

# Foundations of Knowledge Representation

## Lecture 6: Description Logics – Reasoning with Data

Hannes Straß

based on slides of  
Bernardo Cuenca Grau,  
Ian Horrocks, and  
Przemysław Wałęga



# Reasoning with Data

So far we have focused on **terminological reasoning**

- TBoxes represent general, conceptual domain knowledge
- Terminological reasoning is key to design error-free TBoxes

## New Scenario: **Ontology-based data access (OBDA)**

- We have built an (error-free) TBox for our domain
- We want to populate TBox with data (add an ABox)  
ABox & TBox should be **compatible** (no inconsistencies)
- Then, we can **query the data**  
TBox provides vocabulary for queries  
Answers reflect both TBox knowledge and ABox data

# Compatibility of Data and Knowledge

The ABox data should be compatible with the TBox knowledge

$$\mathcal{T} = \{\text{GradSt} \sqcap \text{UnderGradSt} \sqsubseteq \perp\}$$

$$\mathcal{A} = \{\text{John} : \text{GradSt}, \text{John} : \text{UnderGradSt}\}$$

Nothing wrong with the TBox

Nothing wrong with the ABox

There is an obvious error when putting them together

To detect these situations we use the following problem:

**Knowledge Base consistency:**

An instance is knowledge base  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ .

The answer is **true** iff a model  $\mathcal{I} \models \mathcal{K}$  exists.

In a FOL setting,  $\mathcal{K}$  is consistent if and only if  $\pi(\mathcal{K})$  is satisfiable.

# Tableau Algorithm for KB Consistency

Tableau-based knowledge base consistency algorithm:

- **Input:** Knowledge Base  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- **Output:** true iff  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$  is consistent

- 1 Start with input ABox  $\mathcal{A}$
- 2 Apply expansion rules until completion or clash
- 3 Blocking only involves individuals not occurring in  $\mathcal{A}$

Exploit **forest-model property**: construct **forest-shaped** ABox  
root (ABox) individuals can be arbitrarily connected  
tree individuals (introduced by  $\exists$ -rule) form trees

## Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

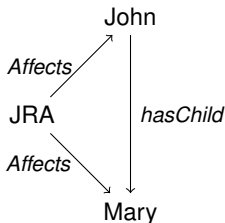
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{\text{JuvArth}\}$

$\text{con}_{\mathcal{A}}(\text{John}) = \emptyset$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \emptyset$

## Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

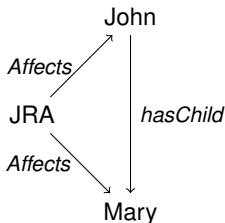
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{\text{JuvArth}, \text{Arth}, \text{JuvDis}\}$

$\text{con}_{\mathcal{A}}(\text{John}) = \emptyset$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \emptyset$

## Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

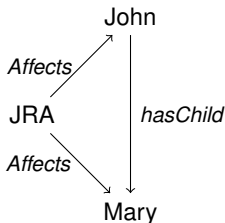
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{\text{JuvArth}, \text{Arth}, \text{JuvDis}, \exists \text{Damages.Joint}\}$

$\text{con}_{\mathcal{A}}(\text{John}) = \emptyset$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \emptyset$

## Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

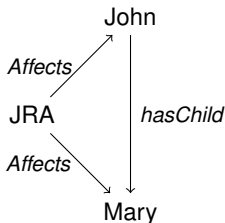
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{ \text{JuvArth}, \text{Arth}, \text{JuvDis}, \exists \text{Damages.Joint}, \exists \text{Affects.Child}, \forall \text{Affects.Child} \}$

$\text{con}_{\mathcal{A}}(\text{John}) = \emptyset$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \emptyset$



## Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

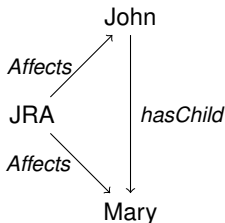
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{ \text{JuvArth}, \text{Arth}, \text{JuvDis}, \exists \text{Damages.Joint}, \exists \text{Affects.Child}, \forall \text{Affects.Child} \}$

$\text{con}_{\mathcal{A}}(\text{John}) = \{ \text{Child} \}$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \{ \text{Child} \}$

# Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

JuvDis  $\sqsubseteq \exists \textit{Affects}. \textit{Child} \sqcap \forall \textit{Affects}. \textit{Child}$

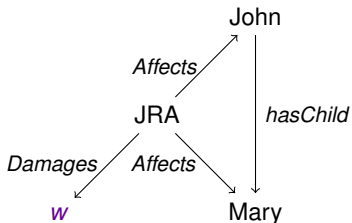
$\exists \textit{hasChild}. \top \sqsubseteq \textit{Adult}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}. \textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\textit{JRA}) = \{ \textit{JuvArth}, \textit{Arth}, \textit{JuvDis}, \exists \textit{Damages}. \textit{Joint}, \exists \textit{Affects}. \textit{Child}, \forall \textit{Affects}. \textit{Child} \}$

