Ontologies & Description Logics

M2 KNOWLEDGE REPRESENTATION

Meghyn Bienvenu (LaBRI - *CNRS & Université de Bordeaux*)

INTRODUCTION TO ONTOLOGIES

WHAT IS AN ONTOLOGY?

An ontology is a formal specification of the knowledge of a particular domain, making it amenable to machine processing

WHAT IS AN ONTOLOGY?

An ontology is a formal specification of the knowledge of a particular domain, making it amenable to machine processing

Such a specification consists of:

- terminology (or vocabulary) of the domain
- semantic relationships between terms
  - relations of specificity or generality, equivalence, disjointness, …

WHAT IS AN ONTOLOGY?

An ontology is a **formal specification of the knowledge of a particular domain**, making it **amenable to machine processing**

Such a specification consists of:

- **terminology** (or vocabulary) of the domain
- **semantic relationships** between terms
  - relations of specificity or generality, equivalence, disjointness, …

For example, in the university domain:

- terms: student, PhD student, professor, course, teaches, takes, supervises
- relationships between terms:
  - every PhD student is a student
  - cannot be both student and professor
  - every course is taught by some professor

WHY USE ONTOLOGIES?

To standardize the terminology of an application domain

- meaning of terms is constrained, so less misunderstandings
- by adopting a common vocabulary, easy to share information

WHY USE ONTOLOGIES?

To **standardize the terminology** of an application domain

- meaning of terms is constrained, so less misunderstandings
- by adopting a common vocabulary, easy to share information

To present an **intuitive and unified view of data**

- ontology can be used to **enrich the data vocabulary**, making it easier for users to formulate their queries

WHY USE ONTOLOGIES?

To standardize the terminology of an application domain

- meaning of terms is constrained, so less misunderstandings
- by adopting a common vocabulary, easy to share information

To present an intuitive and unified view of data

- ontology can be used to enrich the data vocabulary, making it easier for users to formulate their queries
- especially useful when integrating multiple data sources

WHY USE ONTOLOGIES?

To standardize the terminology of an application domain

- meaning of terms is constrained, so less misunderstandings
- by adopting a common vocabulary, easy to share information

To present an intuitive and unified view of data

- ontology can be used to enrich the data vocabulary, making it easier for users to formulate their queries
- especially useful when integrating multiple data sources

To support automated reasoning

- uncover errors in modelling (debugging)

WHY USE ONTOLOGIES?

To **standardize the terminology** of an application domain

- meaning of terms is constrained, so less misunderstandings
- by adopting a common vocabulary, easy to share information

To present an **intuitive and unified view of data**

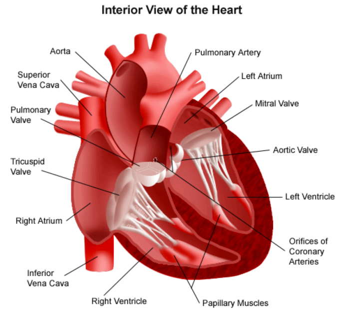
- ontology can be used to **enrich the data vocabulary**, making it easier for users to formulate their queries
- especially useful when **integrating multiple data sources**

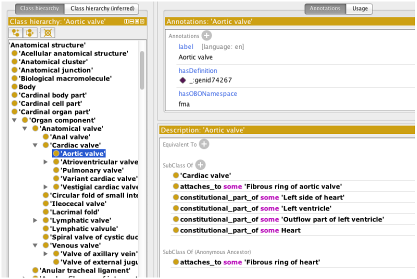
To support **automated reasoning**

- uncover errors in modelling (debugging)
- **exploit knowledge in the ontology during query answering**, to get back a **more complete set of answers** to queries

APPLICATIONS OF ONTOLOGIES: MEDICINE

General medical ontologies: **SNOMED CT**, GALEN
Specialized ontologies: FMA (anatomy), NCI (cancer), …





Querying & exchanging medical records (find patients for medical trials)

Supports analysis of health data for medical research

ZOOM ON: SNOMED CT

Large-scale comprehensive medical ontology

- more than 350,000 terms on all aspects of clinical healthcare:
  - symptoms, diagnosis, medical procedures, body structures, organisms, substances, pharmaceutical products, etc.



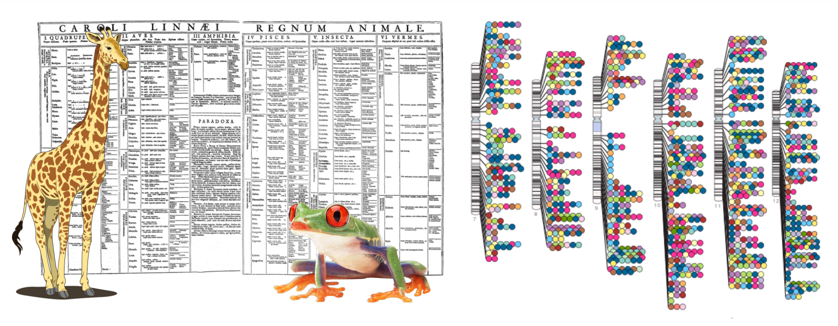


Widely adopted standard for healthcare terminology

- in use in > 80 countries (including U.S., Canada, UK)
- utilized in health IT (e.g. IBM Watson Health, Babylon Health)

APPLICATIONS OF ONTOLOGIES: LIFE SCIENCES

Hundreds of ontologies at BioPortal (http://bioportal.bioontology.org/):
Gene Ontology (GO), Cell Ontology, Pathway Ontology, Plant Anatomy, ...

