Towards fast Atomic Decomposition using Axiom Dependency Hypergraphs*

Francisco Martín-Recuerda[1] and Dirk Walther[2]

[1] Universidad Politécnica de Madrid, Spain
fmartinrecuerda@fi.upm.es
[2] TU Dresden, Germany
Center for Advancing Electronics Dresden
dirk.walther@tu-dresden.de

Abstract. Atomic decomposition of ontologies has been suggested as a tool to understand the modular structure of ontologies. It consists of a polynomial size representation of potentially exponentially many modules of an ontology. Tractable algorithms for computing the atomic decomposition for locality-based modules have been introduced, albeit leaving room for improvement in terms of running time. In this paper, we consider ontologies formulated in OWL-EL. We introduce a notion of an axiom dependency hypergraph for an ontology, which represents how axioms are included in locality-based modules. We use a standard algorithm from graph theory to compute strongly connected components of the axiom dependency hypergraph and show that such components correspond to atoms of the ontology. An empirical evaluation of the algorithm on large fragments of biomedical ontologies confirms a significant improvement in running time.

1 Introduction

The atomic decomposition of an ontology provides a compact representation of all possible modules that can be extracted from a given ontology. It represents a significant contribution to a better understanding of the internal structure of the ontology.

In this paper, we introduce a novel representation model of OWL-EL ontologies based on directed hypergraphs that explicitly represent information for syntactic locality and atomic decomposition. We call this model *Axiom Dependency Hypergraphs* (ADHs). A directed hypergraph is a generalisation of a directed graph in which edges (in this case hyperedges) can connect two nonempty disjoint sets of nodes (or vertices). The nodes in such hypergraphs represent the axioms of an ontology, and hyperedges indicate subset relations between terms

* We thank the reviewers of the workshop WoMO 2013 for their comments. The first author acknowledges the support of the EU project SEALS (FP7-ICT-238975), and the second author the support of the German Research Foundation (DFG) within the Cluster of Excellence 'Center for Advancing Electronics Dresden'.

in axioms. By using standard hyperpath search algorithms it is possible to efficiently extract locality-based modules (in linear time) and to compute the atomic decomposition of the ontology.

The use of hypergraph-based models for locality-based module extraction is not new [8] and it has been recently further extended [6]. However, this is the first time that it has been applied for improving existent algorithms for atomic decomposition of OWL ontologies. We compare a prototypical implementation of our hypergraph model against the fastest implementation of atomic decomposition that we are aware of [10]. Our experiments confirm a significant improvement in running time for (certain fragments of) biomedical ontologies.

The paper is organised as follows. In Section 2, we review the main notions on syntactic $\bot$-locality, atomic decomposition, and hypergraphs. In Section 3, we introduce axiom dependency hypergraphs and discuss their size, and evaluate the performance of atomic decomposition based on these graphs in Section 4. We conclude this paper in a final section.

2 Preliminaries

In this paper, we consider ontologies formulated in the description logic $\mathcal{EL}$, which allows for conjunction and existential restrictions. Large parts of, e.g., biomedical ontologies are expressed in $\mathcal{EL}$. For notions on description logics, we refer to [3] and for $\mathcal{EL}$ to [2]. Moreover, we consider locality-based modules. The notion of locality refers to axioms, i.e., axioms are either local or non-local, and it depends on a signature. A module of an ontology consists of the non-local axioms wrt. a signature. There are two flavours the locality notion comes in: semantic and syntactic locality. Intuitively, an axiom is semantically local wrt. a signature Σ if it does not say anything about the terms in Σ.[1] Checking whether an axiom is semantically local can be computationally expensive as it requires reasoning.[2] The notion of syntactic locality was introduced to allow for efficient computation. Checking for syntactic locality involves checking that an axiom is of a certain form, no reasoning is needed. On the downside, syntactic locality is merely an approximation of semantic locality: all syntactically local axioms are also semantically local, but not *vice versa*. Modules based on syntactic locality can be larger as they contain the modules based on semantic locality and possibly more axioms. We refer to [5] for a more extensive introduction to locality-based modularity.

The notion of $\bot$-locality depends on whether or not a given signature covers all symbols occurring on the LHS of a general concept inclusion, or occurring on the LHS or RHS of a general concept equation. The following proposition makes that precise.[3]

[1] For sets of axioms with this property the term 'safety' is used in the literature [5].

[2] Reasoning with OWL2 is 2-NExpTime-complete in the worst case, but also reasoning with a tractable logic such as $\mathcal{EL}$ can be seen as too difficult in some cases, depending on the size of the input and the application requirements.

[3] The notion of reachability-based modules uses a similar property; see Def. 37 in [8].

Proposition 1. *Let α be an $\mathcal{EL}$-axiom. Let Σ be a signature. Then:*

(i) if $\alpha = (C \sqsubseteq D)$, then α is not syntactic $\bot$-local wrt. Σ iff $\mathsf{sig}(\mathsf{LHS}(\alpha)) \subseteq \Sigma$;
(ii) if $\alpha = (C \equiv D)$, then α is not syntactic $\bot$-local wrt. Σ iff $\mathsf{sig}(\mathsf{LHS}(\alpha)) \subseteq \Sigma$ or $\mathsf{sig}(\mathsf{RHS}(\alpha)) \subseteq \Sigma$. ⊣

The following example illustrates the notion of $\bot$-locality of axioms.