$\text{con}_{\mathcal{A}}(\textit{John}) = \{ \textit{Child} \}$

$\text{con}_{\mathcal{A}}(\textit{Mary}) = \{ \textit{Child} \}$

$\text{con}_{\mathcal{A}}(w) = \{ \textit{Joint} \}$

# Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

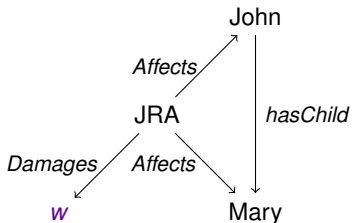
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{ \text{JuvArth}, \text{Arth}, \text{JuvDis}, \exists \text{Damages.Joint}, \exists \text{Affects.Child}, \forall \text{Affects.Child} \}$

$\text{con}_{\mathcal{A}}(\text{John}) = \{ \text{Child}, \text{Adult} \}$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \{ \text{Child} \}$

$\text{con}_{\mathcal{A}}(w) = \{ \text{Joint} \}$

# Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$\text{JuvDis} \sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

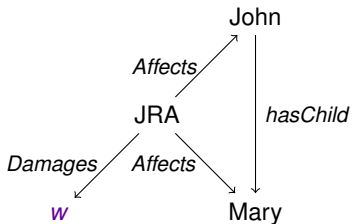
$\exists \text{hasChild.T} \sqsubseteq \text{Adult}$

$\text{Adult} \sqsubseteq \neg \text{Child}$

$\text{Arth} \sqsubseteq \exists \text{Damages.Joint}$

$\text{JuvArth} \sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

Tableau expansion (simplified):



$\text{con}_{\mathcal{A}}(\text{JRA}) = \{ \text{JuvArth}, \text{Arth}, \text{JuvDis}, \exists \text{Damages.Joint}, \exists \text{Affects.Child}, \forall \text{Affects.Child} \}$

$\text{con}_{\mathcal{A}}(\text{John}) = \{ \text{Child}, \text{Adult}, \neg \text{Child} \}$

$\text{con}_{\mathcal{A}}(\text{Mary}) = \{ \text{Child} \}$

$\text{con}_{\mathcal{A}}(w) = \{ \text{Joint} \}$

# Tableau Example (Simplified)

(JRA, John) : *Affects*

JRA : *JuvArth*

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

$JuvDis \sqsubseteq \exists Affects.Child \sqcap \forall Affects.Child$

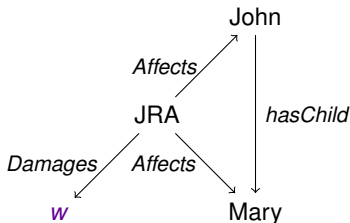
$\exists hasChild.\top \sqsubseteq Adult$

$Adult \sqsubseteq \neg Child$

$Arth \sqsubseteq \exists Damages.Joint$

$JuvArth \sqsubseteq Arth \sqcap JuvDis$

Tableau expansion (simplified):



$con_{\mathcal{A}}(JRA) = \{JuvArth, Arth, JuvDis, \exists Damages.Joint, \exists Affects.Child, \forall Affects.Child\}$

$con_{\mathcal{A}}(John) = \{Child, Adult, \neg Child\}$

$con_{\mathcal{A}}(Mary) = \{Child\}$

$con_{\mathcal{A}}(w) = \{Joint\}$

# Querying the Data

It does not make sense to query an inconsistent  $\mathcal{K}$  (previous example)

- An inconsistent  $\mathcal{K}$  entails all formulas.
- We (typically) fix inconsistencies before start asking queries.

Once determined that  $\mathcal{K}$  is consistent, we want to **query the data**:

- Which children are affected by a juvenile arthritis?
- Which drugs are used to treat JRA?
- Who is affected by an arthritis and is allergic to steroids?

Similar to the types of queries one would pose to a database.

```
SELECT Child.cname
FROM Child, Affects, JuvArth
WHERE Child.cname = Affects.cname AND
      Affects.dname = JuvArth.dname
```

# Querying the Data: Simple Queries

The basic data queries ask for all the instances of a concept:

$$q_1(x) = \text{Child}(x)$$

Set of children?

$$q_2(x) = (\text{Dis} \sqcap \exists \text{Damages.Joint})(x)$$

Set of diseases affecting a joint?

How to (naively) answer these queries? Try each individual name!

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$

( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

JuvDis  $\sqsubseteq \exists \text{Affects.Child} \sqcap \forall \text{Affects.Child}$

Adult  $\sqsubseteq \neg \text{Child}$

Arth  $\sqsubseteq \exists \text{Damages.Joint}$

JuvArth  $\sqsubseteq \text{Arth} \sqcap \text{JuvDis}$

$\mathcal{K} \models \text{JRA} : \text{Child?}$  **No.** JRA is not an answer to  $q_1$

$\mathcal{K} \models \text{John} : \text{Child?}$  **Yes!** John is an answer to  $q_1$

$\mathcal{K} \models \text{Mary} : \text{Child?}$  **Yes!** Mary is an answer to  $q_1$

# Querying the Data: Simple Queries

So, we are interested in the following decision problem:

## Concept Instance Checking:

Given individual name  $a$ , concept  $C$  and KB  $\mathcal{K}$ ,  
an instance is a triple  $\langle a, C, \mathcal{K} \rangle$ .

