

Description Logic \mathcal{ALC}

Petr Křemen

November 21, 2025

1 Understanding \mathcal{ALC}

Consider the following \mathcal{ALC} theory $\mathcal{K} = (\mathcal{T}, \{\})$, where \mathcal{T} contains the following axioms:

$$\begin{aligned} \textit{Man} &\sqsubseteq \textit{Person} \\ \textit{Woman} &\sqsubseteq \textit{Person} \sqcap \neg \textit{Man} \\ \textit{Father} &\equiv \textit{Man} \sqcap \exists \textit{hasChild} \cdot \textit{Person} \\ \textit{GrandFather} &\equiv \exists \textit{hasChild} \cdot \exists \textit{hasChild} \cdot \top \\ \textit{Sister} &\equiv \textit{Person} \sqcap \neg \textit{Man} \sqcap \exists \textit{hasSibling} \cdot \textit{Person} \end{aligned}$$

Ex. 1 — What is the meaning of these axioms? Do they reflect your understanding of reality?

Answer (Ex. 1) — For example, the third axiom defines a concept *Father* as any *Man* that has some *Person* as a child. The fourth axiom is not well defined – it allows grandfathers to be women. More precise version of the fourth axiom might be e.g. $\textit{GrandFather} \equiv \textit{Man} \sqcap \exists \textit{hasChild} \cdot \exists \textit{hasChild} \cdot \top$.

Ex. 2 — Consider the following interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \bullet^{\mathcal{I}})$:

$$\begin{aligned} \Delta^{\mathcal{I}} &= \textit{Person}^{\mathcal{I}} = \{B, A\} \\ \textit{Man}^{\mathcal{I}} &= \{B\} \\ \textit{Woman}^{\mathcal{I}} &= \{A\} \\ \textit{Father}^{\mathcal{I}} &= \textit{GrandFather}^{\mathcal{I}} = \{B\} \\ \textit{hasChild}^{\mathcal{I}} &= \{(B, B)\} \\ \textit{hasSibling}^{\mathcal{I}} &= \{\} \\ \textit{Sister}^{\mathcal{I}} &= \{B\} \end{aligned}$$

1. Is \mathcal{I} a model \mathcal{K} ? If yes, decide, whether \mathcal{I} reflects reality.
2. We know that \mathcal{ALC} has the *tree model property* and *finite model property*. In case \mathcal{I} is a model, is \mathcal{I} tree-shaped? If not, find a model that is tree-shaped.

Answer (Ex. 2) — \mathcal{I} is not a model of \mathcal{K} , as it does not satisfy the last axiom: $Sister^{\mathcal{I}} \neq Person^{\mathcal{I}} \cap (\Delta^{\mathcal{I}} \setminus Man^{\mathcal{I}}) \cap \{x \in \Delta^{\mathcal{I}} | (\exists y \in \Delta^{\mathcal{I}})((x, y) \in hasSibling^{\mathcal{I}} \wedge y \in Person^{\mathcal{I}})\}$ – e.g. no $(B, \bullet) \notin hasSibling^{\mathcal{I}}$

Ex. 3 — How does the situation change when we consider \mathcal{I}_1 which coincides with \mathcal{I} , except that $Sister_1^{\mathcal{I}} = \{\}$?

Answer (Ex. 3) — Now, \mathcal{I}_1 is a model of \mathcal{K} as it satisfies all axioms. However, it does not reflect the reality well, as it states that a person B is his/her own child. This interpretation is finite, yet not tree-shaped. A tree-shaped model ensured by the *tree-model property* of \mathcal{ALC} is e.g. the following infinite model $\mathcal{I}_1 = (\Delta^{\mathcal{I}_1}, \bullet^{\mathcal{I}_1})$, where

$$\begin{aligned}
\Delta^{\mathcal{I}_1} &= Person^{\mathcal{I}_1} = \{PHILIP, CHARLES, WILLIAM\} \\
&= Man^{\mathcal{I}_1} = Father^{\mathcal{I}_1} = GrandFather^{\mathcal{I}_1} = \{A_1, A_2, \dots\}_{i=1.. \infty} \\
Woman^{\mathcal{I}_1} &= Sister^{\mathcal{I}_1} = \{\} \\
hasChild^{\mathcal{I}_1} &= \{(A_i, A_{i+1})\}_{i=1.. \infty} \\
hasSibling^{\mathcal{I}_1} &= \{\}
\end{aligned} \tag{1}$$

Ex. 4 — Using the vocabulary from \mathcal{K} , define the concept “A father having just sons.”

