Integrating Relations for a Domain Ontology

Jin-Xia Huang, Ji-Ae Shin*, Key-Sun Choi

Semantic Web Research Center/Computer Science Department,
Korea Advanced Institute of Science and Technology, Daejeon, Korea
* Information and Communications University, Daejeon, Korea
hgh@world.kaist.ac.kr, jiae@icu.ac.kr, kschoi@cs.kaist.ac.kr

Abstract. This paper addresses an approach for building a relation hierarchy in order to integrate relations from different sources. A relation tree is generated by mapping relations to WordNet synsets, on which duplicated relations are eliminated. As criteria for judging relations not absolutely required for a domain, popularity and uniqueness are proposed. Our research shows that use of a relation hierarchy makes the process of judging necessary relations of a domain much simpler and more efficient.

Keywords: Relation Integration, Relation Hierarchy, Ontology Building

Motivation. Which semantic relations will be adopted for domain ontologies normally decided by human developers subjectively [1]. This lack of objectivity becomes a serious problem when the target ontology is of a large scale. Many human developers and sometimes several teams are involved in the process, and several existing ontologies are adopted, these often lead to the generation of many duplicates or unnecessary relations. Indeed, how to approve proper relations is one of the major issues in building ontology.

Hundreds of relations are proposed for an IT domain ontology. Some of these are duplicate or semantically overlapped (e.g., establishDay and establishDate; instrument, DeviceFor and EquipmentOf); some are unsuitable for the IT domain (e.g., useBody, or beAdequateFor); and none are organized in a relation hierarchy, even for relations which are obviously sub- and super-relations to each other (e.g., time and playTime; isEquipmentOf and isElectronicEquipmentOf). It is difficult for human developers to handle hundreds of relations without a relation hierarchy; and it is also difficult to build a relation hierarchy manually.

The task of this paper is, given relation triples from different sources including existing ontologies and IT domain ontologies currently under development, to generate an integrated relation hierarchy in which duplicate relations are eliminated automatically. With the generated relation hierarchy, unnecessary relations can be distinguished by human developers more efficiently.

Building a relation hierarchy and eliminating duplicate relations. To build relation hierarchy, analyses on word-formation patterns are performed on input relations: first, the relations are segmented into words by capitalized letters (e.g., playTime → play time); then, relation word formations are analyzed with MiniPar [2]
and against patterns to determine their word dependency structures (e.g., \textit{play Time} $\rightarrow$ \textit{play}: modifier, \textit{time}: headword).

The analyzed relations are mapped to WordNet by performing word sense disambiguation (WSD) on relation headwords. SenseLearner \cite{3} is adopted for WSD. Relations which have the same headwords are grouped together, and relations with the shortest names are treated as super-relations of the others (e.g., \textit{time} is treated as a super-relation of \textit{playtime}). By eliminating all synsets to which no relation belongs, a primitive relation hierarchy is generated.

Relations which share the same headwords, modifiers and modificands are considered as duplicate relations. All these relations are eliminated except for the one with the shortest name (e.g., \textit{time}, \textit{isTimeOf} $\rightarrow$ \textit{time}).

638 relations from SUMO \cite{4}, MILO, SUMO Computing Services Ontology\footnote{Hereafter these three ontologies are referred to as “SUMO3.”} \cite{5}, 20 relations from ConceptNet \cite{6}, and 471 relations from IT domain ontology\footnote{We extract all subclasses of the two relation-related top nodes (\textit{Relation} and \textit{InheritableRelation}) from SUMO3, and then treat all sub-relations and instances of these classes as SUMO3 relations. As a result, 619 relations are extracted. Including an extra 19 relations, which are defined in English SUMO relation templates \cite{5} but not included in SUMO3, a total of 638 relations are adopted in tests carried out for this paper.}, are adopted as input relations. As a result of above processing, a relation hierarchy with 928 relations are generated, from which 181 relations out of 1,109 input relations are already eliminated automatically.

**Distinguish unnecessary relations.** The two criteria of \textit{popularity} and \textit{uniqueness} are proposed to distinguish unnecessary relations from the hierarchy. \textit{Popularity} imposes that the frequency of relations in a domain ontology is high enough. \textit{Uniqueness} imposes that there should be no two relations with overlapped semantics; relations that share the same hypernym in the generated relation hierarchy (e.g., \textit{instrument}, \textit{DeviceFor} and \textit{EquipmentOf}) might not be unique, although not necessarily. The final judgment on unnecessary relations should be made by human developers.

**References**

5. The SUMO, MILO and Computing Services Ontology in KIF format. \url{http://ontology.teknowledge.com/}

\footnote{There are 778 relations proposed for our IT domain ontology, but 307 of them are not counted in test because they are already included in the relations of SUMO3.}