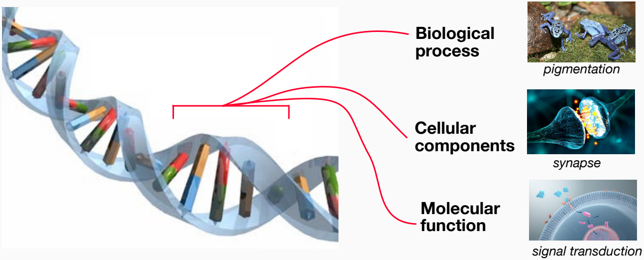


Help scientists share, query, & visualize experimental data

ZOOM ON: GENE ONTOLOGY

Aim: rigorous shared vocabulary to describe the roles of genes across different organisms

Annotations: evidence-based statements relating specific gene product to specific ontology terms



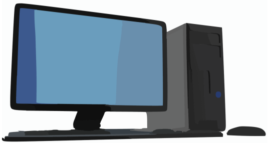
Very successful endeavour:

- > 100K published scientific articles with keyword “Gene Ontology”
- > 700K experimentally-supported annotations

APPLICATIONS OF OMQA: ENTREPRISE INFORMATION SYSTEMS

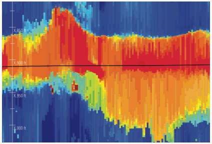
Companies and organizations have lots of data

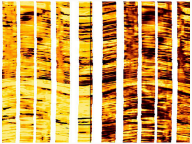
- need easy and flexible access to support decision-making



ZOOM ON: STATOIL USECASE

Goal: aid geologists in gathering information for analysis





Difficulties:

- relevant data stored across > 3000 database tables
  - geologist question = complex query (thousands of terms, 50-200 joins)
- must ask IT staff, may take several days to get answer

Solution:

- ontology provides familiar vocabulary for query formulation
- mappings link database tables to ontology terms
- use reasoning to automatically transform ontology query into executable database query

ONTOLOGY LANGUAGES

Which languages can we use to specify an ontology?

Natural language?

- imprecise, ambiguous, not machine-interpretable

Formal logic? (long studied in math, CS, philosophy)

- **first-order logic (FOL)**
  - expressive, well understood, BUT reasoning undecidable
- **'nice' fragments of FOL** (description logics, existential rules)
  - expressivity vs computational complexity tradeoff

Standardized languages for the Web (OWL, RDFS)

- closely related and based upon logic
- geared to broader public, offer various formats

DESCRIPTION LOGICS

DESCRIPTION LOGICS

Description logics (DLs) are:

- family of knowledge representation languages
- popular means for specifying ontologies
- range from fairly simple to highly expressive

Description logics (DLs) are:

- family of knowledge representation languages
- popular means for specifying ontologies
- range from fairly simple to highly expressive

Formally: correspond to **decidable fragments of first-order logic**

- inherit well-defined semantics
- convenient variable-free syntax
- modelling via unary and binary relations

Description logics (DLs) are:

- family of knowledge representation languages
- popular means for specifying ontologies
- range from fairly simple to highly expressive

Formally: correspond to decidable fragments of first-order logic

- inherit well-defined semantics
- convenient variable-free syntax
- modelling via unary and binary relations

Computational properties well understood (decidability, complexity)

DESCRIPTION LOGICS

Description logics (DLs) are:

- family of knowledge representation languages
- popular means for specifying ontologies
- range from fairly simple to highly expressive

Formally: correspond to decidable fragments of first-order logic

- inherit well-defined semantics
- convenient variable-free syntax
- modelling via unary and binary relations

Computational properties well understood (decidability, complexity)

Many implemented reasoners and tools available for use

BASIC BUILDING BLOCKS

Concept names

- correspond to sets of entities
- unary predicates in logic

Parent Scientist FrenchPastry CapitalCity Whale

BASIC BUILDING BLOCKS

Concept names

- correspond to sets of entities
- unary predicates in logic

Parent	Scientist	FrenchPastry	CapitalCity	Whale

Role names

- relate two entities (binary relation)
- binary predicates in logic

childOf	teaches	ingredientOf	locatedIn	hasHabitat

BASIC BUILDING BLOCKS

Concept names

- correspond to sets of entities
- unary predicates in logic

Parent   Scientist   FrenchPastry   CapitalCity   Whale

Role names

- relate two entities (binary relation)
- binary predicates in logic

childOf   teaches   ingredientOf   locatedIn   hasHabitat

Individual names

- denote particular entities (constants in logic)

kim   csc415   univBordeaux   tramStopPeixotto   panda35

CONSTRUCTORS TO BUILD COMPLEX DESCRIPTIONS

Combine concepts/classes and roles/properties using constructors to build complex descriptions

Example: courses taught by Meghyn which are attended by at least two students not from computer science and only by Master's or Phd students

$$Course \sqcap \exists taughtBy.\{meghyn\} \sqcap \geq 2 attendedBy.(\neg CompSciStudent)$$

$$\forall attendedBy.(MasterStudent \sqcup PhDStudent)$$

Next slides: introduce some of the most common constructors

Notation: $C \sqcap D$ (C, D potentially complex concepts)

Meaning: class of entities that belong to both C and D

Female scientist

$\text{Female} \sqcap \text{Scientist}$

Spicy vegetarian dish

$\text{Spicy} \sqcap \text{VegetarianDish}$

DISJUNCTION

Notation: $C \sqcup D$

Meaning: class of entities that belong to **either C or D (possibly both)**

Cat or dog

$$\mathrm{Cat} \sqcup \mathrm{Dog}$$

Cake or cookie or muffin

$$\mathrm{Cake} \sqcup (\mathrm{Cookie} \sqcup \mathrm{Muffin})$$

NEGATION

Notation: $\neg C$

Meaning: class of entities that are **not in** C

Anything/anyone who is **not a parent**

$$\neg \text{ Parent}$$

Person who is not a parent

$$\text{Person} \sqcap (\neg \text{ Parent})$$

EXISTENTIAL RESTRICTIONS

Notation: $\exists r.C$ (r possibly complex role, C possibly complex concept)