Example 1. Let α and β be the axioms:[4]

$$\alpha := A_1 \sqcap \exists r_1.(A_2 \sqcap \exists r_2.A_3) \sqsubseteq \exists r_3.A_4$$
$$\beta := A_1 \equiv A_2 \sqcap \exists r_1.A_3$$

Axiom α is syntactic $\bot$-local wrt. Σ if any of the symbols occurring in $\mathsf{LHS}(\alpha)$ is not contained in Σ, i.e., if α satisfies any of the following conditions:

- $A_i \notin \Sigma$, for some $i \in \{1, 2, 3\}$; or
- $r_1 \notin \Sigma$ or $r_2 \notin \Sigma$.

Axiom β is syntactic $\bot$-local wrt. Σ if there are two symbols, one occurring in $\mathsf{LHS}(\beta)$ and one in $\mathsf{RHS}(\beta)$, that are not contained in Σ, i.e., if β satisfies any of the following conditions:

- $A_1 \notin \Sigma$ and $A_i \notin \Sigma$, for some $i \in \{2, 3\}$; or
- $A_1 \notin \Sigma$ and $r_1 \notin \Sigma$. ◁

2.1 Extracting modules based on locality

This is the module extraction algorithm from [5], where x stands for any of the semantic and syntactic locality notions, i.e. $x \in \{\emptyset, \Delta, \bot, \top\}$. The algorithm runs in time quadratic in the size of the ontology and signature.

---

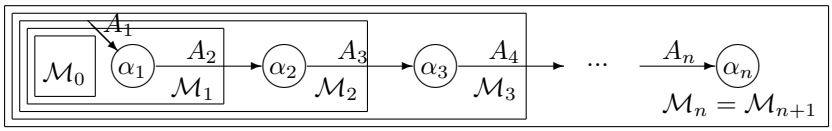
function $\mathsf{Mod}^x_{\mathcal{O}}(\Sigma)$ returns x-local module of $\mathcal{O}$ wrt. Σ
1: $m := 0$
2: $\mathcal{M}_m := \emptyset$
3: do
4: $\quad m := m + 1$
5: $\quad \mathcal{M}_m := \{\alpha \in \mathcal{O} \mid \alpha$ is not x-local wrt. $\Sigma \cup \mathsf{sig}(\mathcal{M}_{m-1})\}$
6: until $\mathcal{M}_m = \mathcal{M}_{m-1}$
7: return $\mathcal{M}_m$

The algorithm takes an ontology $\mathcal{O}$ and a signature Σ as input and returns a subset $\mathcal{M}$ of $\mathcal{O}$. The computed set $\mathcal{M}$ consists of axioms that are not x-local wrt. $\Sigma \cup \mathsf{sig}(\mathcal{M})$, or equivalently, $\mathcal{O} \setminus \mathcal{M}$ consists of axioms that are local wrt. $\Sigma \cup \mathsf{sig}(\mathcal{M})$. The algorithm produces a finite sequence $\mathcal{M}_0, \ldots, \mathcal{M}_n$, $n \geq 0$, of subsets of the ontology $\mathcal{O}$, where $\mathcal{M}_n$ is the $\bot$-local module of $\mathcal{O}$ wrt.

[4] These axioms are of the forms that occur frequently in the biomedical ontology Full-Galen.

$\Sigma \cup \mathsf{sig}(\mathcal{M}_n)$ with Σ being the initial signature. This sequence induces a relation on $\mathcal{O}$ representing the successive inclusion of the axioms into the module. Later we will represent this relation as hyperedges in an axiom dependency hypergraph. This is illustrated in the following example.

Example 2. The signature increases round-by-round due to axioms that are added to the module (line 5). In fact, the RHSs of the axioms introduce new symbols to the signature. Let $\alpha_i = A_i \sqsubseteq A_{i+1}$, for $i \in \{1, ..., n\}$. Let $\mathcal{O} = \{\alpha_1, ..., \alpha_n\}$ and $\Sigma = \{A_1\}$. We run the algorithm on the input $\mathcal{O}$ and Σ. In the first round of the do-until loop (lines 3-6), the axiom α_1 is the only axiom in $\mathcal{O}$ that is not $\bot$-local wrt. $\Sigma \cup \mathsf{sig}(\mathcal{M}_0)$ (line 5), where $\mathcal{M}_0 = \emptyset$ (line 2). So α_1 is added to $\mathcal{M}_1$. In the second round, $\bot$-locality is checked for the extended signature $\Sigma \cup \mathsf{sig}(\mathcal{M}_1) = \{A_1, A_2\}$. Axioms α_1 and α_2 are now non-$\bot$ local wrt. $\Sigma \cup \mathsf{sig}(\mathcal{M}_1)$ and both are added to $\mathcal{M}_2$. The algorithm proceeds this way until all axioms of $\mathcal{O}$ are added to $\mathcal{M}_n$. As no more axioms are added the algorithm exits the do-until loop (line 6) and returns $\mathcal{M}_{n+1} = \mathcal{O}$ as the $\bot$-local module of $\mathcal{O}$ wrt. Σ (line 7). The following picture shows the module extraction process as a graph $G_{\mathsf{mod}^\bot(\mathcal{O},\Sigma)} = (V, E)$, where $V = \{\alpha_1, ..., \alpha_n\}$ and $E = \{(\alpha_i, A_{i+1}, \alpha_{i+1}) \mid i \in \{1, ..., n-1\}\}$, and the sequence $\mathcal{M}_0, ..., \mathcal{M}_n, \mathcal{M}_{n+1}$ produced by the module extraction algorithm. Additionally we indicate with a tilted arrow labelled with A_1 to node α_1 that the concept name A_1 is provided by the initial signature. In this case, α_1 is the first axiom to be included by the module extraction algorithm.



