not fly. Understanding a concept also involves understanding how and which attribute values change over time: people may have eyes of various colours, but they do not change over time, in contrast hair colour does. This dynamic behaviour also forms part of the domain conceptualisation. We believe that this additional information needs to form part of the ontology. In this paper we concentrate on prototypical values, but also mention in passing, one method of dealing with dynamic values.

Representation of concepts within the ontology should be enriched

by information concerning the degree of strength associated with some properties and some measures of how likely that a value is associated to an attribute (*ranking*). This additional information enables us to deal with conflicts and inconsistencies due to inheritance mechanisms, as rules with a higher degree of strength and ranking can be given precedence. Other facets are introduced to represent how the attribute's value can change over time; this is based on the intuition that attributes can change their values either regularly in time or if an event occurs and that these changes contribute to enrich the attribute description.

The remainder of the paper is organised as follows: section 2 describes the problems caused by inheritance mechanisms, while section 3 presents Goldszmidt and Pearl's non-monotonic approach. Section 4 illustrates the extended knowledge model and section 5 the framework used to deal with inheritance conflicts and inconsistencies. Finally, section 6 illustrates conclusions and future work.

2 Providing a motivation for the additional facets

Representing knowledge about the world means representing the objects presumed or hypothesised to exist and be relevant in the world and their relationships. It has been argued that a knowledge representation is a surrogate [3], a stand-in for what is in the world. As any surrogate it is not completely *accurate*; it will necessarily contain simplifying assumptions because of the complexity of the natural world. Indeed even restricting to a subset of the natural world is still overwhelmingly complex. In this respect, a knowledge representation is also a set of *ontological commitments*. Ontological commitments determine not only the objects of the world but also what are the features of these objects that are relevant for the knowledge representation task.

Objects correspond to *classes*: all the member of a class share some common properties. Classes represent *concepts* (the terms can be used as synonyms).

The set of relevant concepts and the relationships holding between them are the *conceptualisation* [5] used to represent the world. When selecting a conceptualisation of the world some decisions have to be taken in order to establish what are the concepts to describe and how to describe them. Concepts identify sets of attribute-value pairs, where the attributes are those deemed important for the knowledge representation task and the values associated with them permit us to distinguish a concept from the others. Usually in the conceptualisation are also represented properties that are *generally* true for that concept, that is the conceptualisation usually describes a prototypical member of that class [13]. This gives rise to an important issue in knowledge representation: properties that are true for a class prototype are not necessarily all true for members of the class represented

¹ Department of Computer Science, The University of Liverpool, Chadwick Building, Peach Street, Liverpool, L69 7ZF, United Kingdom, email: {valli,tbc}@csc.liv.ac.uk

by the concept. Examples of such cases are frequent in everyday life; *Almost all* mammals give birth to live young, but three highly unusual mammals (monotreme) do not. Analogously, birds ability to fly is a property that is generally true; it is a property describing the prototypical bird. This type of information on the descriptive strength of properties should be encompassed in the conceptualisation of the domain and thus in the ontologies derived from it.

An ontology is "*an explicit specification of a conceptualisation*" according to Gruber [7]. That is, the conceptualisation refers to an abstract model of some phenomenon in the world by identifying the concepts that are relevant to that phenomenon; in the ontology, the type of concepts used to describe the phenomenon and the constraints on their use are *explicitly* defined [14]. Ontologies representations should include ways to represent how generally is shared a property among the members of a class. The ability of ontologies to distinguish not only between *hard* statements like "Elephants are animals" and *soft* ones like "Birds fly", but also between degrees of "softness", is crucial for reasoning about the knowledge represented in the ontology. This reasoning can prove helpful in dealing with problem arising from of the hierarchical organisation of concepts in ontologies. Concepts in ontologies are hierarchically organised through the IS-A relationship, with a partial order relation that is the ontology's main structure and that is further enriched by attributes, and by relationships or functions relating concepts. The IS-A relationship introduces also the powerful notion of *inheritance of properties*. Properties are shared by concepts either in their original form or modified in order to give the inheriting class, known as *sub-class*, a more restrictive definition than that provided by the parent concept. Furthermore other properties can be added to form more specialised concepts.