Meaning: class of entities that **are related by r to an entity in C**

Those who teach a doctoral course offered by a French university

$$\exists \text{teaches}.(\text{DoctoralCourse} \sqcap \exists\, \text{offeredBy}.\text{FrenchUniv})$$

Those who teach (something)

$$\exists \text{teaches}.\top$$

Here $\top$ designates the set of all things

UNIVERSAL RESTRICTIONS

Notation: $\forall r.C$ (r possibly complex role, C possibly complex concept)

Meaning: class of entities that are only related by r to entities from C

Those who teach only doctoral courses

$$\forall \text{teaches}.\text{DoctoralCourse}$$

UNIVERSAL RESTRICTIONS

Notation: $\forall r.C$ (r possibly complex role, C possibly complex concept)

Meaning: class of entities that are **only related by r to entities from C**

Those who teach only doctoral courses

$$\forall \text{teaches}.\text{DoctoralCourse}$$

Question: Do they necessarily teach something?

UNIVERSAL RESTRICTIONS

Notation: $\forall r.C$ (r possibly complex role, C possibly complex concept)

Meaning: class of entities that are **only related by r to entities from C**

Those who teach only doctoral courses

$$\forall \text{teaches}.\text{DoctoralCourse}$$

Question: Do they necessarily teach something?

NO: **those who don't teach trivially belong to the class**

UNIVERSAL RESTRICTIONS

Notation: $\forall r.C$ (r possibly complex role, C possibly complex concept)

Meaning: class of entities that are **only related by r to entities from C**

Those who teach only doctoral courses

$$\forall \text{teaches}.\text{DoctoralCourse}$$

Question: Do they necessarily teach something?

NO: **those who don't teach trivially belong to the class**

If we only want to include those who teach, need to specify this:

$$(\forall \text{teaches}.\text{DoctoralCourse}) \sqcap (\exists \text{teaches}.\top)$$

NUMBER RESTRICTIONS

Notation: $\geq k r.C, \leq k r.C$ (r role, C concept, k non-negative integer)

Meaning: class of entities that **are connected via r to at least / at most k elements of C**

Those having at least 3 adult children

$$\geq 3\text{hasChild.Adult}$$

Things that have at most 5 ingredients

$$\leq 5\ \text{hasIngredient.}\top$$

Notation: $\{a\}$ (a an individual name)

Meaning: class consisting solely of a

Canada, USA, or Mexico

$$\{\text{canada}\} \sqcup \{\text{usa}\} \sqcup \{\text{mexico}\}$$

Courses taught by Meghyn

$$\text{Course} \sqcap \exists \text{taughtBy}.\{\text{meghyn}\}$$

So far we've seen how to describe classes and binary relationships

EXPRESSING KNOWLEDGE

So far we've seen how to describe classes and binary relationships

We can now use them to express different kinds of knowledge

Common to distinguish between two kinds of knowledge:

- general domain knowledge — TBox (ontology)
  - finite set of axioms (details on next slides)
- factual knowledge about particular individuals — ABox (data)
  - finite set of assertions $C(a), r(a, b)$ (C concept, r role, a, b inds)

EXPRESSING KNOWLEDGE

So far we've seen how to describe classes and binary relationships

We can now use them to express different kinds of knowledge

Common to distinguish between two kinds of knowledge:

- general domain knowledge — TBox (ontology)
  - finite set of axioms (details on next slides)
- factual knowledge about particular individuals — ABox (data)
  - finite set of assertions $C(a), r(a, b)$ (C concept, r role, a, b inds)

DL knowledge base (KB) = TBox (ontology) + ABox (data)

Note: usage varies, word "ontology" is sometimes used for whole KB

Suppose C, D are (possibly complex) concepts

(General) concept inclusion: $C \sqsubseteq D$

- every member of C is also a member of D (all Cs are Ds)

Concept equivalence: $C \equiv D$

- C and D describe precisely the same sets
- abbreviation for $C \sqsubseteq D$ **and** $D \sqsubseteq C$

EXAMPLES OF CONCEPT AXIOMS

Professors and **lecturers** are **disjoint classes of faculty**

$$\mathsf{Prof} \sqsubseteq \mathsf{Faculty} \quad \mathsf{Lect} \sqsubseteq \mathsf{Faculty} \quad \mathsf{Prof} \sqsubseteq \neg\mathsf{Lect}$$

Every **course** is **either an undergrad or grad course**

$$\mathsf{Course} \equiv \mathsf{UCourse} \sqcup \mathsf{GCourse}$$

The relation takesCourse **connects students to courses**

$$\exists \mathsf{takesCourse}.\top \sqsubseteq \mathsf{Student} \quad \exists \mathsf{takesCourse}^{-}.\top \sqsubseteq \mathsf{Course}$$

Note: speak of **domain** (1st position) / **range** (2nd position) of roles

MORE EXAMPLES OF CONCEPT AXIOMS

Every student takes **at least 2** and **at most 5 courses**

$$\mathsf{Student} \sqsubseteq\ \geq 2\mathsf{takesCourse}.\top \sqcap \leq 5\mathsf{takesCourse}.\top$$

Every **grad student** is **supervised by some faculty member**

$$\mathsf{GStudent} \sqsubseteq \exists\mathsf{supervisedBy}.\mathsf{Faculty}$$

Students who **only take grad-level courses** are **grad students**

$$\mathsf{Student} \sqcap \forall\mathsf{takesCourse}.\mathsf{GCourse} \sqsubseteq \mathsf{GStudent}$$

