Ontology Refinement and Evaluation based on *is-a* Hierarchy Similarity

Takeshi Masuda

The Institute of Scientific and Industrial Research, Osaka University

Abstract. Ontologies are constructed in fields such as medical information and mechanical design. Building high-quality ontologies for use as knowledge bases and models for applications is important; however, it is difficult owing to the required knowledge of ontology and expertise in the target domain. Ontology construction and refinement consumes time and effort. To reduce costs, I developed an ontology refinement support system with two principal functions. The system can evaluate ontologies quantitatively and detect points for refinement and propose means to refine them. It evaluates ontology consistency in terms of classificatory criteria. To develop the system, I followed a guideline for building well-organized ontologies to the effect that "Each subclass of a superclass is distinguished by the values of exactly one attribute of the superclass." When an ontology is built following this guideline, the *is-a* hierarchies are similar. I used these similar *is-a* hierarchies to develop an ontology refinement system.

1 Problem Statement

In this study, I aim to develop an Ontology (i) Quality Refinement and (ii) Quantitative Evaluation system. I focus on consistency of criteria for *is-a* hierarchies [1]. This is a very important guideline for creating high-quality ontology.

First, the ontology refinement system (i) must have functions to detect components that must be refined and propose how to refine them. Using the refinement system, users can find inconsistent components in ontologies and obtain relevant refinement methods without examining every concept and property.

The second function is the quantitative evaluation (ii) of ontologies based on the proposed refinement system. This evaluation system allows us to obtain a quantitative indicator of the consistency of classification criteria. At present, coverage of a number of required concepts is used to evaluate ontologies quantitatively [2]. In addition to these indicators, this system allows evaluation from the perspective of classificatory consistency. This will enable us to search and select an ontology properly when developing a knowledge system. The system can also evaluate partially in an ontology. Thus, the system can be used for processing management while refining an ontology.

2 Relevancy

Ontologies are currently constructed in fields such as medical information [3] and mechanical design [4]. These ontologies are used as knowledge model schemata for application systems. Owing to such usage, the quality of these ontologies is an important factor for application systems, and construction of higher-quality ontologies is a major issue.

However, knowledge and experience regarding ontology construction and expertise in the target domain are necessary for building well-organized ontologies. For this reasons, it is not easy for beginners to construct good ontologies, and ontology construction and refinement support methods are expected. At present, there is research into systems that can correct formal errors in ontology. These systems are embedded in ontology construction tools, such as Protégé. However, we must investigate each concept to improve the quality of its definition. Thus, refinement and evaluation are expensive. It is for this reason that I aim to develop an ontology refinement and evaluation support system.

3 Related Work

There are some supportive methods for ontology refinement to improve ontology quality. One is a method for detecting and correcting formal errors in ontologies. Another is a method for refining the contents of ontologies. The former includes many consistency-checking methods and debugging functions for ontologies [5]. Poveda et al. collected common errors found in Web Ontology Language (OWL) ontologies, called ontology pitfalls, and developed a system for detecting these pitfalls and correcting some of them [6]. On the other hand, in regard to the latter, there are some guidelines [7, 8] and methods [9] for refining the contents of ontologies. Even with these guidelines or methods, human intervention is required to evaluate the correctness of ontology contents. That is, in these approaches, knowledge of the target domains and experience with ontology building are necessary to evaluate their contents. There is a method that requires relatively little human intervention [10], but that system requires external data to refine ontologies. Therefore, these methods are not sufficient to support the content refinement of ontologies. To overcome this problem, we propose an ontology contents refinement and evaluation support system.

4 Research Question

I develop an automatic ontology refinement system. As a refinement target, I focus on the consistency of is-a hierarchies, as recommended by the guideline in [1]. Developing this system requires answering the following two research questions. First, how do we find components that violate the criteria consistency guidelines for classification [1], and second, how can we formulate refinement proposals for each inconsistent component automatically.

