TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science • Institute of Theoretical Computer Science

Efficient Axiomatization of OWL 2 EL Ontologies from Data by means of Formal Concept Analysis

Description Logics and OWL

Description Logics (DLs) are formal languages used in knowledge-based systems that reason and make inferences about complex domains, particularly where precision and explainability are essential. By representing knowledge as ontologies built with DLs, these systems can perform automated reasoning to answer queries and thereby assist in making decisions based on the encoded knowledge. DLs are fundamental to the Semantic Web, a vision of the World Wide Web where information is represented in a machine-readable format. They provide the logical underpinning for the Web Ontology Language (OWL), which is widely used in the Semantic Web to enable better interoperability across different applications, domains, and natural languages. *Applications:* e-commerce, finance, healthcare, life sciences, etc.

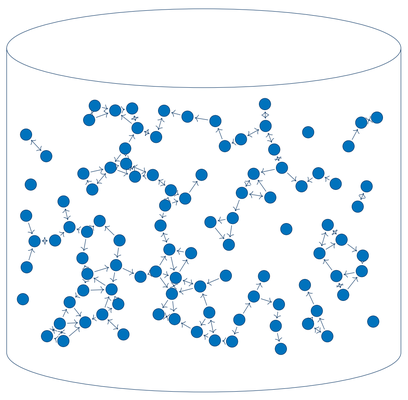
Among the different DLs, the $\mathcal{EL}$ family stands out as a lightweight option. $\mathcal{EL}$ is designed to strike a balance between expressivity and computational complexity, making it an ideal choice for applications where scalability and latency are crucial. It offers a more restricted set of constructs compared to other DLs, but can thus handle large-scale ontologies efficiently. The Web Ontology Language includes it as the profile OWL 2 EL.

Example: SNOMED CT contains the axiom

$$\mathit{Common\_cold} \sqsubseteq \mathit{Disease} \sqcap \exists \mathit{causative\_agent}.\mathit{Virus} \sqcap \exists \mathit{finding\_site}.\mathit{Upper\_respiratory\_tract\_structure} \sqcap \exists \mathit{pathological\_process}.\mathit{Infectious\_process}.$$

Input

- Graph data with node and edge labels, e.g. RDF triples
- Node labels $x : A$ with concept A
- Edge labels $(x, y) : r$ with role r
- Closure assumption needed, as otherwise only tautologies can be axiomatized.



Output

- A **base of axioms**, which only contains axioms valid in the input data and that is **complete**, i.e. it entails every valid axiom.
- Axioms formulated in the $\mathcal{EL}$ family of description logics (OWL 2 EL)
  - Complex concepts can be built from the concepts A and roles r in the input: $C ::= \top \mid \bot \mid A \mid C \sqcap C \mid \exists r.C$
  - Concept inclusions $C \sqsubseteq D$
  - Range restrictions $\top \sqsubseteq \forall r.C$
  - Role inclusions $R \sqsubseteq s$ where $R ::= \epsilon \mid r \mid R \circ R$
  - Syntactic sugar: disjointness axioms $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq \bot$, concept equivalences $C \equiv D$, domain restrictions $\exists r.\top \sqsubseteq C$, role equivalences $r \equiv s$, transitivity axioms $r \circ r \sqsubseteq r$, reflexivity axioms $\epsilon \sqsubseteq r$

Axiomatization

- Formal Concept Analysis (FCA) can compute a base of positive propositional implications $(p_1 \wedge \cdots \wedge p_m) \to (q_1 \wedge \cdots \wedge q_n)$.
- Concept inclusions $C \sqsubseteq D$ and implications $\bigwedge p_i \to \bigwedge q_j$ are similar.
- By reduction to FCA, a base of concept inclusions is computed.
- The role inclusions can be described by finite automata, which are transformed into a base of role inclusions.
- Range restrictions can be found by looking at all role successors.
- The final base is the union of all three bases.
- All steps need at most exponential time.

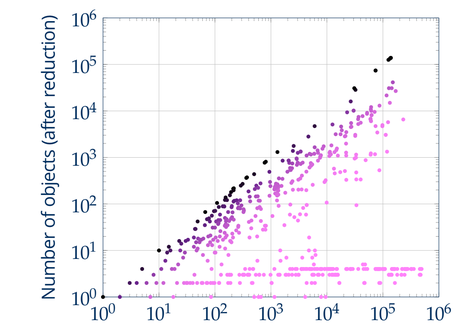
Saving Computation Time

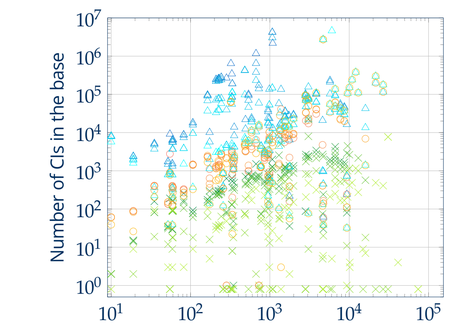
- Input data is reduced by grouping equi-similar objects.
- Option to dispense with disjointness axioms.
- Bound the size of complex concepts in the concept inclusions.

Implementation and Evaluation

- Prototype in Scala and with Java's Fork/Join framework
- Evaluated on 614 test datasets from real-world domains with up to 747,998 objects
- Computer server: 12 CPU cores at 2.80 GHz and 96 GB main memory (older than 10 years)
- Runtime environment: Oracle GraalVM EE 22.3 (Java 19)
- Limits: 8 hours, 80 GB

	$\geq 10^1$ $< 10^2$	$\geq 10^2$ $< 10^3$	$\geq 10^3$ $< 10^4$	$\geq 10^4$
× None (0,32)				
△ Fast (0,32)				
○ Can. (0,32)				
× None (1,8)				
△ Fast (1,8)				
○ Can. (1,8)				
× None (1,32)				
△ Fast (1,32)				
○ Can. (1,32)				
× None (2,32)				
△ Fast (2,32)				
○ Can. (2,32)				
× None (∞,32)				
△ Fast (∞,32)				
○ Can. (∞,32)				
× None (∞,∞)				
△ Fast (∞,∞)				
○ Can. (∞,∞)				





References

Catriel Beeri, Philip A. Bernstein; Rudolf Wille; Bernhard Ganter; Jean-Luc Guigues, Vincent Duquenne; Raymond Reiter; Sergei O. Kuznetsov; Marcel Wild; Monika R. Henzinger, Thomas A. Henzinger, Peter W. Kopke; Gerd Stumme; Sebastian Rudolph; Franz Baader, Sebastian Brandt, Carsten Lutz; Franz Baader, Felix Distel; Petr Krajča, Jan Outrata, Vilém Vychodil; Carsten Lutz, Robert Piro, Frank Wolter; Felix Distel; Yevgeny Kazakov, Markus Krötzsch, František Simančik; Radek Janoštík, Jan Konečný, Petr Krajča

Technische Universität Dresden
Faculty of Computer Science
Institute of Theoretical Computer Science

Francesco Kriegel
francesco.kriegel@tu-dresden.de
https://tu-dresden.de/inf/lat/francesco-kriegel