◁

2.2 Atomic decomposition

Atoms represent sets of highly related axioms in the sense that they always co-occur in modules. A set $\mathfrak{a}$ of axioms from an ontology $\mathcal{O}$ is an *atom of* $\mathcal{O}$ if for every module $\mathcal{M}$ of $\mathcal{O}$, it holds that $\mathfrak{a} \subseteq \mathcal{M}$ or $\mathfrak{a} \cap \mathcal{M} = \emptyset$. We denote with $\mathsf{Atoms}_\mathcal{O}$ the set of all atoms of $\mathcal{O}$. The atoms of an ontology partition the set of its axioms (i.e., each axiom occurs in exactly one atom). A dependency relation between pairs of atoms can be established: an atom $\mathfrak{a}_2$ *depends on* an atom $\mathfrak{a}_1$ in an ontology $\mathcal{O}$ (written $\mathfrak{a}_1 \succcurlyeq_\mathcal{O} \mathfrak{a}_2$) if $\mathfrak{a}_2$ occurs in every module of $\mathcal{O}$ containing $\mathfrak{a}_1$. For a given ontology, the pair $\langle \mathsf{Atoms}_\mathcal{O}, \succcurlyeq_\mathcal{O} \rangle$ consisting of the set of atoms together with the dependency relation between them is called *Atomic Decomposition*.[5] It provides a compact representation of all modules of an ontology as every module is composed of the axioms of certain atoms. Therefore, the atomic decomposition contributes to a better understanding of the internal structure of ontologies with possible implications in ontology engineering and reasoner design. The representation of the modules in terms of atoms and their

[5] Here we use atomic decomposition for $\bot$-local modules denoted as $\langle \mathsf{Atoms}^\bot_\mathcal{O}, \succcurlyeq^\bot_\mathcal{O} \rangle$.

interdependencies is of linear size and it can be computed in polynomial time, whereas there are exponentially many possible modules of an ontology. However, not *all* modules of an ontology can be computed from its atomic decomposition. The atomic decomposition was first introduced in [11].

2.3 Directed hypergraphs

A *directed hypergraph* [4] is a tuple $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a non-empty set of *nodes* (vertices), and $\mathcal{E}$ is a set of *hyperedges* (*hyperarcs*). A *hyperedge* e is also defined as a pair $(T(e), H(e))$, where $T(e)$ and $H(e)$ are nonempty disjoint subsets of $\mathcal{V}$. $H(e)$ ($T(e)$) is known as the *head* (*tail*) and represents a set of nodes where the hyperedge ends (starts). For each node $v \in H(e)$ ($v \in T(e)$), e is an *incoming* (*outgoing*) hyperedge to v. A *hyperpath* from node v_1 to node v_n is a sequence of the form $P(v_1, v_n) = v_1 e_1 v_2 e_2 \cdots e_{n-1} v_n$ such that $v_i \in T(e_i)$ and $v_{i+1} \in H(e_i)$ for every $i \in \{1, ..., n-1\}$. In addition, if node $v_n \in T(e_1)$, the hyperpath $P(v_1, v_n)$ is a *hypercycle*.

A node v_2 is *B-connected*[6] (or forward reachable[7]) from v_1 if (i) $v_2 = v_1$ (v_2 is B-connected from itself), or (ii) there is a hyperedge e such that $v_2 \in H(e)$ and all the elements of $T(e)$ are B-connected from v_1. A *B-hyperpath* from node v_1 to node v_2 in a hypergraph $\mathcal{H}$ is a minimal (in terms of hyperedges and nodes) subhypergraph of $\mathcal{H}$ such that v_2 is B-connected to v_1.

In a directed hypergraph $\mathcal{H}$, two nodes v_1 and v_2 are *strongly B-connected* if v_2 is B-connected to v_1 and *vice versa*. In other words, both nodes, v_1 and v_2, are *mutually reachable*. A *strongly B-connected component (SCC)* is a set of nodes from $\mathcal{H}$ which are all are mutually reachable [1].[8] Note that two nodes that are mutually reachable in a hypergraph belong to the same hypercycle.

In the case of directed graph, it is possible to calculate all the strongly connected components in linear time [9, 7]. Unfortunately, none of these linear time algorithms can be easily adapted to directed hypergraphs due to the more complicated notion of reachability in hypergraphs [1]. For instance, while in a directed graph the last node of a path can be reached from any other node in the path, this may not be the case in a hyperpath [1].

3 Axiom dependency hypergraphs

Axiom dependency hypergraphs are a representation model for ontologies based on directed hypergraphs that explicitly represent information for locality and atomic decomposition. The nodes in such graphs represent the axioms of an ontology, and hyperedges indicate subset relationships between terms in axioms

[6] There is a similar notion called *F-connectivity* [4].