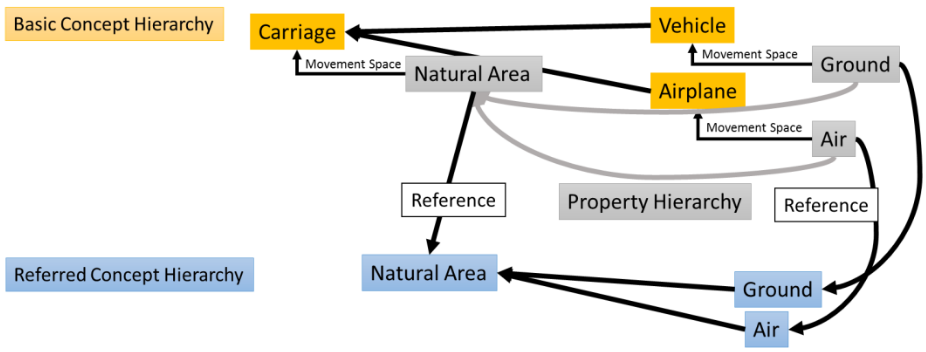

Fig. 1. Similarity among *is-a* Hierarchies

5 Hypotheses

I pose a hypothesis to answer the research questions: if subclasses are not classified by one attribute, then there are consistency errors in the ontology that can be automatically fixed through a comparison of is-a hierarchies.

I investigate this hypothesis by focusing on the criteria consistency guidelines for classification [1]. To follow the guidelines, I find a characteristic relationship among *is-a* hierarchies, that is, "In an ontology that follows the guideline, *is-a* hierarchies that have a reference relationship tend ideally to have similar structures [Fig. 1]." For example in Fig. 1, "Carriage" is a specialized "Vehicle" and "Airplane" (I call this the basic concept hierarchy), and these three concepts have properties in regard to "movement space." Thus, we can say that this hierarchy follows the guideline, because each subclass is specialized by one attribute ("movement space"), while each property refers to other concepts that are in the same ontology as the class constraint concept (range). These referred-to concepts also comprise an *is-a* hierarchy. We call this hierarchy the "referred concept hierarchy." We can see that these three hierarchies have one parent concept and three children concepts. For this example, "if the ontology follows the guideline, the structure of *is-a* hierarchies that have a reference relationship tend to be similar." Therefore, we can find inconsistent components in the ontology by comparing *is-a* hierarchies and make inconsistent components into consistent components through proposals for making them similar. This characteristic is based on structural information such as *is-a* and reference relations; hence, there is an advantage in that this hypothesis can apply without depending on the domains of ontologies.

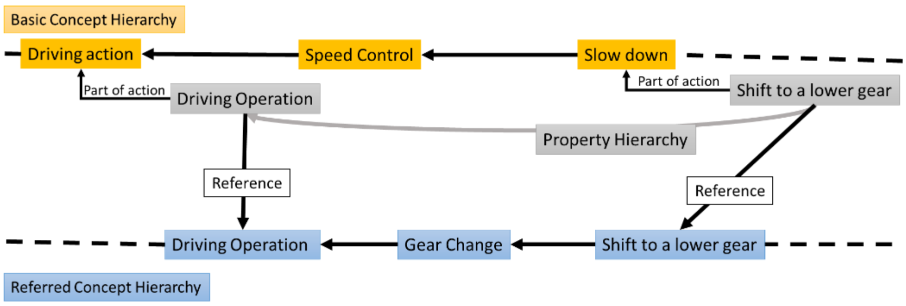

Fig. 2. Example Hierarchies that require refinement

Now, I show an example of the refinement of an ontology [Fig. 2]. In Fig. 2, there are three concepts in the basic and referred concept hierarchies, while there are two properties in the property hierarchy. Hence, these three hierarchies are not similar. The refinement system detects these components from the ontology and proposes a refinement method that makes them similar. In this case, the system proposes to "add a new property on '*Speed Control*' and use '*Gear Change*' as a class constraint." As a result of this proposal, the basic concepts, property, and referred concepts hierarchies become similar [Fig. 3.]. As this example illustrates, the refinement process involves two steps, as follows:

1. Detect candidate components to be revised, because they violate the guideline
2. Propose some methods for refining each detected component

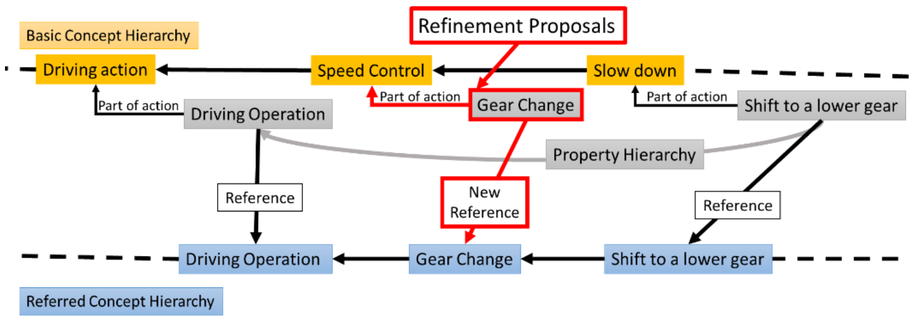

Fig. 3. Example of a refinement proposal

In this study, I develop an *is-a* hierarchy comparison method that detects refinement candidates using this hypothesis. There are four types of candidates to be refined. These groups are divided by the *is-a* structure [Fig. 4.]. These patterns are related to proposals for each refinement candidate. The types of hierarchy comparison methods are as follows.

1. Upper and lower concepts
2. Upper concept only

3. Lower concept only
4. Brothers concepts

In addition to types of refinement candidates, we consider refinement methods for each refinement candidate detected using the previously described comparison method. In essence, a refinement method is an addition of new concepts and properties. There are other refinement methods, such as deletion of existing concepts, but for now we regard existing concepts as correct and focus on the addition of concepts and properties. There are three patterns for adding concepts and properties.

1. Add a new property on an existing concept in the basic concept hierarchy
2. Add a new concept on the basic concept hierarchy, and add a new property on the concept created earlier on the basic concept hierarchy
3. Add a new concept on the referred concept hierarchy, and add a new property on an existing concept on the basic concept hierarchy using the concept created earlier as a class constraint

The refinement proposals are not compulsory, in contrast to formal error collection. Because these proposals are based on a guideline, there need not be strict observance. Consequently, there can be both reasonable and inappropriate proposals. In addition, we find that particular refinement candidates and methods have priority, and this priority has a relation to the structure of an *is-a* hierarchy. I think that this priority could be used to make more precise refinement proposals in the future.

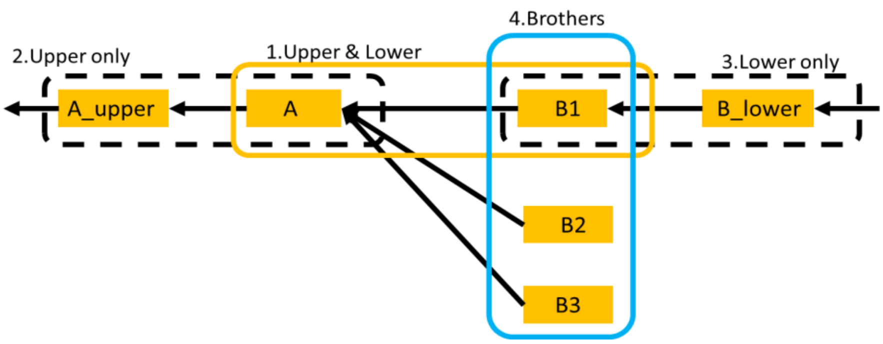

Fig. 4. Groups of candidates

6 Preliminary Results

I conducted a pre-experience to evaluate this refinement proposal system. The purpose of the experiment is to verify the hypotheses discussed in Section 5. This study was designed to assess refinement candidates.

