Ontology Refinement System for Improving Consistency of Classification among Brother Concepts

Takeshi Masuda, Kouji Kozaki, Kazunori Komatani

The Institute of Scientific and Industrial Research, Osaka University

Abstract. The consistency of classification is an indicator of ontologies' quality. In this paper, we focus on the consistency among brother concepts and develop a refinement system that finds inconsistent parts from a target ontology and propose methods to make such parts consistent. To find inconsistent parts and make proposals, the system compares a slot hierarchy and other two hierarchies that have reference relationship to the slot hierarchy.

1 Introduction

Nowadays, ontologies are constructed in various fields such as medical information, mechanical design and so on. These ontologies are used as a schema of knowledge models for application systems. Therefore a construction of better quality ontology is a considerable issue. However, both experience of ontology construction and expertise in the target domain are required to build well organized ontologies. Therefore, it is not easy for beginners to construct good ontologies. Because of these backgrounds, there are some systems that correct some formal errors in ontology and these are embedded in ontology editor like Protégé [1, 2, 3]. While in the case of quality improvement, we have to investigate each concept to check whether the concept should be refined or not. Therefore, we aim to develop the Ontology Refinement support system.

2 Ontology Refinement by Comparing *Is-a* Hierarchies

2.1 Similarity among *Is-a* Hierarchies

In order to develop a refinement method, we focus on an ontology development guideline that "Each subclass of a super class is distinguished by the values of exactly one attribute of the super class. [4]", and found the interesting characteristic of among *is-a* hierarchies under the guideline. We found that conceptual structures are similar to other *is-a* hierarchies in ontologies which follow the guideline. For example, in Fig.1, these 3 hierarchies, "Basic Concept Hierarchy", "Slot Hierarchy" and "Referred Concept Hierarchy", are following guideline and then their structure are similar. In this paper, our refinement system compare these 3 hierarchies and detect un-similar parts as a refinement candidates and make proposals to add new concepts for each candidates.

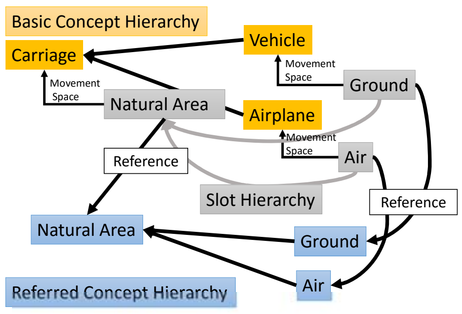

Fig.1 similarity among *is-a* hierarchy

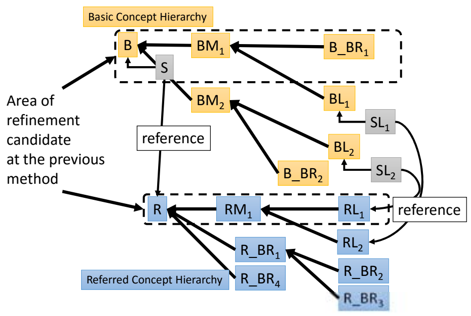

Fig.2 Comparing Brother Concepts

2.2 Comparing among Brother Concepts

In our previous method [5], we compared 3 hierarchies but it is only use super-sub relations whereas in this paper, we also consider comparing among brother concepts' hierarchies. To compare these 3 hierarchies with brother concepts, we focus on "Slot Hierarchy" as a basis. "Slot Hierarchy" consists of a certain slot "S" and its lower slots "SL_m" (m = 1 ~ M, M is a number of lower slots). These lower slots are specialized only once from "S". In this case, "Basic Concept Hierarchy" consists of concepts that has slots in slot hierarchy ("B", "BL_m"), these brother concepts ("B_BR_x", x = 1~ X, X is a number of brother concepts.) and middle concepts ("BM_a", a = 1~ A, A is a number of middle concepts.) (Fig.2.). "Referred Concept Hierarchy" consists likewise.

2.3 Patterns of Refinement Candidates

Comparing 3 hierarchies like Fig.2, if there are no brother concepts (B_BR_x, R_BR_y) that did not have (or be referred from) "S_m", these 3 hierarchies are not similar. So we can classify patterns of refinement candidate by existence of B_BR_x or R_BR_y. To sum up, patterns of refinement candidate are the following four. (i) $X > 0$ and $Y > 0$, (ii) $X = 0$ and $Y > 0$, (iii) $X > 0$ and $Y = 0$, (iv) $X = 0$ and $Y = 0$. (X is a number of bother concepts in "Basic Concept Hierarchy, Y is a number of bother concepts in "Referred Concept Hierarchy". We do not make any proposals for (iv) because it is already similar.)

2.4 Limitations on Refinement Proposals

In our refinement method, refinement proposals are 3 types as follows. A correspondence between refinement candidates and proposals are shown at Table.1.