[7] We walk on a hypergraph from the tails to the heads of the hyperedges. If all nodes in the tail of a hyperedge have been reached it is possible to reach all the nodes of the head of the hyperedge.

[8] Notice that an SCC can be a singleton set as the reachability relation is reflexive, i.e., any axiom is mutually reachable from itself.

(cf. Proposition 1). By using standard hyperpath search algorithms it is possible to efficiently extract locality based modules (in linear time) and to compute the atomic decomposition of the ontology.

The $\bot$-*locality signatures* $\bot\text{-}\mathsf{sig}(\alpha)$ of an axiom α is the set of signatures

$$\bot\text{-}\mathsf{sig}(\alpha) = \begin{cases} \{\mathsf{sig}(\mathsf{LHS}(\alpha))\} & \text{if } \alpha \text{ is of the form } C \sqsubseteq D; \\ \{\mathsf{sig}(\mathsf{LHS}(\alpha)), \mathsf{sig}(\mathsf{RHS}(\alpha))\} & \text{if } \alpha \text{ is of the form } C \equiv D. \end{cases}$$

Intuitively, $\bot\text{-}\mathsf{sig}(\cdot)$ is a function assigning to an axiom α the signatures $\bot\text{-}\mathsf{sig}(\alpha)$ that are relevant for deciding the $\bot$-locality status of α. More precisely, we have that α is not $\bot$-local wrt. S, for every $S \in \bot\text{-}\mathsf{sig}(\alpha)$.

Axiom dependency hypergraphs for $\mathcal{EL}$-ontologies are defined as follows.

Definition 1 (Axiom dependency hypergraph for $\bot$-locality). *Let $\mathcal{O}$ be an $\mathcal{EL}$-ontology. The* $\bot$-locality axiom dependency hypergraph (ADH) $\mathcal{H}^{\bot}_{\mathcal{O}}$ for $\mathcal{O}$ *is defined as* $\mathcal{H}^{\bot}_{\mathcal{O}} = (\mathcal{V}, \mathcal{E})$*, where*

- $\mathcal{V} = \mathcal{O}$*; and*
- $e = (T(e), H(e)) \in \mathcal{E}$ *iff the following holds:*
  - *(i)* $H(e) = \{\beta\}$*, for some* $\beta \in \mathcal{V}$*, and*
  - *(ii)* $T(e) \subseteq \mathcal{V} \setminus \{\beta\}$ *such that* $|T(e)|$ *is minimal with the property* $S \subseteq \mathsf{sig}(T(e))$*, for some* $S \in \bot\text{-}\mathsf{sig}(\beta)$*.*

⊣

Note that the axiom dependency hypergraph for an $\mathcal{EL}$-ontology is defined for the notion of $\bot$-locality.[9] The nodes are the axioms of the ontology, and the hyperedges are associating axioms $\alpha_1, \ldots, \alpha_n$ with a single axiom β such that one of the $\bot$-locality signatures of β is contained in the signature of the axioms α_i. The minimality condition states that each of the axioms α_i is required and none of them is superfluous for covering some of the $\bot$-locality signatures of β.[10]

Example 3. Let $\mathcal{O} = \{\alpha_1, ..., \alpha_5\}$, where

$$\alpha_1 := A \sqsubseteq B \qquad \alpha_3 := E \sqsubseteq A \sqcap C \sqcap D \qquad \alpha_5 := X \sqsubseteq A$$
$$\alpha_2 := B \sqcap C \sqcap D \sqsubseteq E \qquad \alpha_4 := A \sqsubseteq X$$

The ADH $\mathcal{H}^{\bot}_{\mathcal{O}}$ contains the hyperedges $e_1 = (\{\alpha_1, \alpha_3\}, \{\alpha_2\})$, $e_2 = (\{\alpha_1\}, \{\alpha_4\})$, $e_3 = (\{\alpha_2\}, \{\alpha_3\})$, $e_4 = (\{\alpha_3\}, \{\alpha_1\})$, $e_5 = (\{\alpha_3\}, \{\alpha_4\})$, $e_6 = (\{\alpha_4\}, \{\alpha_1\})$, $e_7 = (\{\alpha_4\}, \{\alpha_5\})$, $e_8 = (\{\alpha_5\}, \{\alpha_1\})$ and $e_9 = (\{\alpha_5\}, \{\alpha_4\})$; see Example 4 for a visualisation of the graph. Now obtain $\mathcal{O}'$ by replacing α_2 with $\alpha'_2 := B \sqcap C \sqcap D \equiv E$. Then the ADH $\mathcal{H}^{\bot}_{\mathcal{O}'}$ contains one additional edge $e_{10} = (\{\alpha_3\}, \{\alpha_2\})$. ◁

B-connectivity (forward reachability) in axiom dependency hypergraphs can be used to specify $\bot$-local modules in the corresponding ontology. Denote with $\mathsf{Mod}^{\bot}_{\mathcal{O}}(\alpha)$ the $\bot$-local module of ontology $\mathcal{O}$ wrt. the signature of axiom α.

[9] Axiom dependency hypergraphs for the notion of $\top$-locality can be defined in a similar way.