Answer (Ex. 4) — $FatherOfBoys \equiv Father \sqcap \forall hasChild \cdot Man$

Ex. 5 — Using the vocabulary from \mathcal{K} , define the concept “A man who has no brother, but at least one sister with at least one child.”

Answer (Ex. 5) — $HappyUncle \equiv Man \sqcap \exists hasSibling \cdot (Woman \sqcap \exists hasChild \cdot \top) \sqcap \forall hasSibling \cdot \neg Man$

Ex. 6 — During knowledge modeling, it is often necessary to specify:

global domain and range of given role, e.g. “By *hasChild* (role) we always connect a *Person* (domain) with another *Person* (range)”.

local range of given role, e.g. “Every father having only sons (domain) can be connected by *hasChild* (role) just with a *Man* (range)”.

Show, in which way it is possible to model global domain and range of these roles in \mathcal{ALC} .

Answer (Ex. 6) — Global domain and range can be modeled as:

$$\begin{aligned}
\exists hasChild \cdot \top &\sqsubseteq Person \\
\top &\sqsubseteq \forall hasChild \cdot Person
\end{aligned} \tag{2}$$

Local range is similar and only replaces the top concepts in the global range axiom:

$$\begin{aligned} \exists hasChild \cdot FatherOfSons &\sqsubseteq Person \\ FatherOfSons &\sqsubseteq \forall hasChild \cdot Man \end{aligned} \quad (3)$$

2 Inference Procedures

Ex. 7 — Why inconsistency of an OWL-DL ontology is a problem? What is its consequence?

Answer (Ex. 7) — The logical calculus behind OWL-DL is based on first order logic. Thus, an inconsistent ontology entails all axioms.

Ex. 8 — Show that disjointness of two concepts can be reduced to unsatisfiability of a single concept.

Answer (Ex. 8) — Let's reproduce the flow of equivalent operations for this simple transformation:

$$\mathcal{K} \models C \sqsubseteq \neg D \quad (4)$$

$$(\forall \mathcal{I})(\mathcal{I} \models \mathcal{K}) \Rightarrow (\mathcal{I} \models (C \sqsubseteq \neg D)) \quad (5)$$

$$(\forall \mathcal{I})(\mathcal{I} \models \mathcal{K}) \Rightarrow (C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \setminus D^{\mathcal{I}}) \quad (6)$$

$$(\forall \mathcal{I})(\mathcal{I} \models \mathcal{K}) \Rightarrow (C^{\mathcal{I}} \cap D^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}} \setminus D^{\mathcal{I}}) \cap D^{\mathcal{I}} = \{\}) \quad (7)$$

$$(\forall \mathcal{I})(\mathcal{I} \models \mathcal{K}) \Rightarrow (\mathcal{I} \models (C \sqcap D \sqsubseteq \perp)) \quad (8)$$

$$\mathcal{K} \models C \sqcap D \sqsubseteq \perp \quad (9)$$

$$\mathcal{K} \models (C \sqcap D) \text{ is unsatisfiable} \quad (10)$$

Ex. 9 — A concept C is satisfiable w.r.t. \mathcal{K} iff it is interpreted as a non-empty set in at least one model of \mathcal{K} . Is it possible to find out that C is interpreted as a non-empty set in all models of \mathcal{K} ?

Answer (Ex. 9) — If $\mathcal{K} \cup (C \sqsubseteq \perp)$ is inconsistent for consistent \mathcal{K} , then $C^{\mathcal{I}} \neq \{\}$ for each model \mathcal{I} of \mathcal{K} .