AXIOMS ABOUT ROLES

Axioms giving **relationship between r and s**

- r is contained in s (every pair in r also belongs to s)

$$r \sqsubseteq s \qquad \text{(role inclusion)}$$

- r and s are disjoint (no pairs in common)

$$r \sqsubseteq \neg s$$

- r corresponds to the inverse of s

$$r \equiv s^-$$

EXAMPLES OF ROLE AXIOMS

ParentOf and ChildOf are the inverses of one another

$$\mathsf{parentOf} \equiv \mathsf{childOf}^{-}$$

The relation parentOf is included in ancestorOf

$$\mathsf{parentOf} \sqsubseteq \mathsf{ancestorOf}$$

The friendOf and enemyOf relations are disjoint

$$\mathsf{friendOf} \sqsubseteq \neg\mathsf{enemyOf}$$

MORE AXIOMS ABOUT ROLES / PROPERTIES

Can also state that r is:

- transitive: if $r(x, y)$ and $r(y, z)$, then $r(x, z)$
- functional: if $r(x, y)$ and $r(x, z)$, then $y = z$
- symmetric: if $r(x, y)$, then $r(y, x)$
- reflexive: $r(x, x)$ (for any x)
- also: inverse functional, asymmetric, irreflexive

For example:

funct(hasBirthplace)   trans(locatedIn)

DEFINING A PARTICULAR DL

Lots of DLs: differ on which constructors and axioms allowed

DEFINING A PARTICULAR DL

Lots of DLs: differ on which constructors and axioms allowed

For example, the prototypical expressive DL $\mathcal{ALC}$ is defined by:

- concept constructors: $\bot, \top, \neg, \sqcup, \sqcap, \exists r.C, \forall r.C$
- no role constructors
- only concept inclusions as axioms

DEFINING A PARTICULAR DL

Lots of DLs: differ on which constructors and axioms allowed

For example, the prototypical expressive DL $\mathcal{ALC}$ is defined by:

- concept constructors: $\bot, \top, \neg, \sqcup, \sqcap, \exists r.C, \forall r.C$
- no role constructors
- only concept inclusions as axioms

The lightweight DL $\mathcal{EL}$ is a fragment of $\mathcal{ALC}$:

- concept constructors restricted to: $\top, \sqcap, \exists r.C$

DEFINING A PARTICULAR DL

Lots of DLs: differ on which constructors and axioms allowed

For example, the prototypical expressive DL $\mathcal{ALC}$ is defined by:

- concept constructors: $\bot, \top, \neg, \sqcup, \sqcap, \exists r.C, \forall r.C$
- no role constructors
- only concept inclusions as axioms

The lightweight DL $\mathcal{EL}$ is a fragment of $\mathcal{ALC}$:

- concept constructors restricted to: $\top, \sqcap, \exists r.C$

The highly expressive DL $\mathcal{SHIQ}$ is defined by:

- all $\mathcal{ALC}$ concept constructors, plus: $\geq n\, r.C, \leq n\, r.C$
- inverse roles (r^-)
- concept inclusions, role inclusions, and transitivity axioms

SEMANTICS: THE MEANING OF THINGS

Syntax tells us what are legal expressions in a language

So far we have introduced

- the syntax of DL concepts, roles, axioms, assertions

SEMANTICS: THE MEANING OF THINGS

Syntax tells us what are legal expressions in a language

So far we have introduced

- the syntax of DL concepts, roles, axioms, assertions

However, we need semantics to give the symbols meaning:

- do two concept expressions designate the same set?
- does a given axiom logically follow from a set of axioms?
- does our knowledge base contain contradictory information?

DL semantics: based upon interpretations (like first-order logic, FOL)

INTERPRETATIONS

Interpretation $\mathcal{I}$ takes the form of a pair $(\Delta^\mathcal{I}, \cdot^\mathcal{I})$

- $\Delta^\mathcal{I}$ is a non-empty set
  - (called the interpretation domain or universe)
- $\cdot^\mathcal{I}$ is a **function which maps**
  - individual name $a \mapsto$ **an element of the universe** $a^\mathcal{I} \in \Delta^\mathcal{I}$
  - concept name $A \mapsto$ **unary relation** $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$
  - role name $r \mapsto$ **binary relation** $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$

Each interpretation describes a particular state-of-affairs

Can think of interpretations as possible worlds

EXAMPLE INTERPRETATION

Interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where

$$\Delta^{\mathcal{I}} = \{e_1, e_2, \ldots, e_{15}\}$$

$$\mathit{Student}^{\mathcal{I}} = \{e_1, \ldots, e_7\}$$

$$\mathit{Musician}^{\mathcal{I}} = \{e_4, e_5, e_6, e_8, e_{10}, e_{13}\}$$

$$\begin{aligned} \mathit{supervises}^{\mathcal{I}} = \{&(e_5, e_2), (e_8, e_7), (e_{10}, e_4), (e_{10}, e_6), \\ &(e_9, e_3), (e_9, e_4), (e_{12}, e_{15})\} \end{aligned}$$

$$\mathit{Professor}^{\mathcal{I}} = \{e_8, \ldots, e_{15}\}$$

$$\mathit{Athlete}^{\mathcal{I}} = \{e_6, e_7, e_8\}$$

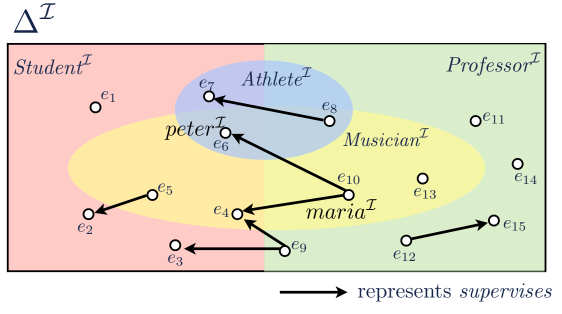
$$\mathit{maria}^{\mathcal{I}} = e_{10}$$