Anomalies arising from inheritance mechanisms have been illustrated in the literature ([1] and [15]), where a distinction is made between *single inheritance* and *multiple inheritance*. The former permits a concept to inherit properties from one parent only and can cause *default conflicts* while the latter permits a concept to inherit properties from more than one parent and can cause inconsistencies in inherited attribute values. In [2], default values are defined as a way to deduce information about a concept if the information is consistent with what is already known about the concept. Reasoning about defaults can become extremely problematic when only *strict inheritance* is allowed, that is when the IS-A link amounts to logical implication or set inclusion. Then, more specific information cannot overrule information obtained from more general classes thus causing wrong conclusions to be inferred. A defeasible approach [15] permits the more specific information to overrule the more general information thus solving the conflict.

Other kind of conflicts can arise when multiple inheritance is supported and conflicting information is inherited from two or more general concepts. In this case a choice has to be made about which value has to be associated with the attribute. This choice can be made by knowledge engineers, or the value can be (semi) automatically provided by determining the property degree of "softness". The same inconsistency problems caused by supporting multiple inheritance can be encountered when trying to integrate ontologies developed for different purposes (the word *integration* is used here to summarise all the possible meanings that the term takes in the ontological engineering field, and that are illustrated in [12]). In fact, with ontology integration an attempt is made to relate concepts in different ontologies. Concepts to relate can be described by the same attributes, but inconsistent values may be associated with them. So, when integrating ontologies, a crucial issue is to choose, among the inconsistent candidates, the value to associate with an attribute. An example of prob-

lems encountered while trying to integrate two or more ontologies can be found in [4]. Once again, the choice is made by knowledge engineers performing the integration who can be assisted by some tool that (semi) automatically chooses the most promising attribute's value among a set of candidates.

3 Reasoning with conflicts

To reason with conflicts and inconsistencies, the approach by Pearl [11] and lately by Goldszmidt and Pearl [6] is followed. The main idea of this approach is that knowledge from an inheritance structure can be associated with a probability expressing the degree of (dis)belief associated with that bit of knowledge. More formally, given the language L of the inheritance structure (that for Pearl is the language of propositional formulas), every sentence in L corresponds to a set of possible *worlds*, where a world is a conjunction of all the properties describing a typical individual in the domain. As some worlds are definitely more typical than others it is necessary to express the differences between all the possible worlds. This is obtained by weighing every world by assigning it a probability ε , which defines a probability distribution P over L . All the inheritance rules such as $Elephant(x) \rightarrow Animal(x)$ impose restriction conditions on P in the form of extreme conditional probability infinitesimally close either to 0 or to 1, where the closer to 1 the probability, the higher the number of individuals (and so sub-classes) inheriting the property. So, if we consider the inheritance rule $Mammal(x) \rightarrow Gives-birth-to-live-young(x)$, this means $P(Gives-birth-to-live-young(x)|Mammal(x)) \geq 1 - \varepsilon$ that for ε arbitrarily small is close to 1, meaning that if all is known is that x is a mammal, than x almost certainly inherits the property of giving birth to live young.

However, decisions can be taken also by approximating the full precision provided from this framework; measures of the degree of belief are taken on a logarithmic scale and beliefs that map into two different values are considered as being of different order of magnitude. Let $P(\omega)$ be the probability distribution defined over a set Ω of possible worlds; if we write the probability $P(\omega)$ as a polynomial in ε (that is $P_\varepsilon(\omega) = 1 - c_1\varepsilon$, or $\varepsilon^2 - c_2\varepsilon^4$ and so on) then the ranking function $\kappa(\omega)$ is defined as the power of the most significant ε -term in $P_\varepsilon(\omega)$. This permits to reason about degree of "softness" about a statement. The rank roughly corresponds to linguistic quantifiers such as *believable*, *unlikely*, *very rare* etc. In fact for $\kappa(\phi) = 0$ it means that both ϕ and $\neg\phi$ are *equally possible*, for $\kappa(\phi) = 1$ it means that $\neg\phi$ is believed, for $\kappa(\phi) = 2$ it means that $\neg\phi$ is *strongly* believed, for $\kappa(\phi) = 2$ it means that $\neg\phi$ is *very strongly* believed and so on.

An inference system (*system* - Z^+) based on the ranking of the probabilities has been developed in [6]. This system is able to draw plausible conclusions in most of the cases by a technique known as *z-entailment*, [6] which guarantees that conclusions in inheritance rules will receive high probabilities whenever the premises receive sufficiently high probabilities. This system can compute the priorities of inheritance rules and provides also consistency checks. *Z-entailment* is enriched with capabilities to handle variable-strength thus allowing to state some defaults "more strongly" than others.