I asked nine evaluators who have experience in creating ontologies and used 150 refinement candidates detected from six ontologies. Each refinement candidate was evaluated by three evaluators. There were multiple refinement proposals for each refinement candidate. The evaluators assessed whether these refinement proposals "agree" or "disagree." I consider that if a refinement candidate received at least one "agree," then the refinement candidate was detected correctly.

The results are shown in Fig. 5 for the types of candidates to be refined. These are types 1–3 of candidates in terms of the hypotheses discussed in Section 5. From these results, we see that the type "Upper & Lower Concept" received very good ratings. Approximately 80% of those candidates are considered to be refined by two-thirds of the evaluators, whereas the "Upper Concept only" type is comparatively less good. However, 25% of the candidates are considered to be refined by two-thirds of the evaluators. As a result of this experiment, our hypotheses regarding proposing refinement candidates automatically are confirmed. In addition to this, we could verify that there are differences of appropriateness among types of refinement candidates.

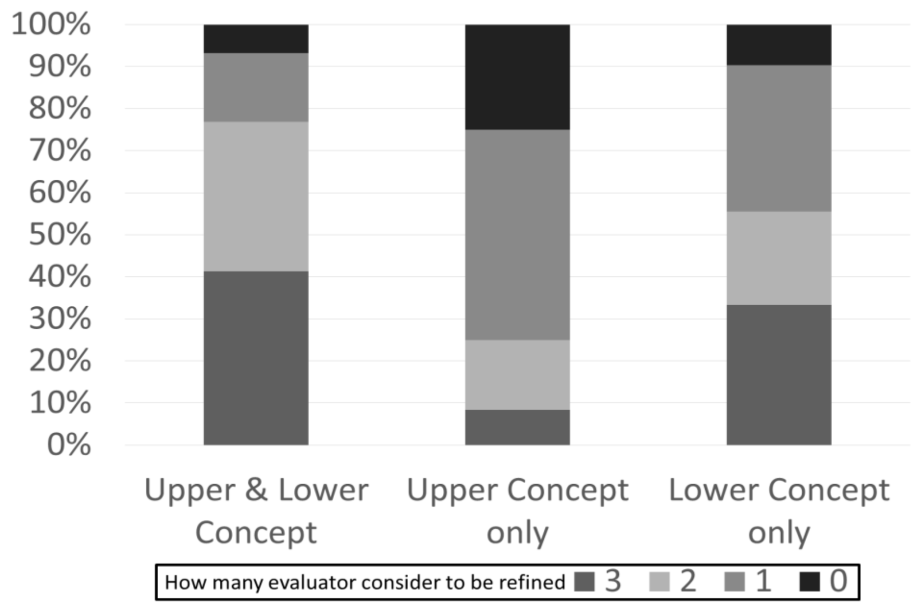

Fig. 5. Pre-experience results

7 Approach

From the preliminary results, we see that the refinement support system based on similarity among *is-a* hierarchies can detect inconsistent components from ontologies and propose refinement methods for each component automatically. I suggest that the refinement system could be applied to an ontology quality evaluation system, because it can find how many components do not follow the guideline. Thus, I think that the number of refinement candidates shows the quality of the ontology quantitatively.

To apply these numbers, we must solve three problems:

1. Weighting each refinement candidate according to its *is-a* hierarchy structure
2. Normalizing the differences in the target domain and the scale of ontologies
3. Determining how to show the evaluation results to users

At first, we start by weighting each refinement candidate by its *is-a* hierarchy structure. From the preliminary results, we see that there are differences of appropriateness among types of refinement candidates. For example, type A contains more appropriate candidates than group B. In addition, the abstraction level can affect the appropriateness. For example, if the candidates consist of top-level concepts, such as "*entity,*" "*thing,*" or "*process,*" then these proposals tend to be inappropriate. While there are more appropriate proposals if the refinement components consist of lower abstract concepts, that is specialized sufficiently. Thus, to apply a number of refinement candidates to evaluate ontology, it is necessary to weight each refinement candidate by an *is-a* hierarchy structure.