$$\mathit{peter}^{\mathcal{I}} = e_6$$

EXAMPLE INTERPRETATION

Interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where

$$\Delta^{\mathcal{I}} = \{e_1, e_2, \ldots, e_{15}\}$$
$$\mathit{Student}^{\mathcal{I}} = \{e_1, \ldots, e_7\}$$
$$\mathit{Musician}^{\mathcal{I}} = \{e_4, e_5, e_6, e_8, e_{10}, e_{13}\}$$
$$\mathit{supervises}^{\mathcal{I}} = \{(e_5, e_2), (e_8, e_7), (e_{10}, e_4), (e_{10}, e_6), (e_9, e_3), (e_9, e_4), (e_{12}, e_{15})\}$$
$$\mathit{Professor}^{\mathcal{I}} = \{e_8, \ldots, e_{15}\}$$
$$\mathit{Athlete}^{\mathcal{I}} = \{e_6, e_7, e_8\}$$
$$\mathit{maria}^{\mathcal{I}} = e_{10}$$
$$\mathit{peter}^{\mathcal{I}} = e_6$$



SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$

SEMANTICS OF CONSTRUCTORS

We next **extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles**, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u,v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $(\forall r.C)^{\mathcal{I}} = \{u \mid \text{for every } v \text{ such that } (u,v) \in r^{\mathcal{I}} \text{ we have } v \in C^{\mathcal{I}}\}$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $(\forall r.C)^{\mathcal{I}} = \{u \mid \text{for every } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ we have } v \in C^{\mathcal{I}}\}$
- $(\geq n\, r.C)^{\mathcal{I}} = \{u \mid \text{at least } n\ v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$

SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $(\forall r.C)^{\mathcal{I}} = \{u \mid \text{for every } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ we have } v \in C^{\mathcal{I}}\}$
- $(\geq n\, r.C)^{\mathcal{I}} = \{u \mid \text{at least } n\ v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$

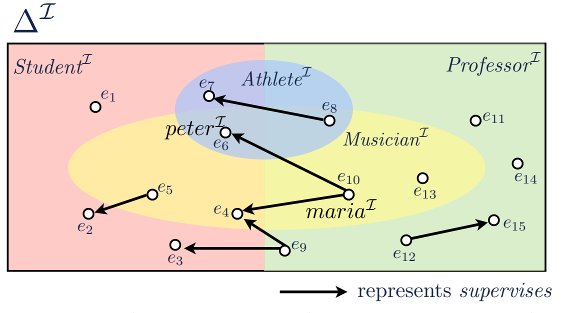
SEMANTICS OF CONSTRUCTORS

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $(\forall r.C)^{\mathcal{I}} = \{u \mid \text{for every } v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ we have } v \in C^{\mathcal{I}}\}$
- $(\geq n\, r.C)^{\mathcal{I}} = \{u \mid \text{at least } n\ v \text{ such that } (u, v) \in r^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$
- $r^{-} = \{(u, v) \mid (v, u) \in r^{\mathcal{I}}\}$

EXAMPLE: SEMANTICS OF CONSTRUCTORS

Reconsider the interpretation $\mathcal{I}$:



For each of the following concepts, give the corresponding set in $\mathcal{I}$:

(1) $\mathit{Athlete} \sqcup \mathit{Musician}$

(2) $\mathit{Athlete} \sqcap \neg\mathit{Musician}$

(3) $\exists \mathit{supervises}.\mathit{Student}$

(4) $\exists \mathit{supervises}^{-}.\mathit{Student}$

(5) $\geq 2\, \mathit{supervises}.\top$

(6) $\forall \mathit{supervises}.\mathit{Student}$

SEMANTICS OF AXIOMS & ASSERTIONS

Satisfaction of axioms:

- $\mathcal{I}$ satisfies a concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a concept equivalence $C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a role inclusion $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
- $\mathcal{I}$ satisfies $(\mathsf{trans}\, r)$ is $r^{\mathcal{I}}$ is a transitive relation, **and so on...**

SEMANTICS OF AXIOMS & ASSERTIONS

Satisfaction of axioms:

- $\mathcal{I}$ satisfies a concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a concept equivalence $C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a role inclusion $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
- $\mathcal{I}$ satisfies (trans r) is $r^{\mathcal{I}}$ is a transitive relation, **and so on...**

Satisfaction of ABox assertions:

- $\mathcal{I}$ satisfies an assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$ satisfies an assertion $r(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

SEMANTICS OF AXIOMS & ASSERTIONS

Satisfaction of axioms:

- $\mathcal{I}$ satisfies a concept inclusion $C \sqsubseteq D$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$
- $\mathcal{I}$ satisfies a concept equivalence $C \equiv D$ if $C^\mathcal{I} = D^\mathcal{I}$
- $\mathcal{I}$ satisfies a role inclusion $r \sqsubseteq s$ if $r^\mathcal{I} \subseteq s^\mathcal{I}$
- $\mathcal{I}$ satisfies $(\mathsf{trans}\, r)$ is $r^\mathcal{I}$ is a transitive relation, **and so on...**

Satisfaction of ABox assertions:

- $\mathcal{I}$ satisfies an assertion $C(a)$ if $a^\mathcal{I} \in C^\mathcal{I}$
- $\mathcal{I}$ satisfies an assertion $r(a, b)$ if $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$

