

Knowledge Representation

Lecture 3: Introduction to Description Logics

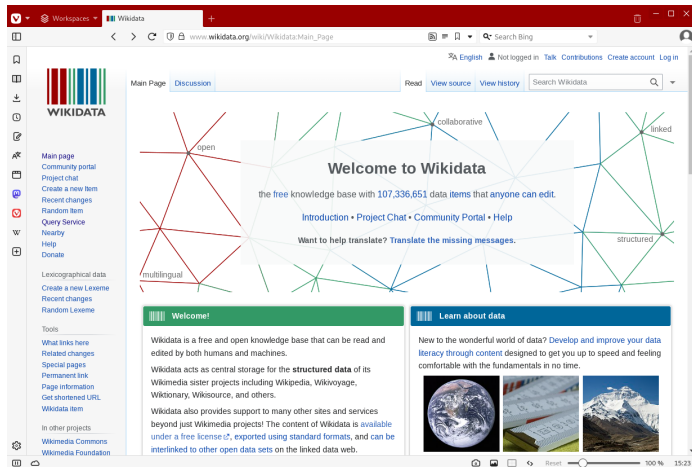
Patrick Koopmann (*using parts by Stefan Borgwardt*)

November 3, 2023

Knowledge Graphs and Ontologies

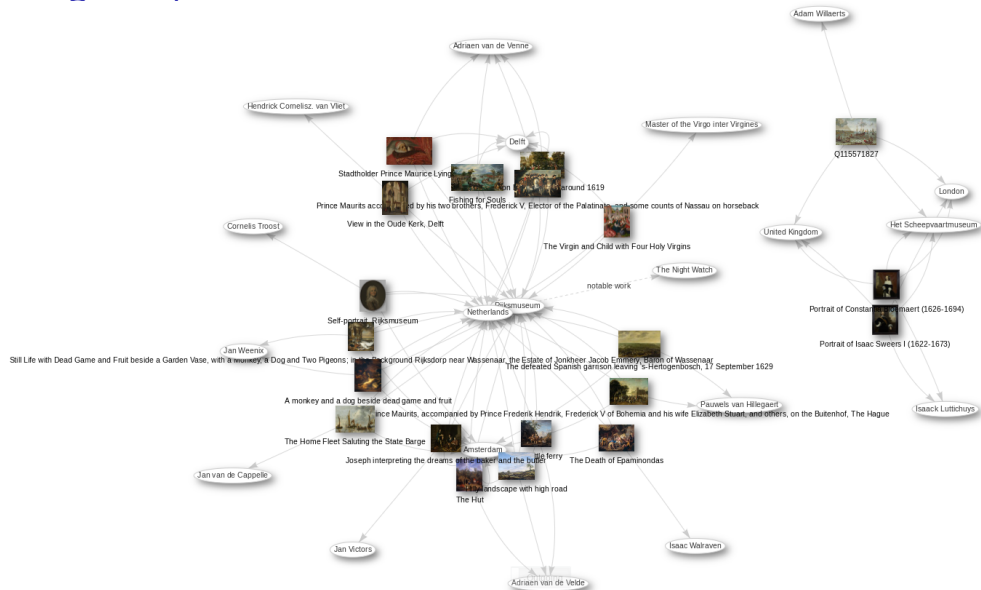
KR Formalisms You Might Know: Knowledge Graphs / Wikidata

Do you also know Wikidata?

