Journal of Biomedical Informatics 80 (2018) 106–119

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

An efficient, large-scale, non-lattice-detection algorithm for exhaustive structural auditing of biomedical ontologies

Guo-Qiang Zhang[a,b,c,*], Guangming Xing[d], Licong Cui[a,b]

[a] Department of Computer Science, University of Kentucky, Lexington, KY, USA
[b] Institute for Biomedical Informatics, University of Kentucky, Lexington, KY, USA
[c] Department of Internal Medicine, University of Kentucky, Lexington, KY, USA
[d] Department of Computer Science, Western Kentucky University, Bowling Green, KY, USA

ARTICLE INFO

Keywords:
Biomedical ontology
Partial order
Graph-theoretic algorithm
SNOMED CT
Lattice vs non-lattice
Quality assurance

ABSTRACT

One of the basic challenges in developing structural methods for systematic audition on the quality of biomedical ontologies is the computational cost usually involved in exhaustive sub-graph analysis. We introduce ANT-LCA, a new algorithm for computing all non-trivial lowest common ancestors (LCA) of each pair of concepts in the hierarchical order induced by an ontology. The computation of LCA is a fundamental step for non-lattice approach for ontology quality assurance. Distinct from existing approaches, ANT-LCA only computes LCAs for non-trivial pairs, those having at least one common ancestor. To skip all trivial pairs that may be of no practical interest, ANT-LCA employs a simple but innovative algorithmic strategy combining topological order and dynamic programming to keep track of non-trivial pairs. We provide correctness proofs and demonstrate a substantial reduction in computational time for two largest biomedical ontologies: SNOMED CT and Gene Ontology (GO). ANT-LCA achieved an average computation time of 30 and 3 sec per version for SNOMED CT and GO, respectively, about 2 orders of magnitude faster than the best known approaches. Our algorithm overcomes a fundamental computational barrier in sub-graph based structural analysis of large ontological systems. It enables the implementation of a new breed of structural auditing methods that not only identifies potential problematic areas, but also automatically suggests changes to fix the issues. Such structural auditing methods can lead to more effective tools supporting ontology quality assurance work.

1. Introduction

In graph-theoretic representation of ontologies in biomedicine such as SNOMED CT [1], ontological concepts correspond to graph nodes, and is-a relations correspond to edges of the graph. When rendering the is-a relations as a graph, the Hasse diagram convention orients more general concepts above (or higher than) more specific concepts.

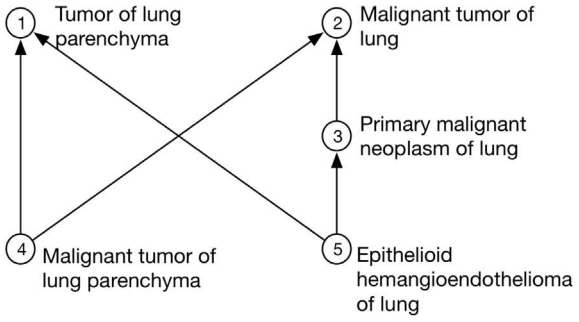
One of the desirable properties of the resulting graph structure is that the subsumption relationship (is-a hierarchy) should form a lattice [2]. There are in general two types of lattice-based approaches to ontology quality assurance. One involves the direct application of Formal Concept Analysis (FCA [3]), mostly for auditing semantic completeness or missing concepts [4]. The second involves the extraction of lattice-violating fragments [5,6], or *non-lattice fragments*, which represent violations of the FCA principle that systematic engineering approaches for constructing concept hierarchies always result in order structures that are lattices in the sense of lattice theory [3]. This non-lattice approach for ontology quality assurance involves the extraction of graph substructures (i.e. sub-orders) that violate the *lattice property*, which requires that any two concept nodes have at most one minimal shared (common) ancestor and at most one maximal shared descendant.

As illustrated recently in [7], the use of the non-lattice approach for improving the quality of an ontology consists of the following general steps:

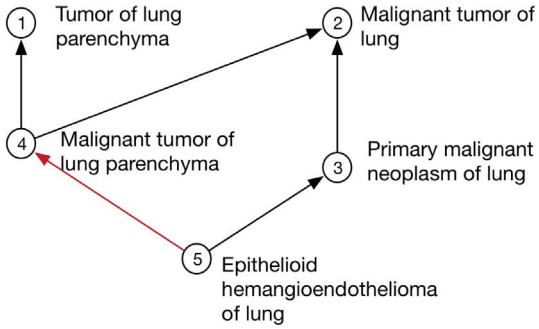
1. Identify node-pairs that violate the lattice property (i.e. non-lattice pairs) and extract the associated non-lattice fragments.
2. Detect ontological defects such as miss-aligned is-a relations or missing concepts in the extracted non-lattice fragments, often leveraging additional or external information.
3. Formulate and generate change suggestions automatically and present the suggestions in a usable format.
4. Perform reviews of the suggested changes and accept or reject such suggestions by a qualified ontology engineer or ontology editor, and incorporate the accepted changes into the next release.

* Corresponding author at: University of Kentucky, 230 Multidisciplinary Science Building, 725 Rose Street, Lexington, KY 40536-0082, USA.
E-mail address: gq.zhang@uky.edu (G.-Q. Zhang).