Important: ABoxes are interpreted under **open-world assumption**

- provide **incomplete information**, $\alpha \notin \mathcal{A}$ does not mean α is false (might be able to infer it using axioms)

SEMANTICS OF AXIOMS & ASSERTIONS

Satisfaction of axioms:

- $\mathcal{I}$ satisfies a concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a concept equivalence $C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a role inclusion $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
- $\mathcal{I}$ satisfies (trans r) is $r^{\mathcal{I}}$ is a transitive relation, **and so on...**

Satisfaction of ABox assertions:

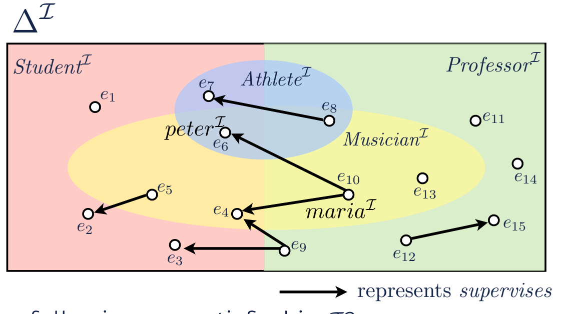
- $\mathcal{I}$ satisfies an assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$ satisfies an assertion $r(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

Important: ABoxes are interpreted under **open-world assumption**

- provide **incomplete information**, $\alpha \notin \mathcal{A}$ does not mean α is false (might be able to infer it using axioms)
- differs from closed-world assumption for databases (where absent means false)

EXAMPLE: SATISFACTION OF AXIOMS & ASSERTIONS

Reconsider the interpretation $\mathcal{I}$:



Which of the following are satisfied in $\mathcal{I}$?

(1)	$Athlete \sqsubseteq Musician$	(2)	$Student \sqsubseteq \neg Professor$
(3)	$\exists supervises.Athlete \sqsubseteq Professor$	(4)	$Professor \equiv \exists supervises.\top$
(5)	$\exists supervises^{-}.\top \sqsubseteq Student$	(6)	$Student \sqsubseteq \forall supervises.Student$
(7)	$Musician(peter)$	(8)	$(\exists supervises.Musician)(maria)$

MODELS, ENTAILMENT, SATISFIABILITY

Models:

- $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$
- $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies every assertion in $\mathcal{A}$
- $\mathcal{I}$ is a model of a KB $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$

MODELS, ENTAILMENT, SATISFIABILITY

Models:

- $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$
- $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies every assertion in $\mathcal{A}$
- $\mathcal{I}$ is a model of a KB $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$

Satisfiability:

- A concept C is satisfiable w.r.t. TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$
- A KB $(\mathcal{T}, \mathcal{A})$ is satisfiable if $(\mathcal{T}, \mathcal{A})$ has at least one model

MODELS, ENTAILMENT, SATISFIABILITY

Models:

- $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$
- $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies every assertion in $\mathcal{A}$
- $\mathcal{I}$ is a model of a KB $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$

Satisfiability:

- A concept C is satisfiable w.r.t. TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$
- A KB $(\mathcal{T}, \mathcal{A})$ is satisfiable if $(\mathcal{T}, \mathcal{A})$ has at least one model

Entailment:

- A TBox $\mathcal{T}$ entails an axiom α (written $\mathcal{T} \models \alpha$) if every model of $\mathcal{T}$ satisfies α
- A KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ entails an ABox assertion α (written $\mathcal{K} \models \alpha$) if every model of $(\mathcal{T}, \mathcal{A})$ satisfies α

EXAMPLES: ENTAILMENT, SATISFIABILITY

The axiom $\mathit{Cobra} \sqsubseteq \mathit{Animal}$ **is entailed** from the following TBox:

$\mathit{Cobra} \sqsubseteq \mathit{Snake}$

$\mathit{Snake} \sqsubseteq \mathit{Reptile}$

$\mathit{Reptile} \sqsubseteq \mathit{Animal}$

EXAMPLES: ENTAILMENT, SATISFIABILITY

The axiom $\mathit{Cobra} \sqsubseteq \mathit{Animal}$ **is entailed** from the following TBox:

$\mathit{Cobra} \sqsubseteq \mathit{Snake}$ $\quad$ $\mathit{Snake} \sqsubseteq \mathit{Reptile}$ $\quad$ $\mathit{Reptile} \sqsubseteq \mathit{Animal}$

The assertion $\mathit{TeachingStaff}(\mathit{rami})$ **is entailed** from the following KB:

$\mathit{teaches}(\mathit{rami}, \mathit{phy}305)$ $\quad$ $\exists \mathit{teaches}.\top \sqsubseteq \mathit{TeachingStaff}$

EXAMPLES: ENTAILMENT, SATISFIABILITY

The axiom $\mathit{Cobra} \sqsubseteq \mathit{Animal}$ **is entailed** from the following TBox:

$\mathit{Cobra} \sqsubseteq \mathit{Snake}$ $\mathit{Snake} \sqsubseteq \mathit{Reptile}$ $\mathit{Reptile} \sqsubseteq \mathit{Animal}$

The assertion $\mathit{TeachingStaff}(\mathit{rami})$ **is entailed** from the following KB:

$\mathit{teaches}(\mathit{rami}, \mathit{phy}305)$ $\exists \mathit{teaches}.\top \sqsubseteq \mathit{TeachingStaff}$

The following **KB is unsatisfiable**:

$\mathit{Childless}(\mathit{tom})$ $\mathit{hasChild}(\mathit{tom}, \mathit{fred})$ $\mathit{Childless} \equiv \forall \mathit{hasChild}.\bot$

