Querying Description Logics

Petr Křemen
petr.kremen@fel.cvut.cz

FEL ČVUT

Our plan

Conjunctive Queries

Evaluation of Conjunctive Queries in $\mathcal{ALC}$

Conjunctive Queries

Query Types

Conjunctive (ABox) queries – queries asking for individual tuples complying with a graph-like pattern.

Metaqueries – queries asking for individual/concept/role tuples. There are several languages for metaqueries, e.g. SPARQL-DL, OWL-SAIQL, etc.

Example

In SPARQL-DL, the query “Find all people together with their type.” can be written as follows :

$$Type(?x, ?c), SubClassOf(?c, Person)$$

Conjunctive (ABox) queries

Conjunctive (ABox) queries are analogous to database SELECT-PROJECT-JOIN queries. A conjunctive query is in the form

$$Q(?x_1, \ldots, ?x_D) \leftarrow t_1, \ldots t_T,$$

where each t_i is either $C(y_k)$, or $R(y_k, y_l)$. Each y_i is either (i) an individual from the ontology, or (ii) variable from a new set V (variables will be differentiated from individuals by the prefix "?") and C denotes a concept and R denotes a role. Next, we need all $?x_i$ to be present also in one of t_i.

Example

"Find all mothers and their daughters having at least one brother." :

$$\begin{aligned} Q(?x, ?z) \quad \leftarrow \quad & Woman(?x), hasChild(?x, ?y), hasChild(?x, ?z), \\ & Man(?y), Woman(?z) \end{aligned}$$

Conjunctive ABox Queries – Semantics

- Conjunctive queries of the form $Q()$ are called *boolean* – such queries only test existence of a relational structure in each model $\mathcal{I}$ of the ontology $\mathcal{K}$.
- Consider any interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. *Evaluation* η is a function from the set of individuals and variables into $\Delta^{\mathcal{I}}$ that coincides with $\mathcal{I}$ on individuals.
- Then $\mathcal{I} \models_{\eta} Q()$, iff
  - $\eta(y_k) \in C^{\mathcal{I}}$ for each atom $C(y_k)$ from $Q()$ and
  - $\langle \eta(y_k), \eta(y_l) \rangle \in R^{\mathcal{I}}$ for each atom $R(y_k, y_l)$ from $Q()$
- Interpretatino $\mathcal{I}$ is a model of $Q()$, iff $\mathcal{I} \models_{\eta} Q()$ for some η.
- Next, $\mathcal{K} \models Q()$ ($Q()$ is satisfiable in $\mathcal{K}$) iff $\mathcal{I} \models Q()$ whenever $\mathcal{I} \models \mathcal{K}$

Conjunctive ABox Queries – Variables

- Queries without variables are not practically interesting. For queries with variables we define semantics as follows. An N-tuple $\langle i_1, \ldots, i_n \rangle$ is a *solution* to $Q(?x_1, \ldots, ?x_n)$ in theory $\mathcal{K}$, whenever $\mathcal{K} \models Q'()$, for a boolean query Q' obtained from Q by replacing all occurences of $?x_1$ in all t_k by an individual i_1, etc.
- In conjunctive queries two types of variables can be defined:

distinguished
: occur in the query head as well as body, e.g. $?x, ?z$ in the previous example. These variables are evaluated as domain elements that are necessarily interpretations of some individual from $\mathcal{K}$. That individual is the binding to the distinguished variable in the query result.

undistinguished
: occur only in the query body, e.g. $?y$ in the previous example. Their can be interpretated as any domain elements.

Conjunctive Queries – Examples

Example

Let's have a theory $\mathcal{K}_4 = (\emptyset, \{(\exists R_1 \cdot C_1)(i_1), R_2(i_1, i_2), C_2(i_2)\})$.

- Does $\mathcal{K} \models Q_1()$ hold for $Q_1() \leftarrow R_1(?x_1, ?x_2)$?
- What are the solutions of the query $Q_2(?x_1) \leftarrow R_1(?x_1, ?x_2)$ for $\mathcal{K}$?
- What are the solutions of the query $Q_3(?x_1, ?x_2) \leftarrow R_1(?x_1, ?x_2)$ for $\mathcal{K}$?