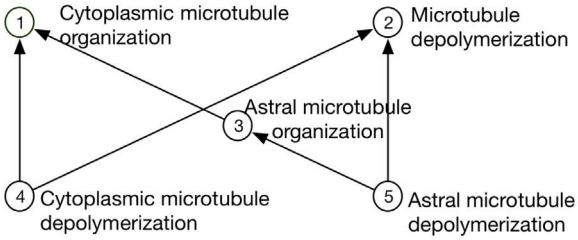
https://doi.org/10.1016/j.jbi.2018.03.004
Received 15 October 2017; Received in revised form 10 March 2018; Accepted 12 March 2018
Available online 13 March 2018
1532-0464/



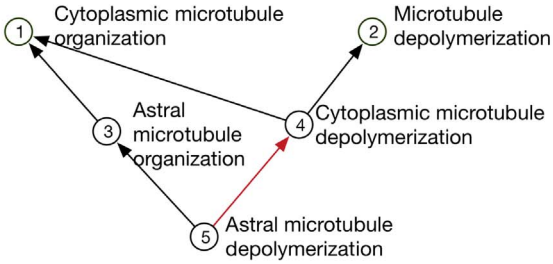
(1A) Non-lattice fragment in SNOMED CT



(1B) Lattice subgraph after missing IS-A relation is added



(2A) Non-lattice fragment in Gene Ontology



(2B) Lattice subgraph after missing IS-A relation is added

Fig. 1. An example (1A) of non-lattice fragment of size 5 in SNOMED CT, as well as the resulting lattice subgraph (2B) after a missing IS-A relation is added (red link). Similarly, (2A) is a non-lattice fragment of size 5 in GO and (2B) is the correction. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The non-lattice approach is unique in that while most ontology quality assurance techniques [8] merely identify potential errors, this approach can not only identify previously undiscovered errors confirmed by domain experts, but also suggest appropriate remediation (i.e., "auto-suggestion") [7,9]. For example, Fig. 1 (top), extracted from the September 2017 release of SNOMED CT (US edition), contains a substructure (1A) of is-a relations on the left, involving 5 concepts. This is a non-lattice fragment, because the concept nodes labeled 1 and 2 have two maximal shared descendants: concept nodes labeled 4 and 5. With a combination of structural and lexical information represented in this fragment, one can infer that "Epithelioid hemangioendothelioma of lung" is-a "Malignant tumor of lung parenchyma." Remarkably, adding such a missing edge (in red color) also makes the resulting subgraph (1B) conforming to the lattice property: concept nodes labeled 1 and 2 now have a unique maximal shared descendant: concept nodes labeled 4 (since concept 5 is no longer "maximal"). Similarly, the lower part of Fig. 1 shows a non-lattice fragment (2A) in the Gene Ontology (GO) on the left, and the corrected structure (2B) on the right.

Both the FCA- and the non-lattice-based approaches incur computational costs that sometimes make exhaustive analyses prohibitive. For example, in Jiang and Chute's work [4], only 10% of SNOMED CT subhierarchies were sampled in order to assess semantic completeness. Three months of sequential computation [5] or three hours of 25-node parallel processing [6] were required to detect non-lattice pairs for each version of SNOMED CT. The detection of non-lattice pairs is a fundamental step for non-lattice-based approach for ontology quality assurance. The non-lattice pairs serve as seeds for systematic generation of non-lattice fragments, but including all nodes in-between the seed nodes and the maximal shared descendants. Therefore, more efficient algorithms for detection of non-lattice pairs is highly desirable.

This paper introduces ANT-LCA, a new algorithm for computing all non-trivial lowest common ancestors (LCA) of each pair of concepts in the graph induced by an ontological system. Here the lowest common ancestors in the context of a graph are exactly the maximal shared descendants in the context of an ontology. In the remainder of the paper, we discuss algorithms in graph-theoretic and order-theoretic terms. But whenever working with specific ontological examples, we switch back to maximal shared descendants. Distinct from existing approaches, ANT-LCA only computes LCAs for non-trivial pairs, those having at least one common ancestor. To skip all trivial pairs that may be of no practical interest, ANT-LCA employs a simple but innovative algorithmic strategy combining topological order and dynamic programming [10] to keep track of non-trivial pairs.

We provide correctness proofs and demonstrate about 2-orders of magnitude reduction, compared with the best parallel algorithms known to date, in computational time for two of the largest biomedical ontologies: SNOMED CT and Gene Ontology (GO). ANT-LCA achieved an average computation time of 30 and 3 sec per version for SNOMED CT and GO, respectively, confirming our complexity analysis with a time-bound involving *pairability-degree* (i.e. the constant in big-O analysis of time-complexity) as a quadratic factor. ANT-LCA overcomes a fundamental computational barrier in subgraph analysis of ontological structures. It enables the implementation of a new breed of structural auditing methods that can not only identifies potential problematic areas, but also automatically suggests specific changes that are needed to fix the quality issues.

2. Background

2.1. LCA on directed acyclic graphs

In a directed acyclic graph (DAG), a common ancestor (CA) of a pair of nodes u, v is a node w that is a shared ancestor of u, v. A lowest CA is a node w such that no other shared ancestor is closer (nearer) to u, v than w. A pair of nodes u, v is trivial if they do not have a shared ancestor, or one of them is the ancestor of the other. Conversely, non-trivial pairs are those having at least one lowest common ancestor other than the nodes already in the pair. Given a subset of nodes X in a DAG, we denote the set of lowest common ancestors of X as $\mathrm{lca}(X)$, and common ancestors of X as $\mathrm{ca}(X)$, respectively. When X is a two-element set $\{a, b\}$ with two or more lowest common ancestors, it is called a *non-lattice pair*.

A pair of nodes (x, y) is called *pairable* if $\mathrm{lca}\{x, y\} \neq \varnothing$, $\mathrm{lca}\{x, y\} \neq \{x\}$, as well as $\mathrm{lca}\{x, y\} \neq \{y\}$. Intuitively, x, y is pairable if they share at least one non-trivial common ancestor. In this case we also