EXAMPLE: ENTAILMENT

Suppose that the ontology $\mathcal{T}$ contains the following axioms:

$$
\begin{aligned}
\mathit{Sheep} &\sqsubseteq \mathit{Animal} \sqcap \forall \mathit{Eats}.\mathit{Grass} && (1)\\
\mathit{Grass} &\sqsubseteq \mathit{Plant} && (2)\\
\mathit{Vegetarian} &\equiv \mathit{Animal} \sqcap (\forall \mathit{Eats}.\neg \mathit{Animal}) && (3)\\
&\quad \sqcap (\forall \mathit{Eats}.\neg(\exists \mathit{PartOf}.\mathit{Animal}))\\
\mathit{Animal} \sqcup \exists \mathit{PartOf}.\mathit{Animal} &\sqsubseteq \neg(\mathit{Plant} \sqcup \exists \mathit{PartOf}.\mathit{Plant}) && (4)
\end{aligned}
$$

Claim: $\mathcal{T} \models \mathit{Sheep} \sqsubseteq \mathit{Vegetarian}$ **Why?**

(animal examples taken from:
`http://owl.man.ac.uk/2003/why/latest/`)

EXAMPLE: UNSATISFIABLE CONCEPT

Now suppose $\mathcal{T}$ contains the following axioms:

$$
\begin{align}
\mathit{Sheep} &\sqsubseteq \mathit{Animal} \sqcap \forall \mathit{eats}.\mathit{Grass} \tag{1}\\
\mathit{Cow} &\sqsubseteq \mathit{Vegetarian} \tag{2}\\
\mathit{MadCow} &\equiv \mathit{Cow} \sqcap \exists \mathit{eats}.(\mathit{Brain} \sqcap \exists \mathit{partOf}.\mathit{Sheep}) \tag{3}\\
\mathit{Vegetarian} &\equiv \mathit{Animal} \sqcap (\forall \mathit{eats}.\neg \mathit{Animal}) \tag{4}\\
&\quad \sqcap (\forall \mathit{eats}.\neg(\exists \mathit{partOf}.\mathit{Animal})) \nonumber\\
\mathit{Animal} \sqcup \exists \mathit{partOf}.\mathit{Animal} &\sqsubseteq \neg(\mathit{Plant} \sqcup \exists \mathit{partOf}.\mathit{Plant}) \tag{5}
\end{align}
$$

Is *MadCow* is satisfiable w.r.t. $\mathcal{T}$ **Why?**

REASONING TASKS

Concept satisfiability

- Input: (possibly complex) concept C, TBox $\mathcal{T}$
- Task: determine whether C is satisfiable w.r.t. $\mathcal{T}$
- Use: debugging ontologies

REASONING TASKS

Concept satisfiability

- Input: (possibly complex) concept C, TBox $\mathcal{T}$
- Task: **determine whether C is satisfiable w.r.t. $\mathcal{T}$**
- Use: debugging ontologies

Axiom Entailment

- Input: axiom α, TBox $\mathcal{T}$
- Task: **determine whether α is entailed by $\mathcal{T}$**
- Use: understanding content of the ontology

REASONING TASKS

Concept satisfiability

- Input: (possibly complex) concept C, TBox $\mathcal{T}$
- Task: determine whether C is satisfiable w.r.t. $\mathcal{T}$
- Use: debugging ontologies

Axiom Entailment

- Input: axiom α, TBox $\mathcal{T}$
- Task: determine whether α is entailed by $\mathcal{T}$
- Use: understanding content of the ontology

Classification

- Input: TBox $\mathcal{T}$
- Task: determine, for every pair of concept names A, B from $\mathcal{T}$, whether $\mathcal{T} \models A \sqsubseteq B$
- Use: visualizing / understanding ontology (also debugging)

MORE REASONING TASKS

Knowledge base satisfiability

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- Task: determine whether $\mathcal{K}$ is satisfiable
- Use: checking for contradictory information

MORE REASONING TASKS

Knowledge base satisfiability

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- Task: **determine whether $\mathcal{K}$ is satisfiable**
- Use: checking for contradictory information

Instance checking

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, concept C
- Task: **find all individuals a such that $\mathcal{K} \models C(a)$**
- Use: basic way to query the ABox

MORE REASONING TASKS

Knowledge base satisfiability

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- Task: **determine whether $\mathcal{K}$ is satisfiable**
- Use: checking for contradictory information

Instance checking

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, concept C
- Task: **find all individuals a such that $\mathcal{K} \models C(a)$**
- Use: basic way to query the ABox

Ontology-mediated query answering (OMQA)

- Input: KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, database query $q(\vec{x})$
  - typically, q is a conjunctive query (see later)
- Task: **determine the certain answer to $q(\vec{x})$ w.r.t. $\mathcal{K}$**, i.e. answers that hold in every model of $\mathcal{K}$
- Use: database-style querying of the data

RELATIONSHIP TO FIRST-ORDER LOGIC

Every DL KB corresponds to a set of first-order logic (FOL) sentences:

- each complex concept maps to a FOL formula with 1 free variable
  - $Person \sqcap \neg Student$ $\quad Person(x) \wedge \neg Student(x)$
  - $\geq 2\, parentOf.\top$ $\quad \exists y, z. parentOf(x, y) \wedge parentOf(x, z) \wedge y \neq z$

RELATIONSHIP TO FIRST-ORDER LOGIC

Every DL KB corresponds to a set of first-order logic (FOL) sentences:

- each complex concept maps to a FOL formula with 1 free variable
  - $Person \sqcap \neg Student$ $\qquad Person(x) \land \neg Student(x)$
  - $\geq 2\, parentOf.\top$ $\qquad \exists y, z. parentOf(x, y) \land parentOf(x, z) \land y \neq z$
- each TBox axiom maps to a FOL sentence
  - $Parent \sqsubseteq \exists parentOf.\top$ $\qquad \forall x.(Parent(x) \rightarrow \exists y. parentOf(x, y))$
  - $parentOf \equiv childOf^{-}$ $\qquad \forall x, y. parentOf(x, y) \leftrightarrow childOf(y, x)$
- each ABox statement maps to a FOL sentence
  - $\forall parentOf.Female(maria)$ $\qquad \forall y.(parentOf(maria, y) \rightarrow Female(y))$

RELATIONSHIP TO FIRST-ORDER LOGIC

Every DL KB corresponds to a set of first-order logic (FOL) sentences:

- each complex concept maps to a FOL formula with 1 free variable
  - $Person \sqcap \neg Student$ $\quad Person(x) \land \neg Student(x)$
  - $\geq 2\, parentOf.\top$ $\quad \exists y, z. parentOf(x, y) \land parentOf(x, z) \land y \neq z$
- each TBox axiom maps to a FOL sentence
  - $Parent \sqsubseteq \exists parentOf.\top$ $\quad \forall x.(Parent(x) \rightarrow \exists y.parentOf(x, y))$
  - $parentOf \equiv childOf^{-}$ $\quad \forall x, y.parentOf(x, y) \leftrightarrow childOf(y, x)$
- each ABox statement maps to a FOL sentence
  - $\forall parentOf.Female(maria)$ $\quad \forall y.(parentOf(maria, y) \rightarrow Female(y))$

DL semantics = standard FOL semantics applied to translated DL KBs

SHORT HISTORY OF DLS

1980-1990 **First implemented DL systems**

Negative theoretical results (undecidability, NP-hard)

Focus on tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$

Complexity: subsumption in PTIME (without TBox)

Algorithms: normalization + structural comparison

SHORT HISTORY OF DLS

1980-1990 **First implemented DL systems**

Negative theoretical results (undecidability, NP-hard)

Focus on tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$

Complexity: subsumption in PTIME (without TBox)

Algorithms: normalization + structural comparison

1990-2005 **Rise of (highly) expressive DLs** like $\mathcal{SROIQ} \approx$ OWL 2

Reasoning provably intractable: EXPTIME-hard or worse

Algorithms: tableaux method with optimizations

Despite high complexity, algorithms work in practice!

SHORT HISTORY OF DLS

1980-1990 **First implemented DL systems**

Negative theoretical results (undecidability, NP-hard)

Focus on tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$

Complexity: subsumption in PTIME (without TBox)

Algorithms: normalization + structural comparison

1990-2005 **Rise of (highly) expressive DLs** like $\mathcal{SROIQ} \approx$ OWL 2

Reasoning provably intractable: EXPTIME-hard or worse

Algorithms: tableaux method with optimizations

Despite high complexity, algorithms work in practice!

2005-now **Renewed interest in lightweight DLs**

$\mathcal{EL}$ family (OWL 2 EL) and DL-Lite family (OWL 2 QL)

Query answering becomes important task

Algorithms: materialization, query rewriting

NEXT SESSIONS

WHAT'S UP NEXT

Next three sessions (with Loïc Paulevé):

- TD on today's material
- TP: Protégé ontology editor

Closer look at DLs from mid-November (with me):

- Reasoning with expressive DLs (TD)
- Reasoning with lightweight DLs (TD, TP)
- Inconsistency handling, debugging, ...

REFERENCES AND EXTRA MATERIAL

TRANSLATION FROM ALC TO FOL

Translating complex concepts using π_x:

$$
\begin{aligned}
\pi_x(A) &= A(x) & \pi_y(A) &= A(y)\\
\pi_x(\neg C) &= \neg\pi_x(C) & \pi_y(\neg C) &= \neg\pi_y(C)\\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(D) & \pi_y(C \sqcap D) &= \pi_y(C) \wedge \pi_x(D)\\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(D) & \pi_y(C \sqcup D) &= \pi_y(C) \vee \pi_y(D)\\
\pi_x(\exists r.C) &= \exists y.r(x,y) \wedge \pi_y(C) & \pi_y(\exists r.C) &= \exists x.r(y,x) \wedge \pi_x(C)\\
\pi_x(\forall r.C) &= \forall y.r(x,y) \rightarrow \pi_y(C) & \pi_y(\forall r.C) &= \forall x.r(y,x) \rightarrow \pi_x(C)
\end{aligned}
$$

Translating knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ using π:

$$
\begin{aligned}
\pi(\mathcal{K}) &= \pi(\mathcal{T}) \cup \pi(\mathcal{A})\\
\pi(\mathcal{T}) &= \{\forall x.\pi_x(C) \rightarrow \pi_x(D) \mid C \sqsubseteq D \in \mathcal{T}\}\\
\pi(\mathcal{A}) &= \{\pi_x(C)[x/a] \mid C(a) \in \mathcal{A}\} \cup \{r(a,b) \mid r(a,b) \in \mathcal{A}\}
\end{aligned}
$$

where $\pi_x(C)[x/a]$ means replacing x by a in $\pi_x(C)$

RELATIONSHIP BETWEEN DLS AND FOL FRAGMENTS

The translation $\pi(\mathcal{K})$ of an $\mathcal{ALC}$ KB

- is expressible in the two-variable fragment
- is expressible in the guarded fragment

Same applies to many other DLs.

The satisfiability problem for formulas in the two-variable and guarded fragments is known to be decidable

- obtain decidability results for many DLs

Can also use translation to obtain complexity upper bounds

- exploit EXPTIME result for bounded-variable guarded fragment