(a). add new Slot
(b). add new concept to Basic Concept Hierarchy and new Slot
(c). add new concept to Referred Concept Hierarchy and new Slot

Table.1. correspondence between refinement candidates and proposals

		Where to add new slot	Concept chosen as class constraint	Acceptable Candidate Type	A number of proposals
a	1	B_BRx	R_BRy	(i)	X*Y
	2		R_Lm	(i), (iii)	X*M
b	1	New concept in BCH	R_BRy	(i), (ii)	X(not leaf)*Y
	2		Rn_Lm	(i), (ii), (iii)	X(not leaf)*M
c	1	B_BRx	New concept in RCH	(i), (iii)	Y(not leaf)*X

BCH : basic concept hierarchy, RCH : referred concept hierarchy
B_BRx: brother concepts that don't have slot (x : 1..X)
R_BRy: brother concepts that are not referred (y : 1..Y)
M : a number of referred concepts

At this comparison with brother concept hierarchies, a number of refinement proposal become enormous. Because there are much more comparison concepts in brother concepts hierarchies than comparison among upper and lower concepts. For example, I assume that some numbers M = 2 (lower slot of "S"), X = 40 (brother concept in basic concept hierarchy "B_BR$_x$") and number of leaf concepts is 18, Y = 3 (brother concept in referred concept hierarchy R_BR$_y$) and number of leaf concepts is 2. This example is X > 0 and Y > 0, then candidate type is (i). From table.1, all proposals, a1 ~ c1, are suggested. Therefore total number of proposals for this example is 328. This number is quite large to consider as refinement proposal, even this example has a few brother concept in referred concept hierarchy. If there are same number of brother concepts in referred concept hierarchy, a number of proposals can be over a thousand per a candidates. For the above reasons, we consider 2 limitations on refinement proposals.

limitation1: Compare brother concepts that have same parents

limitation2: Compare brother concepts that are specialized same level.

3 Evaluation

3.1 Evaluation Methods

We conducted a pre-experience to evaluate this refinement proposal method. We use a race ontology that is made by author and this ontology contains 213 concepts. This experience was designed to asses 2 points: (1) how many appropriate candidates are detected and (2) how the 2 limitations work.

3.2 Results and Discussions

(1) How many appropriate candidates are detected.

Table.2 shows the result. By previous method, 149 candidates are detected and 18% are proposed correctly. While by new method, 11 candidates are detected and 63% are proposed appropriately. Both new and previous method's candidates are not repeated. However average number of proposals are 1154, which is too much to consider.

(2) How the 2 limitations work

Table.3 shows the result. By “limitation1”, average number of proposals decreased to 13, it is 100 times fewer than no limitation. In this case, 4 candidates still have correct proposal under “limitation 1”, but 3 candidates cannot be proposed any suggestions.. While by “limitation 2”, average proposals also decreased to 377 but it is still enormous to see. But it has 5 correct candidates, it is better than “limitation1”.

Table.2. Comparison between previous and new refinement method

	All candidates	A number of Candidates that have correct proposals	A percentage of Candidates that have correct proposals	Average number of proposals
Previous method	149	27	18%	7
New method	11	7	63%	1154

Table.3. Result of the 2 limitations

	All Candidates	Candidates that have correct proposals	Candidates that don't have any proposals	Average number of proposals
No limitation	7	7	0	1154
Limitation 1	7	4	3	13
Limitation 2	7	5	2	377

4 Conclusion

In this paper, we focus on similarity among brother concepts’ hierarchies. However a number of refinement proposals are drastically increased by the explosion of combination. Then we provide 2 types of limitations to prevent this explosion. As a result, we could improve an accuracy of proposals and suppress the number of proposals. In future work, we consider some refinement candidates that we cannot make any proposal and integrate the previous refinement method and the new refinement method.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers JP25280081, JP 26240033, JP24120002.

References

1. [Sirin 07] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and 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)
2. [Horridge 11] Horridge, M.: Justification based explanation in ontologies, PhD thesis, University of Manchester (2011)
3. [Noy 01] Noy, N. F., Sintek, M., Decker, S., Crub´ezy, M., Fergerson, R. W., and Musen, M. A.: Creating Semantic Web Contents with Prot´eG´e-2000, IEEE Intelligent Systems, Vol. 16, No. 2, pp. 60–71 (2001)
4. [Mizoguchi 06] Tutorial on ontological engineering - Part 2: Ontology development, tools and languages New Generation Computing, OhmSha & Springer, 22(1), pp.61-96 (2004)
5. [Masuda 15] Masuda, T., Kozaki, K., Komatani, K. : Evaluation of Method to Improve Ontology Consistency based on Similarities among is-a Hierarchies in Referring Relation, JSAI2015, 2M1-3,2015.