Secondly, we consider normalizing the differences in the target domain and the scale of ontologies. We cannot evaluate ontologies using only one measure, because every ontology has a different domain and scale, for example, when a large-scale ontology is compared with a small-scale ontology. If the refinement system detected the same number of candidates from these ontologies, there is a high probability that a large ontology is better. We must find such kinds of factors and consider how those factors affect evaluation.

Finally, we develop the ontology evaluation system. I am assuming two uses for this evaluation system. One use is the ranking of ontologies. The system orders ontologies using these scores, and then users can select a high-quality one. Another use is in a progress management system during the refining of an ontology. The system evaluates an ontology and scores it. Then, the user can verify which components are progressing.

8 Evaluation Plan

I plan to evaluate the evaluation system in practical use. The ontologies I propose to apply are of three different types. The first is a biomimetic ontology that is used for comparatively light tasks, such as searching the semantic web. The second is a sustainability science ontology that is used to share concepts among a number of people. The last is a clinical medicine ontology that requires the highest quality for a sophisticated medical information system. I will apply the refinement and evaluation system to these three ontology types, and these results will be evaluated by both ontology and domain experts. After this experiment, I will improve the system using the feedback from this experiment. For example, different degrees of strictness in refinement and evaluation are required. The system must be able to manage a range of requirements. After these evaluations and consequent improvements, I will open the system to the semantic web community and use many more ontologies.

9 Reflections

In current studies, there are many systems that correct formal errors, some of them mounted on ontology construction tools, such as Protégé. There are also some guidelines and refinement methods, such as OntoClean, for refinement of ontology definitions. However, since there are no automatic systems, we must consider each concept individually to find refinement components and methods. In my research, although the system can deal with only "classificatory criterion consistency," I developed a refinement system that detects refinement candidates and proposals automatically. To develop this system, I developed a comparison method among *is-a* hierarchies that have a reference relationship, and I can show this method has the ability to propose refinement candidates automatically.

In addition to ontology refinement, there are some existing ontology evaluation methods, but these evaluation methods use a scale of ontology or coverage of required concepts to evaluate ontologies quantitatively, whereas I developed a quantitative evaluation system in my research. This system allows evaluation from the perspective of classificatory consistency. This evaluation system is an application of the refinement system previously mentioned. We can use this evaluation system when we choose an ontology from many ontologies and manage processing ontology refinement.

References

1. Tutorial on ontological engineering - Part 2: Ontology development, tools and languages New Generation Computing, OhmSha & Springer, Vol.22, No.1, pp.61–96 (2004)
2. Ceusters, W. Pain Assessment Terminology in the NCBO BioPortal: Evaluation and Recommendations. In ICBO, pp. 1-6 (2014)
3. BioPortal, http://bioportal.bioontology.org/
4. Kitamura, Y., Koji, Y., Mizoguchi, R.: An ontological model of device function: Industrial deployment and lessons learned, J. Applied Ontology, Vol. 1, No. 3/4, pp. 237–262 (2006)
5. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner, Web Semantics: science, services and agents on the World Wide Web, Vol. 5, No. 2, pp. 51–53 (2007)
6. María Poveda-Villalón, Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez.: Validating Ontologies with OOPS! Knowledge Engineering and Knowledge Management Lecture Notes in Computer Science Volume 7603, pp 267–281 (2012)
7. OBO Foundry, http://www.obofoundry.org/principles/fp-000-summary.html
8. Noy, N.F., McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880 (2001)
9. Guarino, N., Welty, C.A.: An Overview of OntoClean, in Handbook on Ontologies, Springer Berlin Heidelberg (2009)
10. Zablith, F. Evolva: a comprehensive approach to ontology evolution. In Proceedings of the PhD Symposium of the 6th European Semantic Web Conference (ESWC-09), Heraklion, Greece, 944–948 (2009)