The answer is **true** iff  $\mathcal{K} \models a : C$

In  $\mathcal{ALC}$  (and extensions) this problem is reducible to KB consistency:

$(\mathcal{T}, \mathcal{A}) \models a : C$     iff     $(\mathcal{T}, \mathcal{A} \cup \{a : \neg C\})$  inconsistent



# Querying the Data: Simple Queries

So, we are interested in the following decision problem:

## Concept Instance Checking:

Given individual name  $a$ , concept  $C$  and KB  $\mathcal{K}$ ,  
an instance is a triple  $\langle a, C, \mathcal{K} \rangle$ .

The answer is **true** iff  $\mathcal{K} \models a : C$

In  $\mathcal{ALC}$  (and extensions) this problem is reducible to KB consistency:

$$(\mathcal{T}, \mathcal{A}) \models a : C \quad \text{iff} \quad (\mathcal{T}, \mathcal{A} \cup \{a : \neg C\}) \text{ inconsistent}$$

Note that we can assume, w.l.o.g., that  $C$  is a concept name:

$$(\mathcal{T}, \mathcal{A}) \models a : C \quad \text{iff} \quad (\mathcal{T} \cup \{X \equiv C\}, \mathcal{A}) \models a : X$$

where  $X$  is a concept name that does not occur in  $\mathcal{T}$  or  $\mathcal{A}$ .

# Querying the Data: Simple Queries

What about instances of a role:

$q_2(x, y) = hasChild(x, y)$  Set of parent-child tuples?

How to (naively) answer these queries? Try each pair of individuals!

ABox $\mathcal{A}$	TBox $\mathcal{T}$	$(\mathcal{K} = (\mathcal{T}, \mathcal{A}))$
JRA : JuvArth	JuvDis $\sqsubseteq$	$\exists Affects.Child \sqcap \forall Affects.Child$
(JRA, Mary) : <i>Affects</i>	Adult $\sqsubseteq$	$\neg Child$
(John, Mary) : <i>hasChild</i>	Arth $\sqsubseteq$	$\exists Damages.Joint$
	JuvArth $\sqsubseteq$	$Arth \sqcap JuvDis$

$\mathcal{K} \models (John, John) : hasChild?$  **No.** (John, John) is not an answer to  $q_2$

$\mathcal{K} \models (John, Mary) : hasChild?$  **Yes!** (John, Mary) is an answer to  $q_2$

$\mathcal{K} \models (John, JRA) : hasChild?$  **No.** (John, John) is not an answer to  $q_2$

...

# Querying the Data: Simple Queries

So, we are interested in the following decision problem:

## Role Instance Checking:

Given a pair of individual names  $(a, b)$ , role  $R$  and KB  $\mathcal{K}$ , an instance is a triple  $\langle (a, b), R, \mathcal{K} \rangle$ .

The answer is **true** iff  $\mathcal{K} \models (a, b) : R$

Can this problem be reduced to knowledge base consistency?

$(\mathcal{T}, \mathcal{A}) \models (a, b) : R$  iff  $(\mathcal{T}, \mathcal{A} \cup \dots)$  is inconsistent

# Querying the Data: Simple Queries

So, we are interested in the following decision problem:

## Role Instance Checking:

Given a pair of individual names  $(a, b)$ , role  $R$  and KB  $\mathcal{K}$ , an instance is a triple  $\langle (a, b), R, \mathcal{K} \rangle$ .

The answer is **true** iff  $\mathcal{K} \models (a, b) : R$

Can this problem be reduced to knowledge base consistency?

$(\mathcal{T}, \mathcal{A}) \models (a, b) : R$  iff  $(\mathcal{T}, \mathcal{A} \cup \{a : \forall R.X, b : \neg X\})$  is inconsistent

where  $X$  is a concept name that does not occur in  $\mathcal{T}$  or  $\mathcal{A}$ .

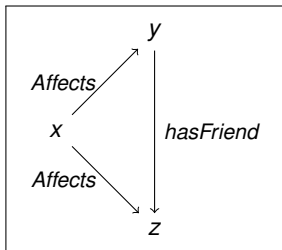
# Limitations of Concept-based Queries

Some natural queries cannot be expressed using a concept:

$$q(y) = \exists x \exists z (Affects(x, y) \wedge Affects(x, z) \wedge hasFriend(y, z))$$

Set of people ( $y$ ) affected by the same disease as a friend?

Query Graph:



We can only represent **tree-like** queries as concepts

Related to the tree model property of DLs

We need a more expressive query language ...