[10] Note that the minimality condition in Def. 1 is to avoid superfluous hyperedges, but it is not required in terms of correctness.

Proposition 2. *Let $\mathcal{O}$ be an $\mathcal{EL}$-ontology and α an $\mathcal{EL}$-axiom. Then: $\mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha) = \{\beta \mid \alpha$ is B-connected to β in $\mathcal{H}^{\perp}_{\mathcal{O}}\}$.* ⊣

Reachability-based modules have been defined in other hypergraph respresentations of ontologies in [8]. This proposition can be shown similarly.

The set of atoms for an $\mathcal{EL}$-ontology $\mathcal{O}$ corresponds to the set of strongly connected components in the axiom dependency hypergraph for $\mathcal{O}$. Let $\mathsf{SCCs}(\mathcal{H}^{\perp}_{\mathcal{O}})$ be the set of strongly connected components of the hypergraph $\mathcal{H}^{\perp}_{\mathcal{O}}$.

Proposition 3. *Let $\mathcal{O}$ be an $\mathcal{EL}$-ontology. Then: $\mathsf{Atoms}^{\perp}_{\mathcal{O}} = \mathsf{SCCs}(\mathcal{H}^{\perp}_{\mathcal{O}})$.* ⊣

The proposition can readily be seen as follows. As a corollary of Proposition 2, mutually reachable axioms are contained in the same modules, which is equivalent to the axioms forming an atom of the ontology. The nodes in a strongly connected component are all mutually reachable. Thus, both notions are equivalent.

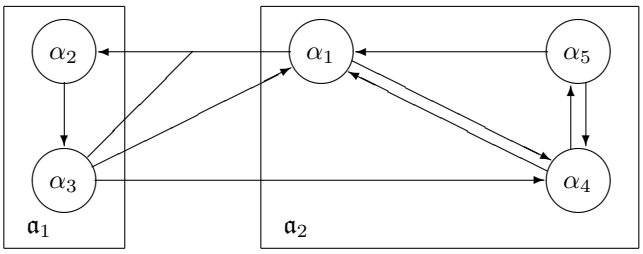
The dependencies between atoms for an $\mathcal{EL}$-ontology $\mathcal{O}$ (as given by the relation $\succcurlyeq^{\perp}_{\mathcal{O}}$) correspond to a certain binary relation over the set of strongly connected components of $\mathcal{H}^{\perp}_{\mathcal{O}}$, which is defined as follows. For $S_1, S_2 \in \mathsf{SCCs}(\mathcal{H}^{\perp}_{\mathcal{O}})$, we say that S_2 *depends* on S_1 (written $S_1 \rightarrow^{\perp}_{\mathcal{O}} S_2$) if there is an hyperedge $e = (T(e), H(e))$ in $\mathcal{H}^{\perp}_{\mathcal{O}}$ such that $T(e) \subseteq S_1$ and $H(e) \subseteq S_2$. Let $\rightarrow^{\perp}_{\mathcal{O}}{}^{*}$ be the reflexive, transitive closure of $\rightarrow^{\perp}_{\mathcal{O}}$.

Proposition 4. *Let $\mathcal{O}$ be an $\mathcal{EL}$-ontology. Let $\mathfrak{a}_1, \mathfrak{a}_2 \in \mathsf{Atoms}^{\perp}_{\mathcal{O}}$ and $S_1, S_2 \in \mathsf{SCCs}(\mathcal{H}^{\perp}_{\mathcal{O}})$ such that $\mathfrak{a}_1 = S_1$ and $\mathfrak{a}_2 = S_2$. Then: $\mathfrak{a}_1 \succcurlyeq^{\perp}_{\mathcal{O}} \mathfrak{a}_2$ iff $S_1 \rightarrow^{\perp}_{\mathcal{O}}{}^{*} S_2$.* ⊣

Example 4. Consider again the ontology $\mathcal{O}$ and its axiom dependency hypergraph $\mathcal{H}^{\perp}_{\mathcal{O}}$ from Example 3. We obtain the following $\perp$-local modules for the axioms:

$$\mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha_1) = \{\alpha_1, \alpha_4, \alpha_5\} \qquad \mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha_4) = \{\alpha_1, \alpha_4, \alpha_5\}$$
$$\mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha_2) = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\} \qquad \mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha_5) = \{\alpha_1, \alpha_4, \alpha_5\}$$
$$\mathsf{Mod}^{\perp}_{\mathcal{O}}(\alpha_3) = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$$

The resulting atoms in $\mathsf{Atoms}_{\mathcal{O}}$ are $\mathfrak{a}_1 = \{\alpha_2, \alpha_3\}$ and $\mathfrak{a}_2 = \{\alpha_1, \alpha_4, \alpha_5\}$, where $\mathfrak{a}_1 \succcurlyeq \mathfrak{a}_2$, i.e., $\mathfrak{a}_2$ depends on $\mathfrak{a}_1$. The ADH $\mathcal{H}^{\perp}_{\mathcal{O}}$ and the atoms of $\mathcal{O}$ can be depicted as follows:



Consider the strongly connected components of $\mathcal{H}^{\perp}_{\mathcal{O}}$. Axiom α_1 is B-connected with the axioms α_4 and α_5, α_4 is B-connected with α_1 and α_5, and α_5 is B-connected with α_1 and α_4. Axiom α_2 is B-connected with α_3 and *vice versa*.