The specification of the inheritance rules is made by associating with each rule a parameter δ which expresses the *degree of strength* of the rule. The ranking is computed as the sum between a *priority function* Z^+ associated δ with an inheritance rule and the degree of strength δ , each of which reflects different considerations to be taken into account when drawing conclusions about inheriting properties. The degree of strength δ_i associated with an inheritance rule $r_i = \phi_i \rightarrow \psi_i$

establishes the relative strength with which ψ_i is committed to be accepted in the context of ϕ_i while the priority $Z^+(r_i)$ expresses the degree of surprise concerning the finding of a world in that violates r_i , which includes also the degree of surprise associated with ϕ_i . Therefore, for each rule $r_i = \phi_i \rightarrow \psi_i$ the Z^+ ordering is determined by both the degree of strength and the ranking function. This type of ordering guarantees that features of more specific contexts override conflicting features of less specific order, thus allowing to solve the problems due to conflicting default information in single inheritance structures. Furthermore, whenever the ranking functions associated with the rules do not permit us to distinguish between inheritance rules, because no specificity consideration is made, the Z^+ ordering depends on the degree of strength alone, therefore permitting preference of one inheritance rule over the other(s). In this way the system can deal with types of multiple inheritance conflicts, where conflicting values are inherited for the same attribute from different parents.

4 The extended knowledge model

So far, all the efforts to deal with inconsistencies have been performed by hand by a knowledge engineer who is expert in the domain that is being described, and who can thus associate the correct default value to an attribute. Choosing between several conflicting defaults requires an extremely rich semantics. For this reason performing the choice automatically is quite unrealistic, but a more realistic possibility is a semi-automatic approach, where an inferential system presents the knowledge engineer with a list of sound choices (according to the inference process), but leaves the actual choice to the domain expert.

The model of knowledge used to represent the ontology plays a crucial role in the framework proposed in this paper, as it provides the elements necessary to apply the Goldszmidt and Pearl inference process. The proposed knowledge model is frame-based [10]. Our model is based on *classes*, *slots*, and *facets*. *Classes* correspond to concepts and are collections of objects sharing the same properties, hierarchically organised into a multiple inheritance hierarchy, linked by *is-a* links. Classes are described in terms of *slots*, or attributes, that can either be sets or single values. A slot is described by a name, a domain, a value type and by a set of additional constraints, here called *facets*. Facets can contain the documentation for a slot, constrain the value type or the cardinality of a slot, and provide further information concerning the slot and the way in which the slot is to be inherited by the subclasses. Our framework suggests the introduction of a set of facets that describe in detail the attribute and its behaviour in the concept description to accommodate different inheritance mechanisms, both within and between ontologies, and changes over time. This additional information is to be used in case of inconsistencies as a guide towards the most reasonable and informed suggestion to be presented to the domain expert, who will then validate such suggestion. The facets we introduce are:

- **Value:** There are three possibilities:
 - If the concept that is being defined is very high in the hierarchy (so high that any distinction based on the attribute's value is not possible), then **Value** is equal to **Domain**;
 - If the concept is still general, but it is possible to determine that it can have different attribute values for its children then **Value** is set equal to **Sub-domain** \subset **Domain**;
 - If the concept is defined in terms of a specific value for an attribute then **Value** is set $v \in$ **Domain**.

For the *third case* only further information about the type of value (see next item) or the degree of strength (see item below) can be added;