# Conjunctive Queries

The language of **conjunctive queries**

- Generalises concept-based queries in a natural way  
arbitrarily-shaped queries vs tree-like queries
- Widely used as a query language in databases  
Corresponds to Select-Project-Join fragment of rel. algebra  
Fragment of relational calculus using only  $\exists$  and  $\wedge$
- Implemented in most DBMS

**We next study the problem of CQ answering over DL knowledge bases**

We will **not** study the problem of answering FOL queries over DL KBs

Corresponds to general relational calculus queries

Leads to an undecidable decision problem.

# Conjunctive Queries — Definition

## Definition (Conjunctive query)

Let  $\mathbf{V}$  be a set of *variables*. A *term*  $t$  is a variable from  $\mathbf{V}$  or an individual name from  $\mathbf{I}$ .

A *conjunctive query* (CQ)  $q$  has the form  $\exists x_1 \dots \exists x_k (\alpha_1 \wedge \dots \wedge \alpha_n)$

- $k \geq 0$ ,  $n \geq 1$ ,  $x_1, \dots, x_k \in \mathbf{V}$
- each  $\alpha_j$  is a **concept atom**  $A(t)$  or a **role atom**  $r(t, t')$  with  $A \in \mathbf{C}$ ,  $r \in \mathbf{R}$ , and  $t, t'$  **terms**
- $x_1, \dots, x_k$  called **quantified variables**; all other variables in  $q$  called **answer variables**
- the **arity** of  $q$  is the number of answer variables
- $q$  is called **Boolean** if it has arity zero

To indicate that the answer variables in a CQ  $q$  are  $\vec{x}$ , we often write  $q(\vec{x})$  instead of just  $q$ .

## Example Conjunctive Queries

- 1 Return all pairs of individual names  $(a, b)$  such that  $a$  is a professor who supervises student  $b$ :

$$q_1(x_1, x_2) = \text{Professor}(\underline{x_1}) \wedge \text{supervises}(\underline{x_1}, \underline{x_2}) \wedge \text{Student}(\underline{x_2}).$$

- 2 Return all individual names  $a$  such that  $a$  is a student supervised by some professor:

$$q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x})).$$

- 3 Return all pairs of students supervised by the same professor:

$$q_3(x_1, x_2) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x_1}) \wedge \text{supervises}(y, \underline{x_2}) \wedge \text{Student}(\underline{x_1}) \wedge \text{Student}(\underline{x_2})).$$

- 4 Return all students supervised by professor smith (an individual name):

$$q_4(x) = \text{supervises}(\text{smith}, \underline{x}) \wedge \text{Student}(\underline{x}).$$



# Answers on an Interpretation

We first define query answers on a given interpretation  $\mathcal{I}$ .

## Definition

Let  $q$  be a conjunctive query and  $\mathcal{I}$  an interpretation. We use  $\text{term}(q)$  to denote the terms in  $q$ .

A **match of  $q$  in  $\mathcal{I}$**  is a mapping  $\pi : \text{term}(q) \rightarrow \Delta^{\mathcal{I}}$  such that

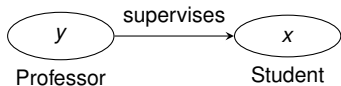
- $\pi(a) = a^{\mathcal{I}}$  for all  $a \in \text{term}(q) \cap \mathbf{I}$ ,
- $\pi(t) \in A^{\mathcal{I}}$  for all concept atoms  $A(t)$  in  $q$ , and
- $(\pi(t_1), \pi(t_2)) \in r^{\mathcal{I}}$  for all role atoms  $r(t_1, t_2)$  in  $q$ .

Let  $\vec{x} = x_1, \dots, x_k$  be the answer variables in  $q$  and  $\vec{a} = a_1, \dots, a_k$  be individual names from  $\mathbf{I}$ . We call the match  $\pi$  of  $q$  in  $\mathcal{I}$  an  $\vec{a}$ -match if  $\pi(x_i) = a_i^{\mathcal{I}}$  for  $1 \leq i \leq k$ .

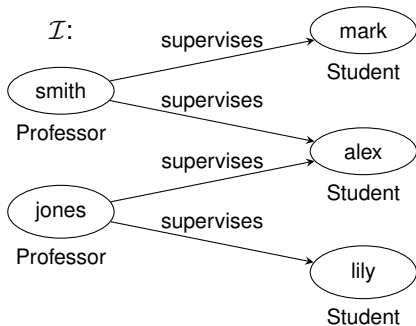
We say that  $\vec{a}$  is an **answer to  $q$  on  $\mathcal{I}$**  if there is an  $\vec{a}$ -match  $\pi$  of  $q$  in  $\mathcal{I}$ . We use  $\text{ans}(q, \mathcal{I})$  to denote the set of all answers to  $q$  on  $\mathcal{I}$ .