Evaluation of Conjunctive Queries in $\mathcal{ALC}$

Satisfiability of $\mathcal{ALC}$ Boolean Queries

- Satisfiability of the boolean query $Q()$ having a tree shape can be checked by means of the *rolling-up technique*.
  - Each two atoms $C_1(y_k)$ and $C_2(y_k)$ can be replaced by a single query atom of the form $(C_1 \sqcap C_2)(y_k)$.
  - Each query atom of the form $R(y_k, y_l)$ can be replaced by the term $(\exists R \cdot X)(y_k)$, if y_l occurs in at most one other query atom of the form $C(y_l)$ (if there is no $C(y_l)$ atom in the query, consider w.l.o.g. that C is $\top$). X equals to
    - (i) C, whenever y_l is a variable,
    - (ii)$C \sqcap Y_l$, whenever y_l is an individual. Y_l is a *representative concept* of individual y_l occuring neither in $\mathcal{K}$ nor in Q. For each y_l it is necessary to extend ABox of $\mathcal{K}$ with concept assertion $Y_l(y_l)$.

Satisfiability of $\mathcal{ALC}$ Boolean Queries (2)

... after rolling-up the query we obtain the query $Q()' \leftarrow C(y)$, that is satisfied in $\mathcal{K}$, iff $Q()$ is satisfied in $\mathcal{K}$:

- If y is an individual, then $Q'()$ is satisfied, whenever $\mathcal{K} \models C(y)$ (i.e. $\mathcal{K} \cup \{(\neg C)(y)\}$ is inconsistent)
- If y is a variable, then $Q'()$ is satisfied, whenever $\mathcal{K} \cup \{C \sqsubseteq \bot\}$ is inconsistent. Why ?

Example

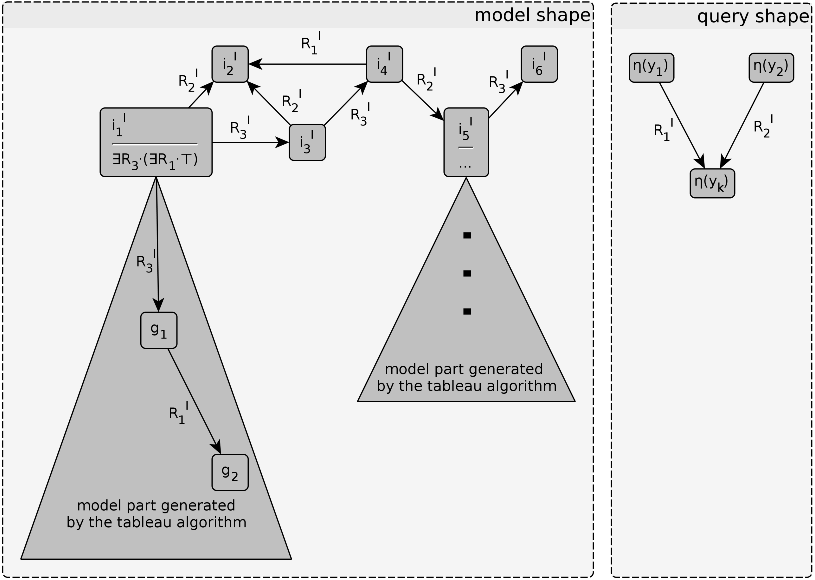
Consider a query $Q_4() \leftarrow R_1(?x_1, ?x_2), R_2(?x_1, ?x_3), C_2(?x_3)$. This query can be rolled-up into the query $Q_4' \leftarrow (\exists R_1 \cdot \top \sqcap \exists R_2 \cdot C_2)(?x_1)$. This query is satisfiable in $\mathcal{K}_4$, as $\mathcal{K}_4 \cup \{(\exists R_1 \cdot \top \sqcap \exists R_2 \cdot C_2) \sqsubseteq \bot\}$ is inconsistent.

Satisfiability of Boolean Queries in $\mathcal{ALC}$ (3)

... and what to do with queries with distinguished variables ?