Axioms α_2, α_3 are each B-connected with α_1, α_4 and α_5, but not *vice versa*. Hence, $\{\alpha_1, \alpha_2, \alpha_4\}$ and $\{\alpha_2, \alpha_3\}$ are the strongly connected components of $\mathcal{H}^{\perp}_{\mathcal{O}}$. Moreover, we say that the former component *depends* on the latter as any two axioms contained in them are unilaterally and not mutually B-connected. Note that the atoms $\mathfrak{a}_1$ and $\mathfrak{a}_2$ of $\mathcal{O}$ and their dependency coincide with the strongly connected components of $\mathcal{H}^{\perp}_{\mathcal{O}}$. $\lhd$

For ontologies consisting of axioms of the form $A \sqsubseteq C$, where A is a concept name, the locality dependencies between axioms can be represented in directed graphs (i.e., hypergraphs in which the tail of any hyperedge contains only one axiom). For such ontologies $\mathcal{O}$, the definition of the axiom dependency hypergraph $\mathcal{H}^{\perp}_{\mathcal{O}} = (\mathcal{V}, \mathcal{E})$ can be simplified (cf. Def. 1). The condition for a hyperedge $e = (\{\alpha\}, \{\beta\})$, for $\alpha, \beta \in \mathcal{V}$, to be contained in $\mathcal{E}$ is now:

$$e = (\{\alpha\}, \{\beta\}) \in \mathcal{E} \;\text{ iff }\; \mathsf{sig}(\mathsf{LHS}(\beta)) \subseteq \mathsf{sig}(\alpha)$$

The advantage of this representation is that we can directly use a standard linear time algorithm for calculating strongly connected components of directed graphs [9, 7]. We notice the majority of axioms in biomedical ontologies (that we have considered in the evaluation part of this paper) are axioms of the form $A \sqsubseteq C$. For general ontologies, we can apply a three-phase approach similar to the one suggested in [1] for hypergraphs: first, we compute the strongly connected components for axioms of the form $A \sqsubseteq C$ using a linear-time algorithm; second, we collapse the strongly connected components into single nodes; and, finally, we compute the strongly connected components of the revised axiom dependency hypergraph using the notion of mutual reachability (which can be computed in at most quadratic time [1]).

3.1 Size of axiom dependency hypergraphs (worst-case)

We observe that axiom dependency hypergraphs may be of exponential size.

Proposition 5. *There exists ontologies $\mathcal{O}$ such that the axiom dependency hypergraph $\mathcal{H}_{\mathcal{O}}$ contains exponentially many hyperedges in the number of axioms of $\mathcal{O}$.* $\dashv$

We show the proposition with the following example.

Example 5. Let n, k be two natural numbers with $n > 0$ and $k > 0$. Ontology $\mathcal{O}_{n,k}$ is defined as:

$$\mathcal{O}_{n,k} = \{\alpha := X_1 \sqcap \ldots \sqcap X_n \sqsubseteq Y\} \cup \{\alpha^i_j := A^i_j \sqsubseteq X_i \mid i \in \{1, ..., n\}, j \in \{1, ..., k\}\}$$

Axiom α is the only axiom in $\mathcal{O}_{n,k}$ with a complex LHS, which in turn is a conjunction of n many concept names $X_1, ..., X_n$. The axioms α^i_j state that each concept name X_i subsumes k many concept names $A^i_1, ..., A^i_k$. The corresponding axiom dependency graph $\mathcal{H}_{\mathcal{O}_{n,k}}$ is the the tuple $\mathcal{H}_{\mathcal{O}_{n,k}} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{O}_{n,k}$ and

$$\mathcal{E} = \{(\{\beta_1, ..., \beta_n\}, \{\alpha\}) \mid \beta_i \in \{\alpha^i_1, ..., \alpha^i_k\}, i \in \{1, ..., n\}\}.$$

It can readily be seen that $\mathcal{H}_{\mathcal{O}_{n,k}}$ contains k^n many hyperedges, where n, k are each bound by the number of axioms in $\mathcal{O}_{n,k}$. $\lhd$

The axiom dependency hypergraph serves as a tool to determine the atoms of an ontology via calculating the strongly connected components of the graph. Example 5 illustrates a problem with the size of the axiom dependency hypergraphs as defined in Def. 1. We note, however, that no hyperedge in $\mathcal{H}_{\mathcal{O}_{n,k}}$ is needed for the purpose of determining the atoms for $\mathcal{O}_{n,k}$. For every axiom β in $\mathcal{O}_{n,k}$, the $\bot$-local module $\mathsf{Mod}^{\bot}_{\mathcal{O}}(\beta)$ is a singleton set consisting of β itself. Consequently, the atoms of $\mathcal{O}_{n,k}$ are singleton sets as well as are the strongly connected components of $\mathcal{H}_{\mathcal{O}_{n,k}}$. Note that we obtain the same strongly connected components after removing every hyperedge from $\mathcal{H}_{\mathcal{O}_{n,k}}$. This observation suggests a possible solution to avoid the exponential blow-up problem. The idea is to build the hypergraph using only the "necessary" hyperedges that are required for the computation of the strongly connected components which in turn correspond to atoms. We hypothesize that the hyperedges needed are the ones that preserve the mutual reachability relation between the axioms of the ontology. This means that no more incoming hyperedges are needed per axiom than the total number of axioms in the ontology. We leave a formal treatment of this idea for future work.