# Answers on Interpretation $\mathcal{I}$

$q_2(x)$ :



$\mathcal{I}$ :

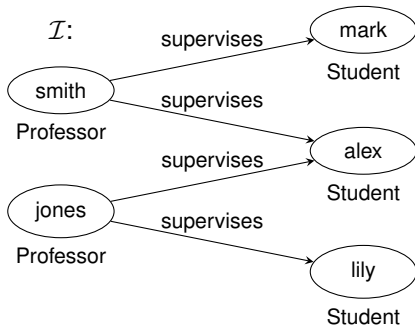
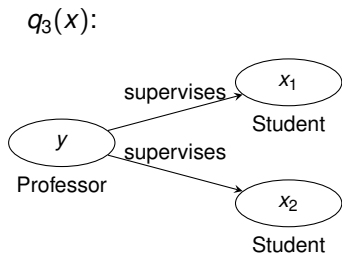


$$q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x}))$$

- There are 3 answers to  $q_2(x)$  on  $\mathcal{I}$ : mark, alex, and lily.

Note that a match is a **homomorphism** from the query to the interpretation (both viewed as a graphs).

# Answers on Interpretation $\mathcal{I}$



$$q_3(x_1, x_2) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x_1}) \wedge \text{supervises}(y, \underline{x_2}) \wedge \text{Student}(\underline{x_1}) \wedge \text{Student}(\underline{x_2})).$$

- There are 7 answers to  $q_3(x_1, x_2)$  on  $\mathcal{I}$ , including (mark, alex), (alex, lily), (lily, alex) and (mark, mark).

Note that a match need not be injective, e.g., (mark, mark).

# Certain Answers

Usually interested in answers on a KB, which may have many models.  
In this case so-called **certain answers** provide a natural semantics.

## Definition

Let  $q$  be a CQ and  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$  be a KB.

We say that  $\vec{a}$  is a **certain answer to  $q$  on  $\mathcal{K}$**  if

- all individual names from  $\vec{a}$  occur in  $\mathcal{A}$
- $\vec{a} \in \text{ans}(q, \mathcal{I})$  for every model  $\mathcal{I}$  of  $\mathcal{K}$

We use  $\text{cert}(q, \mathcal{K})$  to denote the set of all certain answers to  $q$  on  $\mathcal{K}$ :

$$\text{cert}(q, \mathcal{K}) = \bigcap_{\mathcal{I} \models \mathcal{K}} \text{ans}(q, \mathcal{I})$$

## Certain Answers

E.g., consider the  $\mathcal{ALCI}$  KB  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ :

$$\mathcal{T} = \{\text{Student} \sqsubseteq \exists \text{supervises}^{-} . \text{Professor}\},$$

$$\mathcal{A} = \{\text{smith} : \text{Professor}, \text{mark} : \text{Student}, \text{alex} : \text{Student}, \text{lily} : \text{Student}, \\ (\text{smith}, \text{mark}) : \text{supervises}, (\text{smith}, \text{alex}) : \text{supervises}\}.$$

■  $q_4(x) = \text{supervises}(\text{smith}, \underline{x}) \wedge \text{Student}(\underline{x});$

■  $q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x}));$

■  $q_1(x_1, x_2) = \text{Professor}(\underline{x}_1) \wedge \text{supervises}(\underline{x}_1, \underline{x}_2) \wedge \text{Student}(\underline{x}_2);$

## Certain Answers

E.g., consider the  $\mathcal{ALCI}$  KB  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ :

$\mathcal{T} = \{\text{Student} \sqsubseteq \exists \text{supervises}^- . \text{Professor}\},$

$\mathcal{A} = \{\text{smith} : \text{Professor}, \text{mark} : \text{Student}, \text{alex} : \text{Student}, \text{lily} : \text{Student},$   
 $(\text{smith}, \text{mark}) : \text{supervises}, (\text{smith}, \text{alex}) : \text{supervises}\}.$

- $q_4(x) = \text{supervises}(\text{smith}, \underline{x}) \wedge \text{Student}(\underline{x});$   
 $\text{cert}(q_4, \mathcal{K}) = \{\text{mark}, \text{alex}\}$ : there are models of  $\mathcal{K}$  in which smith supervises other students, but only mark and alex are supervised by smith in *all* models.
- $q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x}));$
- $q_1(x_1, x_2) = \text{Professor}(\underline{x}_1) \wedge \text{supervises}(\underline{x}_1, \underline{x}_2) \wedge \text{Student}(\underline{x}_2);$

# Certain Answers

E.g., consider the  $\mathcal{ALCI}$  KB  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ :

$$\mathcal{T} = \{\text{Student} \sqsubseteq \exists \text{supervises}^{-} . \text{Professor}\},$$

$$\mathcal{A} = \{\text{smith} : \text{Professor}, \text{mark} : \text{Student}, \text{alex} : \text{Student}, \text{lily} : \text{Student}, \\ (\text{smith}, \text{mark}) : \text{supervises}, (\text{smith}, \text{alex}) : \text{supervises}\}.$$