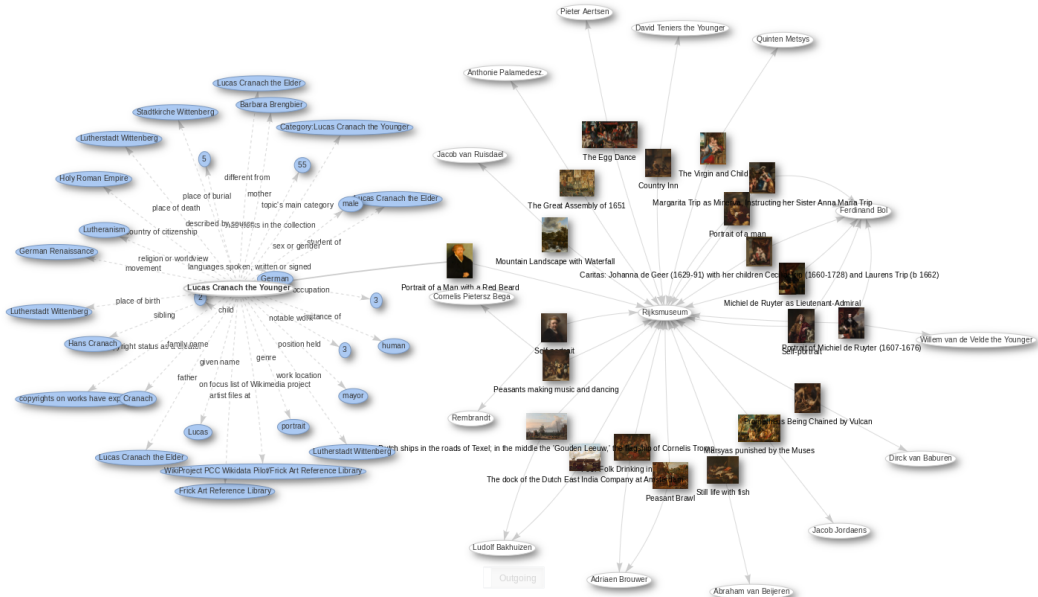


The screenshot shows the Wikidata Main Page in a web browser. The page features a central welcome message: "Welcome to Wikidata" followed by "the free knowledge base with 107,336,651 data items that anyone can edit." Below this, there are links for "Introduction", "Project Chat", "Community Portal", and "Help". A call to action says "Want to help translate? Translate the missing messages." The page is decorated with a network graph of colored lines (red, green, blue) connecting various concepts like "open", "collaborative", "linked", "structured", and "multilingual". On the left, there is a sidebar with navigation links such as "Main page", "Community portal", "Project chat", "Create a new item", "Recent changes", "Random item", "Query Service", "Nearby", "Help", "Donate", "Lexicographical data", "Create a new Lexeme", "Recent changes", "Random Lexeme", "Tools", "What links here", "Related changes", "Special pages", "Permanent link", "Page information", "Get shortened URL", "Wikidata item", "In other projects", "Wikimedia Commons", and "Wikimedia Foundation". At the bottom, there are two boxes: "Welcome!" and "Learn about data". The "Learn about data" box contains the text: "New to the wonderful world of data? Develop and improve your data literacy through content designed to get you up to speed and feeling comfortable with the fundamentals in no time." and three images: a globe, a stack of books, and a snow-capped mountain.

Knowledge Graph Wikidata



Knowledge Graph Wikidata



Knowledge Graphs in Google



ada lovelace



Images

Videos

News

Comparison sites

Quotes

Inventions

Education

Biography

Full name

All filters

Tools

About 9,150,000 results (0.38 seconds)

Ada Lovelace

Mathematician and writer



YouTube • SciShow

The Brilliant Mind of Ada Lovelace: Victorian Countess & ...

Introducing a brilliant woman of science from the 1800s: Ada Lovelace. She was...

5 Apr 2013

Born

December 10,
1815, London,
United
Kingdom

Died

November 27,
1852,
Marylebone,
London, Unite...

Britannica

Ada Lovelace |
Biography,
Computer, & Facts -
Britannica

3 weeks ago



Wikipedia

https://en.wikipedia.org/wiki/Ada_Lovelace

Ada Lovelace

Lovelace, identified as **Ada** Augusta Byron, is portrayed by Lily Lesser in the second season of The Frankenstein Chronicles. She is employed as an "analyst" to ...

[Ada Lovelace \(microarchitecture\)](#) · [Earl of Lovelace](#) · [Analytical engine](#) · [Lady Byron](#)




About



Augusta Ada King, Countess of Lovelace was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical general-purpose computer, the Analytical Engine. She was the first to recognise that the machine had applications beyond pure calculation. [Wikipedia](#)


Born: December 10, 1815, London, United Kingdom

People also ask :

Knowledge Graphs in Google








YouTube • SciShow

The Brilliant Mind of Ada Lovelace: Victorian Countess & ...

Introducing a brilliant woman of science from the 1800s: Ada Lovelace. She was ...


5 Apr 2013




Britannica

Ada Lovelace | Biography, Computer, & Facts - Britannica

3 weeks ago






Wikipedia

https://en.wikipedia.org/wiki/Ada_Lovelace

Ada Lovelace

Lovelace, identified as **Ada Augusta Byron**, is portrayed by Lily Lesser in the second season of *The Frankenstein Chronicles*. She is employed as an "analyst" to ...

[Ada Lovelace \(microarchitecture\)](#) · [Earl of Lovelace](#) · [Analytical engine](#) · [Lady Byron](#)



About

Augusta Ada King, Countess of Lovelace was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical general-purpose computer, the *Analytical Engine*. She was the first to recognise that the machine had applications beyond pure calculation. [Wikipedia](#)

Born: December 10, 1815, [London, United Kingdom](#)

Died: November 27, 1852, [Marylebone, London, United Kingdom](#)


Children: [Byron King-Noel, Viscount Ockham](#), [Anne Blunt, 15th Baroness Wentworth](#), [Ralph King-Milbanke, 2nd Earl of Lovelace](#)

Spouse: [William King-Noel, 1st Earl of Lovelace](#) (m. 1835–1852)

Parents: [Lord Byron](#), [Lady Byron](#)

Siblings: [Allegra Byron](#)

Grandchildren: [Ada Byron Milbanke, 14th Baroness Wentworth](#), [Judith Blunt-Lytton, 16th Baroness Wentworth](#)




Wikipedia





<https://nl.wikipedia.org/wiki/Ada...>

Ada Lovelace

Augusta **Ada Byron King**, **Lady Lovelace**, geboren Augusta **Ada Byron** (Londen, 10 december 1815 – aldaar (Marylebone), 27 november 1852) was een Britse wiskundine.