3.2 Size of axiom dependency hypergraphs for real ontologies

Primitive concept inclusions, i.e. axioms of the form $A \sqsubseteq C$, where A is a concept name and C a possibly complex concept, are of prime importance for biomedical ontologies. For instance, the majority of axioms in the biomedical ontologies from the Bioportal in the following table are primitive concept inclusions.[11]

Ontology	Signature size	#axioms $A \sqsubseteq C$	percentage of all axioms in ontology	#axioms $C \equiv D$	#role axioms
CPO (12/2011)	136 090	306 111	80.61%	73 461	96
FMA-lite	75 168	119 558	99.99%	0	3
Full-Galen (v1.1)	24 088	25 563	67.81%	9 968	2 165
GO v1.1 (1986)	34 220	61 990	99.99%	0	5
NCBI (v1.2)	847 796	847 755	100%	0	0
RH-MeSH (08/2012)	286 382	403 210	100%	0	0
Snomed CT (2010a)	291 207	227 698	78.18%	63 446	12

Most ontologies contain other axioms as well such as concept equations and role axioms.[12] The table shows the percentage of axioms of the form $A \sqsubseteq C$ to all axioms in each ontology. The ontologies NCBI and RH-Mesh are taxonomies (no logical operators); they consist entirely of axioms of the form $A \sqsubseteq B$, where A, B are concept names.

[11] `http://bioportal.bioontology.org`

[12] Other axiom types are not shown here, e.g., disjointness axioms in CPO.

Ontology $A \sqsubseteq C$-fragment	Signat. size	ADH #nodes	ADH #edges	ADH #SCCs
CPO (12/2011)	136 027	306 111	1 474 962	131 482
FMA-lite	75 141	119 558	1 021 863	54 649
Full-Galen (v1.1)	16 412	25 563	179 770	12 502
GO v1.1 (1986)	34 175	61 990	255 215	34 167
NCBI (v1.2)	847 760	847 755	1 695 474	847 755
RH-MeSH (08/2012)	286 380	403 210	1 435 631	286 263
Snomed CT (2010a)	240 021	227 698	556 474	227 698

The difference between the number of SCCs and the number of nodes in the axiom dependency hypergraph can be seen as a measure of cyclicity of the graph.

The following example illustrates the limitations of axiom dependency graphs. Axioms of the form $C \sqsubseteq D$ or $C \equiv D$ (i.e. axioms whose $\bot$-locality signatures contains more than one symbol) can cause many hyperedges to be included in the axiom dependency hypergraph.

Example 6. Consider this concept equation α from the ontology Full-Galen:

$$\begin{aligned}\mathsf{Incontinence} \equiv\ & \mathsf{Transport} \\ & \sqcap \exists\, \mathsf{actsOn.BodySubstance} \\ & \sqcap \exists\, \mathsf{hasIntentionality.}(\mathsf{Intentionality} \sqcap \exists\, \mathsf{hasAbsoluteState.involuntary}) \\ & \sqcap \exists\, \mathsf{hasUniqueAssociatedDisplacement.}(\mathsf{Displacement} \\ & \qquad \sqcap \exists\, \mathsf{isDisplacementTo.GRAILExteriorOfBody})\end{aligned}$$

The axiom dependency hypergraph $\mathcal{H}^{\bot}_{\text{Full-Galen}}$ contains up to 9.2×10^{12} many hyperedges of the form $e = (T(e), \{\alpha\})$, where $T(e)$ is a set of axioms containing the 12 signature symbols from $\bot\text{-}\mathsf{sig}(\alpha)$.[13] $\lhd$

As indicated in Section 3.1, the number of hyperedges may be reduced while preserving the strongly connected components. The idea is that no more incoming hyperedges are needed per axiom as there are axioms in the ontology. In the case of Full-Galen, we have an upper bound of 37 696 many incoming hyperedges per axiom. In particular, for axiom α from Example 6 we estimate up to 21 095 many hyperedges to preserve the mutual reachability relation involving α.

4 Evaluation

We have implemented a Java program that takes an $\mathcal{EL}$-ontology $\mathcal{O}$ (with axioms of the form $A \sqsubseteq C$) as an input and builds the corresponding axiom dependency hypergraph $\mathcal{H}^{\bot}_{\mathcal{O}}$. Then it computes the set $\mathsf{SCCs}(\mathcal{H}^{\bot}_{\mathcal{O}})$ of strongly connected components of $\mathcal{H}^{\bot}_{\mathcal{O}}$ and the dependencies between these components. We compare the running times of atomic decomposition using FaCT++ with the times needed by the hypergraph-based approach. The following table lists the results.[14]

[13] The number of hyperedges is merely an estimated upper bound (as it does not take into account the minimality condition in Def. 1).

[14] All experiments were performed on an Intel Xeon E5-2640 2.50 GHz with Java version 1.7.0_40 (command line parameters -Xss1G -Xms4G -Xmx8G). We used FaCT++, 64bit, Version 1.6.2 (19 February 2013) and the OWL API version 3.4.4.