- $q_4(x) = \text{supervises}(\text{smith}, \underline{x}) \wedge \text{Student}(\underline{x})$ ;  
 $\text{cert}(q_4, \mathcal{K}) = \{\text{mark}, \text{alex}\}$ : there are models of  $\mathcal{K}$  in which smith supervises other students, but only mark and alex are supervised by smith in *all* models.
- $q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x}))$ ;  
 $\text{cert}(q_2, \mathcal{K}) = \{\text{mark}, \text{alex}, \text{lily}\}$ : note that lily is included because she is a student and thus the TBox enforces that she has a supervisor who is a professor in every model of  $\mathcal{K}$ .
- $q_1(x_1, x_2) = \text{Professor}(\underline{x}_1) \wedge \text{supervises}(\underline{x}_1, \underline{x}_2) \wedge \text{Student}(\underline{x}_2)$ ;

# Certain Answers

E.g., consider the  $\mathcal{ALCI}$  KB  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ :

$\mathcal{T} = \{\text{Student} \sqsubseteq \exists \text{supervises}^{-} . \text{Professor}\},$

$\mathcal{A} = \{\text{smith} : \text{Professor}, \text{mark} : \text{Student}, \text{alex} : \text{Student}, \text{lily} : \text{Student},$   
 $(\text{smith}, \text{mark}) : \text{supervises}, (\text{smith}, \text{alex}) : \text{supervises}\}.$

- $q_4(x) = \text{supervises}(\text{smith}, \underline{x}) \wedge \text{Student}(\underline{x});$   
 $\text{cert}(q_4, \mathcal{K}) = \{\text{mark}, \text{alex}\}$ : there are models of  $\mathcal{K}$  in which smith supervises other students, but only mark and alex are supervised by smith in *all* models.
- $q_2(x) = \exists y (\text{Professor}(y) \wedge \text{supervises}(y, \underline{x}) \wedge \text{Student}(\underline{x}));$   
 $\text{cert}(q_2, \mathcal{K}) = \{\text{mark}, \text{alex}, \text{lily}\}$ : note that lily is included because she is a student and thus the TBox enforces that she has a supervisor who is a professor in every model of  $\mathcal{K}$ .
- $q_1(x_1, x_2) = \text{Professor}(\underline{x}_1) \wedge \text{supervises}(\underline{x}_1, \underline{x}_2) \wedge \text{Student}(\underline{x}_2);$   
 $\text{cert}(q_1, \mathcal{K}) = \{(\text{smith}, \text{mark}), (\text{smith}, \text{alex})\}$ : lily always has a supervisor, but there is no supervisor on which all models agree.



# Boolean Conjunctive Query Answering

(arbitrary) CQ answering reduces to Boolean CQ answering

Given query  $q$  of arity  $n$  and  $\mathcal{K} = (\mathcal{T}, \mathcal{A})$  in which  $m$  individual names occur

- Iterate through  $m^n$  tuples of arity  $n$
- For each tuple  $\vec{a} = (a_1, \dots, a_n)$  create a Boolean query  $q_{\vec{a}}$  by replacing the  $i$ th answer variable with  $a_i$
- $\vec{a} \in \text{cert}(q, \mathcal{K})$  iff  $\mathcal{K} \models q_{\vec{a}}$

## Boolean Conjunctive Query Entailment:

An instance is a pair  $\langle \mathcal{K}, q \rangle$  with  $\mathcal{K}$  a KB and  $q$  a Boolean CQ. The answer is **true** iff  $\mathcal{I} \models q$  for each  $\mathcal{I} \models \mathcal{K}$ .

This problem is **not trivially reducible** to knowledge base consistency

It is EXPTIME-complete for  $\mathcal{ALC}$ , the same as consistency (proof beyond this course)

# Boolean Conjunctive Query Answering

Many types of query **can** be reduced to KB consistency:

- Concept and role instance queries, e.g.,  $q() = C(a)$  and  $q() = r(a, b)$
- Fully ground queries, e.g.,  $q() = C(a) \wedge D(b) \wedge r(a, b)$  — check each atom independently
- Forest shaped queries, e.g.,  $q() = \exists x(C(a) \wedge D(x) \wedge r(a, x))$  — roll up tree parts of query

Reduction may or may not be possible in general (possible for *SHIQ*; **open problem for *SHOIQ***).

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query?  $(\mathcal{K} = (\mathcal{T}, \mathcal{A}))$

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}.\textit{Child} \sqcap \forall \textit{Affects}.\textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}.\textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$\mathcal{A} \models q_1$  Yes

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query?  $(\mathcal{K} = (\mathcal{T}, \mathcal{A}))$

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}. \textit{Child} \sqcap \forall \textit{Affects}. \textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}. \textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query? ( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}.\textit{Child} \sqcap \forall \textit{Affects}.\textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}.\textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{K} \models q_2$  Yes

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query? ( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}.\textit{Child} \sqcap \forall \textit{Affects}.\textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}.\textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{K} \models q_2$  Yes

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query? ( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}.\textit{Child} \sqcap \forall \textit{Affects}.\textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}.\textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{K} \models q_2$  Yes

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{K} \models \neg q_3$  No

# Conjunctive Query Answering

How to interpret the answer to a Boolean Query? ( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}.\textit{Child} \sqcap \forall \textit{Affects}.\textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}.\textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{K} \models q_2$  Yes

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{K} \models \neg q_3$  No

$\mathcal{A} \not\models q_4, \mathcal{A} \not\models \neg q_4$  ???