3 Tableaux Algorithm for \mathcal{ALC}

Ex. 10 — Decide, whether the \mathcal{ALC} concept $\exists hasChild \cdot (Student \sqcap Employee) \sqcap \neg(\exists hasChild \cdot Student \sqcap \exists hasChild \cdot Employee)$ is satisfiable (w.r.t. an empty TBox). Show the run of the tableau algorithm in detail.

Ex. 11 — Decide, whether the theory/ontology $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is consistent. Show the run of the tableau algorithm in detail.

$$\bullet \mathcal{T} = \{\exists hasChild \cdot \top \equiv Parent\}$$

• $\mathcal{A} = \{hasChild(JOHN, MARY), Woman(MARY)\}$

Ex. 12 — Decide and show, whether the ontology

$$\mathcal{K}_1 = (\mathcal{T} \cup \{Parent \sqsubseteq \forall hasChild \cdot \neg Woman\}, \mathcal{A})$$

is consistent.

Ex. 13 — Decide and show, whether the ontology

$$\mathcal{K}_2 = (\mathcal{T} \cup \{Parent \sqsubseteq \exists hasChild \cdot Parent\}, \mathcal{A})$$

is consistent.

Answer (Ex. 13) — To check the consistency, we will use the tableau algorithm for \mathcal{ALC} . To keep description compact, we shorten $Parent, hasChild, Woman$ as P, h, W . First, we need to internalize the TBOX

$$\begin{aligned} \{\exists h \cdot \top &\equiv P, \\ P &\sqsubseteq \exists h \cdot P\} \end{aligned}$$

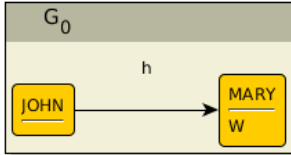
into the single axiom $\top \sqsubseteq \top_C$, such that \top_C is:

$$(\neg(\exists h \cdot \top) \sqcup P) \sqcap (\neg P \sqcup \exists h \cdot \top) \sqcap (\neg P \sqcup \exists h \cdot P) \quad (11)$$

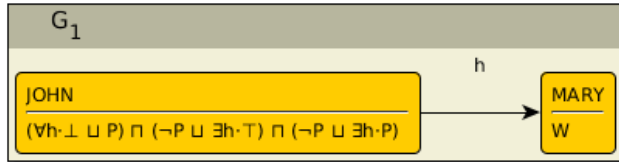
Now, we transform all concepts in \mathcal{K}_2 (here only \top_C) into negational normal form. T_C :

$$(\forall h \cdot \perp \sqcup P) \sqcap (\neg P \sqcup \exists h \cdot \top) \sqcap (\neg P \sqcup \exists h \cdot P) \quad (12)$$

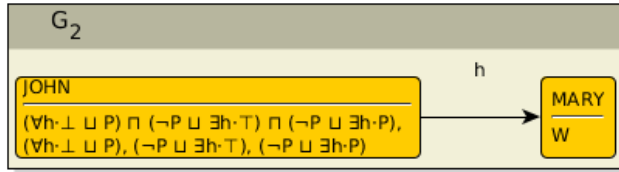
The initial state $S_0 = \{G_0\}$ of the algorithm contains a single *completion graph* G_0 representing the input ABOX



G_0 does not contain a direct clash (there is neither \perp , nor A and $\neg A$ in the label of a single node). G_0 is not complete w.r.t \mathcal{ALC} completion rules, as the $\sqsubseteq -$ rule is applicable. Applying the rule on the node $JOHN$ we get a new tableau algorithm state $S_1 = \{G_1\}$ where G_1 is