Ontology fragment with axioms of the form $A \sqsubseteq C$	time FaCT++	time hypergraph-based approach					speedup
		load ont.	build ADH	comp. SCC	comp. deps.	total	
CPO (12/2011)	1 262.69 *s*	9.99 *s*	0.85 *s*	0.59 *s*	0.52 *s*	11.94 *s*	105.8
FMA-lite	19 991.92 *s*	9.50 *s*	0.49 *s*	6.43 *s*	0.20 *s*	16.62 *s*	1 202.9
Full-Galen (v1.1)	115.77 *s*	2.87 *s*	0.09 *s*	0.20 *s*	0.05 *s*	3.21 *s*	36.1
GO v1.1 (1986)	44.33 *s*	3.42 *s*	0.15 *s*	0.16 *s*	0.09 *s*	3.82 *s*	11.6
NCBI (v1.2)	49 227.86 *s*	73.72 *s*	1.42 *s*	0.87 *s*	1.19 *s*	77.22 *s*	637.5
RH-MeSH (08/2012)	6 921.62 *s*	15.24 *s*	1.31 *s*	0.63 *s*	0.66 *s*	17.84 *s*	388.0
Snomed CT (2010a)	4 538.65 *s*	14.47 *s*	0.48 *s*	0.31 *s*	0.32 *s*	15.58 *s*	291.3

The times for FaCT++ are the total times needed (2nd column), which includes loading the ontology, initialising the reasoner and computing the atoms (but not saving the atoms). For the hypergraph-based approach, we have listed the times needed for the OWL-API to load the ontology and to do some initialisation (3rd column), to build the axiom dependency hypergraph (4th column), to compute the strongly connected components of the graph (5th column), to compute the dependencies between the strongly connected components (6th column) and the total time needed (6th column). The last column shows for each ontology the speedup achieved for atomic decomposition using the hypergraph-based approach compared to FaCT++. On FMA-lite an over 1 000-fold speedup was realised.

FaCT++ improves upon the original algorithm for atomic decomposition [11] as described in [10]. However, the algorithm implemented in FaCT++ runs in time almost cubic (in particular, the computation of the transitive closure in Line 7 of Algorithm 4 in [10]). In contrast, the approach based on axiom dependency hypergraphs runs in time linear for the considered fragment of the ontologies.

5 Conclusion

We have suggested a hypergraph-based method for efficient atomic decomposition of $\mathcal{EL}$-ontologies consisting of primitive concept inclusions. We have introduced the notion of an axiom dependency hypergraph, in which, different to other hypergraph representations of ontologies, axioms are nodes that are connected in a way mimicking how axioms are included in a module by the locality-based module extraction algorithm.

Modularisation and atomic decomposition has been suggested as a means to understand the internal structure of ontologies. Axiom dependency hypergraphs are an explicit representation of these notions. A module can be extracted from the hypergraph by collecting the nodes reachable from the nodes that are determined by the input signature. Moreover, it is possible to directly apply standard off-the-shelf graph algorithms for computing the atoms of certain $\mathcal{EL}$-ontologies in linear time. For the atomic decomposition of FMA-lite, a staggering 1 000-fold speedup could be achieved compared to the reasoner FaCT++.

The disadvantage of the axiom dependency hypergraphs, as introduced in this paper, is their exponential size in the worst case for general concept inclusions. We have indicated that the blow-up in size can be avoided by omitting hyperedges while still preserving the mutual reachability relation between axioms. In this way, no more than quadratically many hyperedges are required for determining the atoms.

For future work, the idea of avoiding the exponential blow-up of the axiom dependency hypergraphs for general $\mathcal{EL}$-ontologies needs to be formalised, implemented and evaluated. It would also be interesting to extend the current approach to other locality notions such as $\top$- and $\bot\top^*$-locality and to richer languages such as $\mathcal{SROIQ}$.

References

1. X. Allamigeon. Strongly connected components of directed hypergraphs. *CoRR*, abs/1112.1444, 2011.
2. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI-05*. Morgan-Kaufmann Publishers, 2005.
3. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, 2007.
4. G. Gallo, G. Longo, S. Pallottino, and S. Nguyen. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42(23):177–201, 1993.
5. B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: theory and practice. *JAIR*, 31:273–318, 2008.
6. R. Nortje, A. Britz, and T. Meyer. A normal form for hypergraph-based module extraction for SROIQ. In *Proc. of AOW 2012*, volume 969 of *CEUR Workshop Proceedings*, pages 40–51. CEUR-WS.org, 2012.
7. M. Sharir. A strong connectivity algorithm and its applications to data flow analysis. *Computers & Mathematics with Applications*, 7(1):67–72, 1981.
8. B. Suntisrivaraporn. *Polynomial time reasoning support for design and maintenance of large-scale biomedical ontologies*. PhD thesis, TU Dresden, Germany, 2009.
9. R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.
10. D. Tsarkov. Improved algorithms for module extraction and atomic decomposition. In *Proc. of DL'12*, volume 846 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.
11. C. D. Vescovo, B. Parsia, U. Sattler, and T. Schneider. The modular structure of an ontology: Atomic decomposition. In *Proc. of IJCAI'11*, pages 2232–2237, 2011.