# Conjunctive Query Answering

How to interpret the answer to a Boolean Query? ( $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ )

ABox  $\mathcal{A}$ :

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

TBox  $\mathcal{T}$ :

JuvDis  $\sqsubseteq \exists \textit{Affects}. \textit{Child} \sqcap \forall \textit{Affects}. \textit{Child}$

Adult  $\sqsubseteq \neg \textit{Child}$

Arth  $\sqsubseteq \exists \textit{Damages}. \textit{Joint}$

JuvArth  $\sqsubseteq \textit{Arth} \sqcap \textit{JuvDis}$

$q_1 = \textit{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \textit{Child}(\text{Mary})$

$q_3 = \textit{Adult}(\text{Mary})$

$q_4 = \exists y (\textit{Damages}(\text{JRA}, y) \wedge \textit{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{K} \models q_2$  Yes

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{K} \models \neg q_3$  No

$\mathcal{A} \not\models q_4, \mathcal{A} \not\models \neg q_4$  ???

$\mathcal{K} \not\models q_4, \mathcal{K} \not\models \neg q_4$  ???

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox $\mathcal{A}$	Database $\mathcal{D}$									
(JRA, John) : <i>Affects</i>	<table><tr><td><i>Affects</i></td><td></td><td>JuvArthritis</td></tr><tr><td>JRA</td><td>John</td><td>JRA</td></tr><tr><td>JRA</td><td>Mary</td><td></td></tr></table>	<i>Affects</i>		JuvArthritis	JRA	John	JRA	JRA	Mary	
<i>Affects</i>			JuvArthritis							
JRA		John	JRA							
JRA	Mary									
JRA : JuvArth										
(JRA, Mary) : <i>Affects</i>										

$\mathcal{A} \models q_1$  Yes

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	

JuvArthritis
JRA

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	
		JuvArthritis
		JRA

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	
		JuvArthritis
		JRA

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{D} \not\models q_2$  No

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		JuvArthritis	
JRA	John	JRA	
JRA	Mary		

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{D} \not\models q_2$  No

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	

JuvArthritis
JRA

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{D} \not\models q_2$  No

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{D} \not\models q_3$  No

# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	
		JuvArthritis
		JRA

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{D} \not\models q_2$  No

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{D} \not\models q_3$  No

$\mathcal{A} \not\models q_4, \mathcal{A} \not\models \neg q_4$  ???



# Conjunctive Query Answering

$\mathcal{A}$  is seen as a FOL knowledge base, but  $\mathcal{D}$  is seen as a FOL model

ABox  $\mathcal{A}$

(JRA, John) : *Affects*

JRA : JuvArth

(JRA, Mary) : *Affects*

Database  $\mathcal{D}$

<i>Affects</i>		
JRA	John	
JRA	Mary	

JuvArthritis
JRA

$q_1 = \text{Affects}(\text{JRA}, \text{Mary})$

$q_2 = \text{Child}(\text{Mary})$

$q_3 = \text{Adult}(\text{Mary})$

$q_4 = \exists y (\text{Damages}(\text{JRA}, y) \wedge \text{Organ}(y))$

$\mathcal{A} \models q_1$  Yes

$\mathcal{D} \models q_1$  Yes

$\mathcal{A} \not\models q_2, \mathcal{A} \not\models \neg q_2$  ???

$\mathcal{D} \not\models q_2$  No

$\mathcal{A} \not\models q_3, \mathcal{A} \not\models \neg q_3$  ???

$\mathcal{D} \not\models q_3$  No

$\mathcal{A} \not\models q_4, \mathcal{A} \not\models \neg q_4$  ???

$\mathcal{D} \not\models q_4$  No

# Ontologies vs Database Systems

## Conceptual DB-Schema:

- Typically formulated as an ER or UML diagram
- Used in DB design
- Schema leads to a set of FOL-based constraints
- Constraints are used to check conformance of the data
- Constraints are disregarded for query answering

In databases, query answering is a FOL model checking problem

## Description Logics TBoxes:

- Formulated in a Description Logic (fragment of FOL)
- TBox axioms are used to check conformance of the data  
The way this is done differs from DBs
- TBox axioms participate in query answering

