Structural Distance between $\mathcal{EL}^+$ Concepts*

Boontawee Suntisrivaraporn
School of Information and Computer Technology
Sirindhorn International Institute of Technology, Thammasat University, Thailand
sun@siit.tu.ac.th

Abstract. The inexpressive Description Logics in the $\mathcal{EL}$ family have been successful mainly due to their tractability of standard reasoning tasks like subsumption, adoption in modeling of biomedical ontologies, and standardization as an OWL 2 EL profile. This paper proposes two enhanced subsumption algorithms that not only test whether a particular subsumption holds but also give a numeric indicator showing the structural distance between the two concepts in the subsumption relationship. Structural distance is defined in terms of the effort required to obtain the subsumption in question. Both algorithms extend the standard subsumption algorithm for $\mathcal{EL}^+$ by an axiom labeling technique, originally proposed for finding justifications.

Keywords: Description logic, Subsumption, Structural distance.

1 Introduction

Description Logics (DLs) are a family of logic-based knowledge representation formalisms, which can be used to develop ontologies in a formally well-founded way. This is true both for expressive DLs, which are the logical basis of the Web Ontology Language OWL 2, and for lightweight DLs of the $\mathcal{EL}$ family, which are used in the design of large-scale medical ontologies such as SNOMED CT and form one of the W3C-recommended tractable OWL profiles, OWL 2 EL. One of the main advantages of employing a logic-based ontology language is that reasoning services can be used to derive implicit knowledge from the one explicitly represented. DL systems can, for example, classify a given ontology, i.e., compute all the subsumption (i.e. subclass–superclass) relationships between the concepts defined in the ontology and arrange these relationships as a hierarchical graph. The advantage of using a lightweight DL of the $\mathcal{EL}$ family is that classification is tractable, i.e. a subsumption hierarchy of a given ontology can be computed in polynomial time.

Computation of a subsumption hierarchy is crucial to the ontology development life cycle because it graphically depicts inferred subsumption among those explicitly stated in the ontology. Apart from the fact that a particular subsumption in the hierarchy is stated or inferred, however, the ontology developer knows

---

* This work is partially funded by Thailand Research Fund under grant MRG5380245 and the NRU project of Thailand Office for Higher Education Commission.
nothing concerning the effort required to compute it. Stated subsumptions are trivial to compute and do not require actual reasoning beyond transitive closure computation. On the other hand, inferred subsumptions may or may not be difficult to compute, and this fact is completely invisible to the ontology developer. For example, w.r.t. an example ontology \(O_{\text{med}}\) in Fig. 1, both Pancarditis and Pericarditis are inferred subsumees of HeartDisease and will be positioned alongside each other beneath it in the hierarchy. Though observed from the subsumption hierarchy to be symmetric, the structures of their definitions differ which directly affect the effort required to compute them. At the minimum, it requires 4 steps to obtain the former subsumption, whereas the latter requires 6 steps. For larger ontologies with more complex nested structures, this difference will be greater. We believe that the structural distance between concepts in a subsumption, i.e. the minimum effort required internally by the reasoning algorithm, has merit for ontology design. When the subsumption hierarchy is equipped with this information, the ontology developer is better-informed about subsumption links as to how structurally distant they are from the subsumer. Statistics on the structural distance can help visualize locations of difficult subsumptions and may reveal structural discrepancies in the ontology.

In this paper, we present two algorithms for computing structural distance for a given subsumption. Both extend the polynomial-time subsumption algorithm for \(\mathcal{EL}^+\) with axiom labeling strategies, originally proposed for axiom pinpointing \([1,4]\). The first algorithm is shown to retain tractability of the original algorithm, but whenever the number of axioms is considered, exponential blowup arises. The rest of the paper is organized as follows. The next section introduces the DL \(\mathcal{EL}^+\) together with its polynomial-time subsumption algorithm. Our extensions for computing structural distance are presented in Section 3. Related work is discussed in Section 4, and conclusion is given in the last section.

2 Preliminaries

In DLs, concept descriptions are inductively defined with the help of a set of constructors, starting with a set \(\mathcal{CN}\) of concept names and a set \(\mathcal{RN}\) of role names. \(\mathcal{EL}^+\) concept descriptions are formed using the constructors shown in the upper part of Table 1. An \(\mathcal{EL}^+\) ontology or TBox is a finite set of general concept inclusion (GCI) and role inclusion (RI) axioms, whose syntax is shown in the lower part of Table 1. Conventionally, \(r, s\) possibly with subscripts are used to range over \(\mathcal{RN}\), \(A, B\) to range over \(\mathcal{CN}\), and \(C, D\) to range over concept descriptions. It is worth noting that GCIs generalize primitive concept definitions \((A \sqsubseteq C)\) and full concept definitions \((A \equiv C)\), whereas RIs generalize transitivity \((r \circ r \sqsubseteq r)\) and reflexivity \((\epsilon \sqsubseteq r, \text{where} \epsilon \text{stands for composition of zero roles})\).

The semantics of \(\mathcal{EL}^+\) is defined in terms of interpretations \(I = (\Delta^I, \cdot^I)\), where the domain \(\Delta^I\) is a non-empty set of individuals, and the interpretation function \(\cdot^I\) maps each concept name \(A \in \mathcal{CN}\) to a subset \(A^I\) of \(\Delta^I\) and each role name \(r \in \mathcal{RN}\) to a binary relation \(r^I\) on \(\Delta^I\). The extension of \(\cdot^I\) to arbitrary concept descriptions is inductively defined, as shown in the semantics column of