- Let's consider just queries that form "connected component" and contain for some variable y_k at least two query atoms of the form $R_1(y_1, y_k)$ and $R_2(y_2, y_k)$.
- Question: *Why is it enough to take just one connected component?*
- **Let's make use of the tree model property of $\mathcal{ALC}$. Each pair of atoms $R_1(y_1, y_k)$ and $R_2(y_2, y_k)$ can be satisfied only if y_k is interpreted as a domain element, that is an interpretation of an individual – y_k can be treated as distinguished. Why (see next slide) ?**
- For $\mathcal{SHOIN}$ and $\mathcal{SROIQ}$ there is no sound and complete decision procedure for general boolean queries.

$\mathcal{ALC}$ Model Example



Queries with Distinguished Variables – naive pruning

Consider arbitrary query $Q(?x_1, \ldots, ?x_D)$. How to evaluate it ?

- **naive way**: Replace each distinguished variable x_i with each individual occuring in $\mathcal{K}$. *Solutions* are those D-tuples $\langle i_1, \ldots, i_D \rangle$, for which a boolean query created from Q by replacing each x_k with i_k is satisfiable.

Example

Remind that $\mathcal{K}_4 = (\emptyset, \{(\exists R_1 \cdot C_1)(i_1), R_2(i_1, i_2), C_2(i_2)\})$. The query

$$Q_5(?x_1) \leftarrow R_1(?x_1, ?x_2), R_2(?x_1, ?x_3), C_2(?x_3)$$

has *solution* $\langle i_1 \rangle$ as

$$Q_5'() \leftarrow R_1(i_1, ?x_2), R_2(i_1, ?x_3), C_2(?x_3)$$

can be rolled into $Q_5''()$ for which $\mathcal{K}_4 \models Q_5''$:

$$Q_5''() \leftarrow (\exists R_1 \cdot \top \sqcap \exists R_2 \cdot C_2)(i_1)$$

Queries with Distinguished Variables – naive pruning

... another example

Example

The query

$$Q_6(?x_1, ?x_3) \leftarrow R_1(?x_1, ?x_2), R_2(?x_1, ?x_3), C_2(?x_3)$$

has *solution* $\langle i_1, i_2 \rangle$ as

$$Q_6'() \leftarrow R_1(i_1, ?x_2), R_2(i_1, i_2), C_2(i_2)$$

can be rolled into Q_6'' for which $\mathcal{K}_4 \cup \{\mathbf{I_2(i_2)}\} \models Q_6''$.

$$Q_6''() \leftarrow (\exists R_1 \cdot \top \sqcap \exists R_2 \cdot (C_2 \sqcap I_2))(i_1).$$

Similarly $Q_7(?x_1, ?x_2) \leftarrow R_1(?x_1, ?x_2), R_2(?x_1, ?x_3), C_2(?x_3)$ has no solution.

Queries with Distinguished Variables – iterative pruning

- **... a bit more clever strategy than replacing all variables**: First, let's replace just the first variable $?x_1$ with each individual from $\mathcal{K}$, resulting in Q_2. If the subquery of Q_2 containing all query atoms from Q_2 without distinguished variables is not a logical consequence of $\mathcal{K}$, then we do not need to test potential bindings for other variables.
- Many other optimizations are available.

laboratory Gerstner

Queries with Distinguished Variables – iterative pruning

Example

For the query $Q_6(?x_1, ?x_3)$, the naive strategy needs to check four different bindings (resulting in four tableau algorithm runs)

$$\begin{gathered}\langle i_1, i_1\rangle, \\ \langle \mathbf{i_1}, \mathbf{i_2}\rangle, \\ \langle i_2, i_1\rangle, \\ \langle i_2, i_2\rangle.\end{gathered}$$

Out of them only $\langle i_1, i_2\rangle$ is a solution for Q_6. Consider only partial binding $\langle i_2\rangle$ for $?x_1$. Applying this binding to Q_6 we get $Q_7(?x_3) = R_1(i_2, ?x_2), R_2(i_2, ?x_3), C_2(?x_3)$. Its distinguished-variable-free subquery is $Q'_7() = R_1(i_2, ?x_2)$ and $\mathcal{K}_4 \not\models Q'_7$. Because of **monotonicity** of $\mathcal{ALC}$, we do not need to check the two bindings for $?x_3$ in this case which saves us one tableau algorithm run.