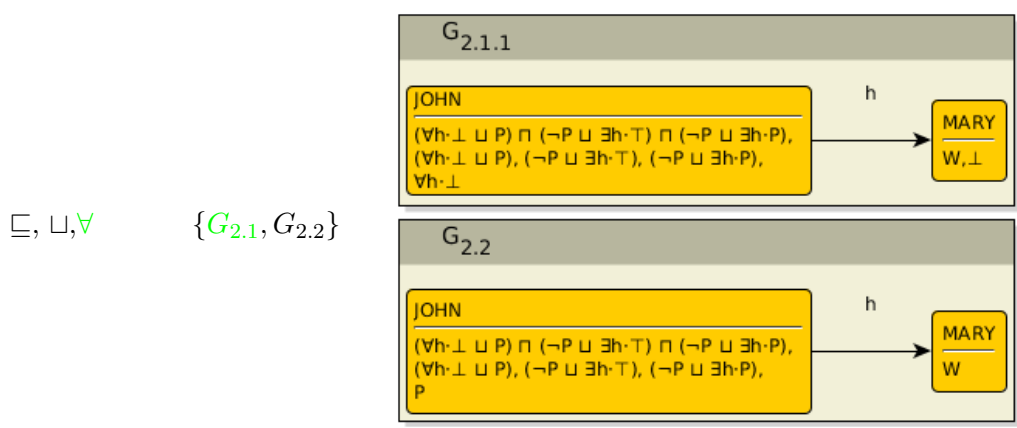
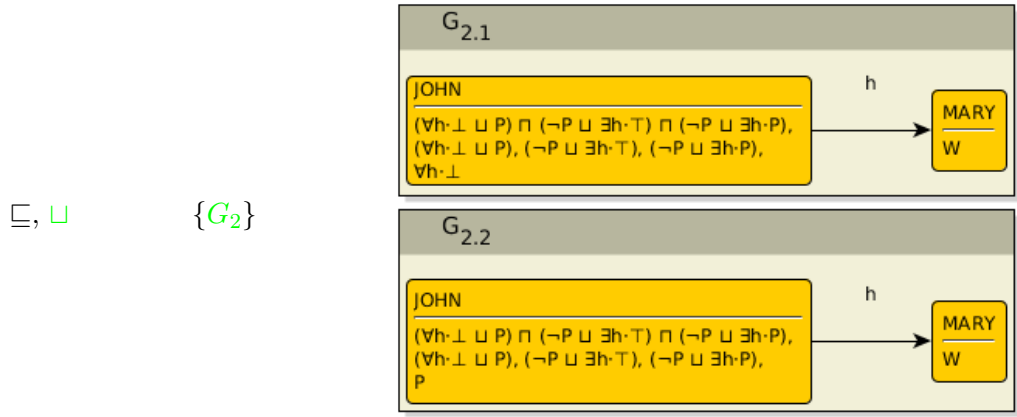


G_1 is clash-free and not complete as well. Two rules are applicable – the \sqsubseteq -rule and the \sqcap -rule. We apply the latter one (as a heuristic, we expect the clash to be found earlier using the \sqcap -rule) and get the state $S_2 = \{G_2\}$ where G_2 is