In description logics, query answering is a FOL entailment problem

# KB Consistency: Practicality Issues

- Addition of ABox may greatly exacerbate practicality problems
  - No obvious limit to size of data—could be millions or even billions of individuals
  - Tableau algorithm applied to whole ABox
- Optimisations can ameliorate but not eliminate problem
- Can exploit **decomposition** of an ABox:
  - $\mathcal{A}$  can be decomposed into a set of disjoint connected components  $\{\mathcal{A}_1, \dots, \mathcal{A}_n\}$  s.t.:

$$\begin{aligned}\mathcal{A} &= \mathcal{A}_1 \cup \dots \cup \mathcal{A}_n \\ \forall_{1 \leq i < j \leq n} \text{ind}(\mathcal{A}_i) \cap \text{ind}(\mathcal{A}_j) &= \emptyset\end{aligned}$$

where  $\text{ind}(\mathcal{A}_i)$  is individuals (constants) occurring in  $\mathcal{A}_i$

- An *ALC* KB  $(\mathcal{T}, \mathcal{A})$  is consistent iff  $(\mathcal{T}, \mathcal{A}_i)$  is consistent for each  $\mathcal{A}_i$  in a decomposition  $\{\mathcal{A}_1, \dots, \mathcal{A}_n\}$  of  $\mathcal{A}$

# ABox Decomposition

E.g., given KB':

JRA : JuvArth

(JRA, Mary) : *Affects*

(John, Mary) : *hasChild*

(Paul, Miranda) : *hasChild*

Paul : Adult

JuvDis  $\sqsubseteq \exists Affects.Child \sqcap \forall Affects.Child$

$\exists hasChild.T \sqsubseteq Adult$

Adult  $\sqsubseteq \neg Child$

Arth  $\sqsubseteq \exists Damages.Joint$

JuvArth  $\sqsubseteq Arth \sqcap JuvDis$

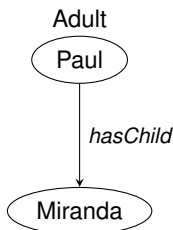
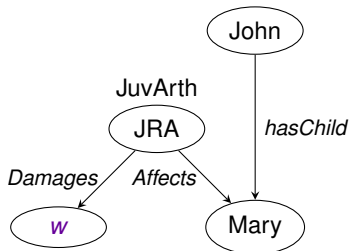
# ABox Decomposition

E.g., given KB':

JRA : JuvArth  
(JRA, Mary) : *Affects*  
(John, Mary) : *hasChild*  
(Paul, Miranda) : *hasChild*  
Paul : Adult

$JuvDis \sqsubseteq \exists Affects.Child \sqcap \forall Affects.Child$   
 $\exists hasChild.T \sqsubseteq Adult$   
 $Adult \sqsubseteq \neg Child$   
 $Arth \sqsubseteq \exists Damages.Joint$   
 $JuvArth \sqsubseteq Arth \sqcap JuvDis$

Perform separate consistency tests on the disjoint connected components:



# Query Answering: Practicality Issues

- Recall our example query

$$q(y) = \exists x \exists z (Affects(x, y) \wedge Affects(x, z) \wedge hasFriend(y, z))$$

- To answer this query we have to:

- check for each individual  $a$  occurring in  $\mathcal{A}$  if  $(\mathcal{T}, \mathcal{A}) \models q_{[y/a]}$ , where  $q_{[y/a]}$  is the Boolean CQ

$$q() = \exists x \exists z (Affects(x, a) \wedge Affects(x, z) \wedge hasFriend(a, z))$$

- checking  $(\mathcal{T}, \mathcal{A}) \models q_{[y/a]}$  involves performing (possibly many) consistency tests
- each test could be very costly
- And what if we change the query to

$$q(x, y, z) = Affects(x, y) \wedge Affects(x, z) \wedge hasFriend(y, z)?$$

- In general, there are  $n^m$  “candidate” answer tuples, where  $n$  is the number of individuals occurring in  $\mathcal{A}$  and  $m$  the arity of the query

# Optimised Query Answering

Many optimisations are possible, e.g.:

- Exploit the fact that we can't entail ABox roles in  $\mathcal{ALC}$ , i.e.:

$$(\mathcal{T}, \mathcal{A}) \models R(a, b) \text{ iff } R(a, b) \in \mathcal{A}$$

- Only check candidate tuples with relevant relational structure
- E.g., for

$$q(y, z) = \exists x(\text{JuvArth}(x) \wedge \text{Affects}(x, y) \wedge \text{hasFriend}(y, z))$$

only check tuples  $(a, b)$  s.t.:

$$\text{hasFriend}(a, b) \in \mathcal{A}$$

and for these only need to check Boolean CQ:

$$\exists x.(\text{JuvArth}(x) \wedge \text{Affects}(x, a) \wedge \text{Affects}(x, b))$$

# Conflicting Requirements

Ontology-based data access applications require:

- 1 Very expressive ontology languages  
As large fragment of FOL as possible
- 2 Powerful query languages  
As large fragment of SQL as possible
- 3 Efficient query answering algorithms  
Low complexity, easy to optimise

The requirements are in conflict!

↪ We need to make compromises.