- **Type of value:** $\{Necessary, Prototypical, Inherited, Distinguishing\}$. An attribute's value is a *Necessary* one if the value is true for all concept's children. It describes necessary conditions in the concept's description. An attribute's value is a *Prototypical* one if the value is generally true for any children of the concept that is being defined, that is the value is generally true for any prototypical instance of the concept, but exceptions are permitted with a degree of softness expressed by the facet *Ranking*. An attribute's value can be *Inherited* from some super concept or it can be a *Distinguishing* value, that is a value that differentiates among siblings;
- **Degree of strength:** a number describing how relevant is, in the concept's description, the property represented by the attribute. For example, to reason about birds ability to fly, the attribute *species* is more relevant than the attribute *feather colour*;
- **Ranking:** an integer describing the probability ranking associated to the fact that the attribute takes the value specified in the facet **Value**. The possible values are for this facet are 1: *All*, 2: *Almost all*, 3: *Most*, 4: *Possible*, 5: *A Few*, 6: *Almost none*, 7: *None*. So, to represent the soft statement *Birds fly* we could add a slot, **Fly** that takes value *Yes* with *Degree of strength* equal to "Most";
- **Change frequency:** $\{Regular, Once\ only, Volatile\}$. This facet describes how often an attribute's value changes. If the information is set equal to *Regular* it means that the value changes at regular time intervals; if set equal to *Once only* it indicates that only one change is possible, and finally *Volatile* indicates that the attribute's value can change more than once. If the change frequency is *Regular* then the time interval is specified otherwise the event causing the attribute to change is specified;
- **Time interval:** This information can either be empty (if the change frequency is not *Regular*) or it contains the time interval between two changes;
- **Event:** This facet is either empty (if the change frequency is *Regular* and the time interval is set) or it is the set of events E that causes a change in the attribute's value. The logical theory chosen to reason about events is the *Event Calculus*, [8] and the information **Event**= e_i is interpreted as one of the following Event calculus expressions:

1. $Hold(before(e_i, P))$ that is, the property P holds *BEFORE* the event e_i ;
2. $Hold(after(e_i, P))$ that is, the property P holds *AFTER* the event e_i ;

where the interpretation is decided on the information **Event Validity** (see below). For each event $e_i \in E$ we specify also the *Event Property* and the *Event Validity* facets as follows:

- **Event Property:** $\{V\}$. This facet describes the value taken by the attribute before or after the event E. If this bit of information is empty it means that the event E causes a change in the attribute's value that cannot be specified, maybe because the value can be identified only by considering the instances of the concept;
- **Event Validity:** $\{Before, After\}$. It states whether the property V specified in the item above holds before the event E or after the event E.

The above facets describe how crucial the slot is in characterising a class, and what conditions determine a change in the value of the slot

for that class. These changing conditions are used to query the domain expert while solving default inconsistencies to try to associate to a slot a value as close to the true one as possible. These facets could also be used by the domain experts to learn more about the attribute they are dealing with.

5 The framework to deal with inheritance conflicts

When dealing with heterogeneous resources, mismatches in the names of concepts and attributes [16] occur. The first step of our framework consists of resolving names mismatches, in a way similar to that reported in [4]. This is necessary to avoid cases of implicit inconsistencies, where attributes describing two parent concepts are denoted with different names, while describing the same property. Then the attempt to relate the concepts in the ontologies composing the structure can begin.

In the remainder we present the steps composing this framework, explaining how a system can resolve inconsistencies when trying to build multiple shared ontologies. Ontologies are assumed to be represented by the knowledge model illustrated above. The domain user **DE** interacts with the system in several steps:

- The first step of our framework consists of scanning both the class names and the slot names in all the ontologies to find possible synonyms. Synonyms are evaluated intensionally, selecting them on the basis of a general thesaurus such as WordNet [9]. For the attributes, however, also an extensional check is performed, by checking the similarity in the attribute's domains.
- Once the name mismatches are resolved, the system proceeds both bottom-up and top down trying to relate classes. When it finds two or more classes that are suitable parents for the class the system is handling, then a consistency check is performed, according to the technique by Goldszmidt and Pearl [6];
- If an inconsistency is detected then the *priority functions* (see section 3) for the inheritance rules are computed on the grounds of both the *rankings* of probabilities and the *degree of strength*. These facets permits to solve both default conflicts and inconsistencies due to either multiple inheritance or to the integration of diverse ontologies. If both these facets are present then the system goes to the following step, otherwise it asks the **DE** to insert either the ranking or the degree of strength for the slot.
- After all the priority functions are computed and ordered, the system presents the **DE** with the slot's value with the best score. If one of the slot's facets contains information concerning events that can cause the attribute to change this information is also presented to the **DE** who is request to validate the events.
- The **DE** decides whether to accept the system suggestion or to ask the system to present the list of possible choices in rank order.