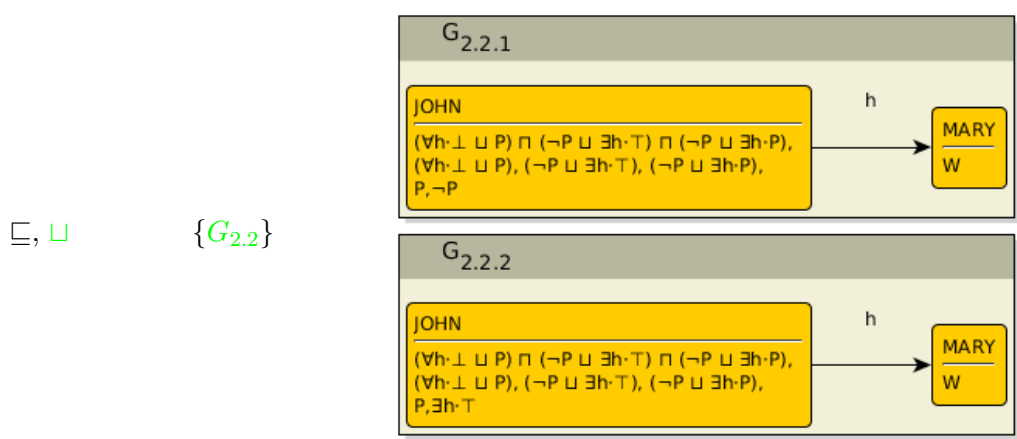


From now on we will proceed more quickly forward and show only tableau reasoner state with the information about rule application and clashing graphs. Whenever more rules are applicable, the one that is applied is marked in green, as well as the chosen graph. Graphs containing a clash are no more shown in the algorithm state.

applicable rules	state before applying the rule	state after applying the rule
------------------	--------------------------------	-------------------------------



$G_{2.1.1}$ contains a direct clash.

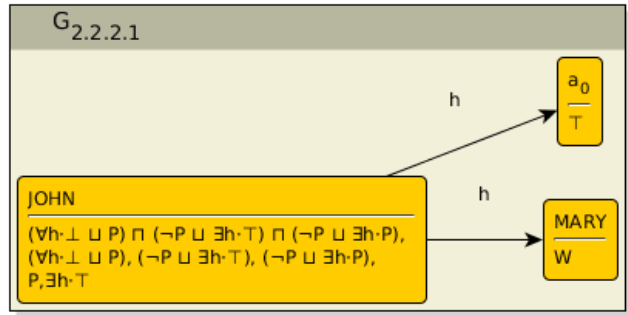


$G_{2.2.1}$ contains a direct clash.

applicable rules	state before applying the rule	state after applying the rule
------------------	--------------------------------	-------------------------------

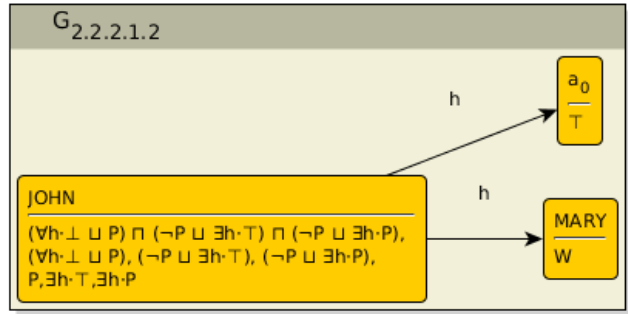
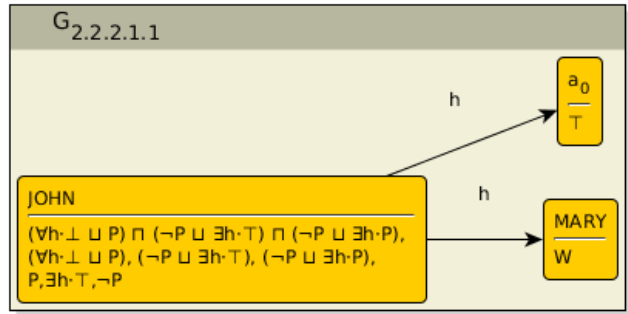
$\sqsubseteq, \sqcup, \exists$

$\{G_{2.2.2}\}$



\sqsubseteq, \sqcup

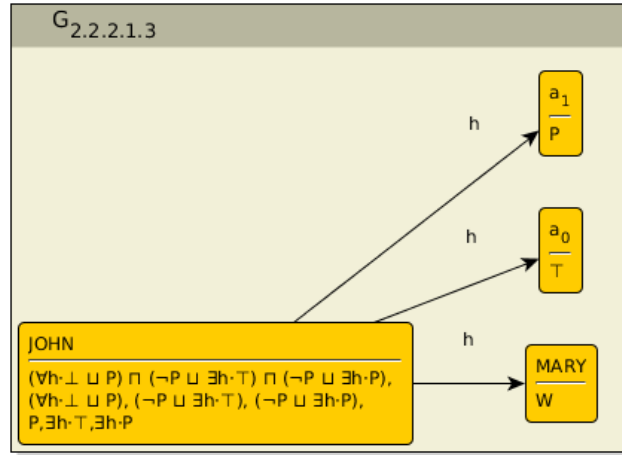
$\{G_{2.2.2.1}\}$



$G_{2.2.2.1.1}$ contains a direct clash.

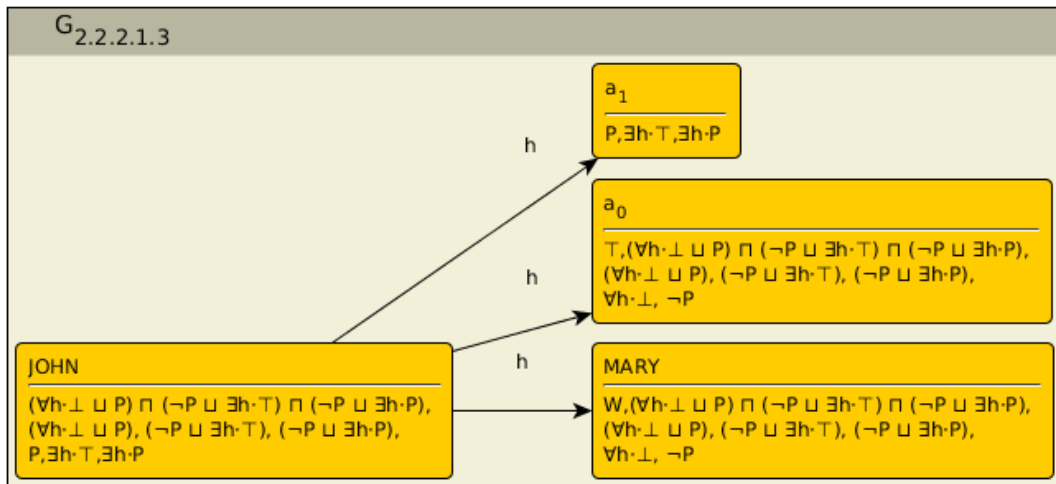
applicable rules	state before applying the rule	state after applying the rule
------------------	--------------------------------	-------------------------------

\sqsubseteq, \exists $\{G_{2.2.2.1.2}\}$



a_1 is blocked by *JOHN* as the label of a_0 is a subset of the label of *JOHN*.

Now, applying the sequence of rules ($\sqsubseteq, \sqcup, \sqcup, \sqcup$) for *MARY*, a_0 and a_1 , we get the graph¹



¹We do not depict the whole algorithm state, as it contains several graphs of the similar size like G_3 due to the fact that the presence of the label $\forall h \cdot \perp$ in the node a_0 and *MARY* does lead to a clash, contrary to the case of *JOHN*. This fact generates several alternative disjuncts for each node. Also notice that we chose one set of disjuncts for a_1 and another set of disjuncts for *MARY* and a_0 in order to avoid clash.

In this graph, a_1 is blocked by *JOHN* and thus, the \exists rules do not apply. Therefore, this graph is complete and clash-free. The ontology \mathcal{K}_2 is consistent.

4 Practically

Ex. 14 — Go through the Protégé Crash Course on the tutorial web pages.

Ex. 15 — Model the ontology in Section 1 in Protégé and check (using the Pellet/Hermit reasoner) whether your solutions in the previous tasks were correct.

Ex. 16 — Adjust the Pizza ontology (<https://github.com/owlcs/pizza-ontology>), so that the class *IceCream* and *CheesyVegetableTopping* become satisfiable. Explain, why the Pizza ontology is consistent, although it contains unsatisfiable classes.

Ex. 17 — Upload the original pizza ontology into GraphDB - try different repository types (OWL-Max, OWL-Horst) and see how the inferences differ (e.g. Find all kinds of food, find all kinds of CheesyPizza). Notice the weak OWL reasoning capabilities in GraphDB – to use more complicated OWL reasoning you might export inferences using "Export inferred axioms as ontology" and import into GraphDB.