6 Conclusion and future work

This paper has presented a semi-automatic framework to deal with inheritance conflicts and inconsistencies while integrating ontologies. This framework represents ontologies by a frame-based language where the classical set of slot's facets is extended to encompass other information in order to associate with each attribute a degree of strength and other information concerning the behaviour of the attribute, thus permitting to deal with default conflicts and inconsistencies. After analysing the motivation for extending the frame-based language, we have presented a formal approach to deal with inconsistencies using the framework here described.

This framework provides domain experts trying to integrate different ontologies with a tool that checks the inconsistencies and presents them with a list of suggestions that are evaluated according to a priority function. The final choice is always left to the domain experts, but the system provides them with a set of possible choices and with information concerning how and when the attribute changes. Future work will concentrate on extending the framework by introducing some form of temporal reasoning based on event logics that extend the facets.

Acknowledgment

This research is funded by BT, United Kingdom.

REFERENCES

- [1] R.J. Brachman, 'On the epistemological status of semantic networks', in *Readings in Knowledge Representation*, eds., R.J. Brachman and H.J. Levesque, 191–215, Morgan Kaufmann, Los Altos, CA, (1985).
- [2] B. Carpenter, 'Skeptical and credulous default unification with application to templates and inheritance', in *Default reasoning and Lexical Organization*, eds., T. Briscoe, A. Copestake, and V. de Paiva, Cambridge University Press, (1993).
- [3] R. Davis, H. Shrobe, and P. Szolovits, 'What is a knowledge representation?', *AI Magazine*, **14**(1), 17–33, (1993).
- [4] N. Friedman Noy and M.A. Musen, 'SMART: Automated support for ontology merging and alignment', in *Proceedings of Knowledge Acquisition Workshop KAW'99*, Banff, Canada, (1999).
- [5] M.R. Genesereth and N.J. Nilsson, *Logical foundations of Artificial Intelligence*, Morgan Kaufmann, 1987.
- [6] M. Goldszmidt and J. Pearl, 'Qualitative probabilistic for default reasoning, belief revision, and causal modelling', *Artificial Intelligence*, **84**(1-2), 57–112, (1996).
- [7] T. R. Gruber, 'A translation approach to portable ontology specifications', *Knowledge Acquisition*, **5**(2), 199–220, (1993).
- [8] R. Kowalski and M. Sergot, 'A logic-based calculus of events', *New Generation Computing*, **4**, 67–95, (1986).
- [9] G.A. Miller, 'Nouns in wordnet: a lexical inheritance system', *International Journal of Lexicography*, **3**(4), 245–264, (1990).
- [10] M. Minsky, 'A framework for representing knowledge', in *Cognitive Science*, eds., A. Collins and E.H. Smith, 191–215, Morgan Kaufmann, Los Altos, CA, (1992).
- [11] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA, revised second printing edn., 1988.
- [12] H.S. Pinto, A. Gómez-Pérez, and J.P. Martins, 'Some issues on ontology integration', in *Proceedings of the IJCAI'99 Workshop on Ontology and Problem-Solving Methods: Lesson learned and Future Trends*, ed., V.R. Benjamins, volume 18, pp. 7.1–7.11, Amsterdam, (1999). CEUR Publications.
- [13] E.H. Rosch, 'Cognitive representations of semantic categories', *Journal of Experimental Psychology: General*, **104**, 192–233, (1975).
- [14] R. Studer, V.R. Benjamins, and D. Fensel, 'Knowledge engineering, principles and methods', *Data and Knowledge Engineering*, **25**(1-2), 161–197, (1998). Interesting paper with a good overview on knowledge engineering.
- [15] D.S. Touretzky, *The Mathematics of Inheritance Systems*, Morgan Kaufmann, 1986.
- [16] P.R.S. Visser, D.M. Jones, T.J.M. Bench-Capon, and M.J.R. Shave, 'Assessing heterogeneity by classifying ontology mismatches', in *Formal Ontology in Information Systems. Proceedings FOIS'98, Trento, Italy*, ed., N. Guarino, pp. 148–162, Amsterdam, The Netherlands, (1998). IOS Press.
- [17] P.R.S. Visser and V.A.M. Tamma, 'An experience with ontology-based agent clustering', in *Proceedings of the IJCAI'99 Workshop on Ontology and Problem-Solving Methods: Lesson learned and Future Trends*, ed., V.R. Benjamins, volume 18, pp. 12.1–12.13, Amsterdam, (1999). CEUR Publications.