People also search for



Knowledge Graphs

Knowledge graphs are used in many places:

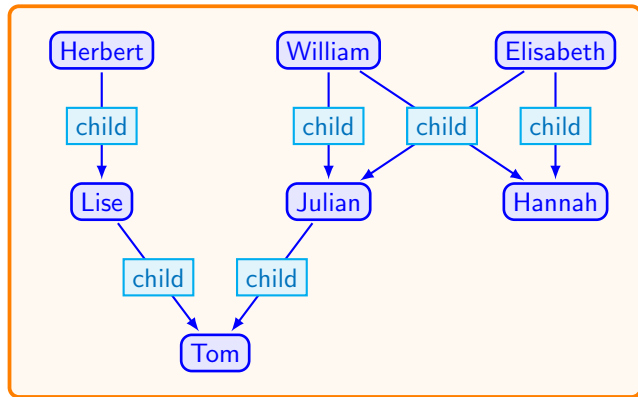
- ▶ Wikidata and DBpedia reflect knowledge of Wikipedia
- ▶ Search Engines: Google, Bing, Yahoo, etc.
- ▶ Question answering systems: WolframAlpha, Siri, Alexa
- ▶ Social Networks: Facebook, LinkedIn

Having **explicit representations** of knowledge has also advantages for AI systems

Knowledge graphs on their own are powerful, but they also have **limitations**:

- ▶ we **only** have the explicit knowledge
- ▶ graphs may be **incomplete**
- ▶ it is not straightforward to link **different knowledge sources**

Example: Family Tree



Consider the following queries:

- ▶ Who are the grandparents of Tom?
- ▶ Who is the aunt of Tom?
- ▶ Who is a parent?
- ▶ Who has a parent?

Other Sources of Knowledge

Knowledge may occur in many other different forms:

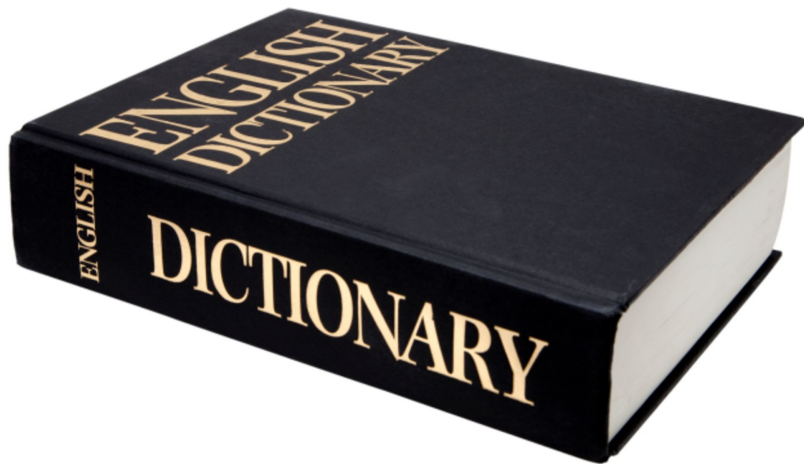
- ▶ web pages
- ▶ databases
- ▶ spreadsheets
- ▶ diagrams
- ▶ ...

It can be useful for an AI system to “understand” these data

- ▶ ... to infer implicit information
- ▶ ... to integrate data from different sources
- ▶ ... to intelligently answer queries

Motivation for Ontologies

Intuitively, the computer would need a dictionary...



Motivation for Ontologies

... where it can look up the meaning of the used terms.

Parent *n*

- 1 a father or a mother
- 2 an ancestor or precursor

Grandparent *n*

- 1 a parent of a parent

Aunt *n*

- 1 the sister of one's father or mother.
- 2 the wife of one's uncle.

- ▶ Such a dictionary has many advantages:
 - ▶ More intelligent querying of data
 - ▶ Integrating data from different sources
 - ▶ Extending incomplete data sets
- ▶ Ontologies are such “dictionaries for computer systems”

Ontologies

Ontology as a Discipline of Philosophy

Ontology *philosophy* the study of the nature of being [Greek *on* being + *logy*]

Branch of metaphysics

- ▶ What kinds of things/entities exist?
- ▶ What does it mean to exist?
- ▶ How are entities related?
- ▶ What are their basic categories? (eg. substances, property, relation, event,)
- ▶ Which entities are the most fundamental?
- ▶ “Ontology” is concerned with rather general and fundamental concepts.

Ontologies in Computer Science

Computer scientists are more pragmatic:

“For AI systems, what ‘exists’ is exactly that which can be represented.”
(Gruber, 1993)

- ▶ We are not concerned with *Ontology*, but with *ontologies*.

Definition of Ontologies

There are many definitions of ontologies in the literature. We use the following:

“An ontology is a **formal, explicit specification** of a **shared conceptualization**.” (Gruber 1994; Staab, Studer, 2009)

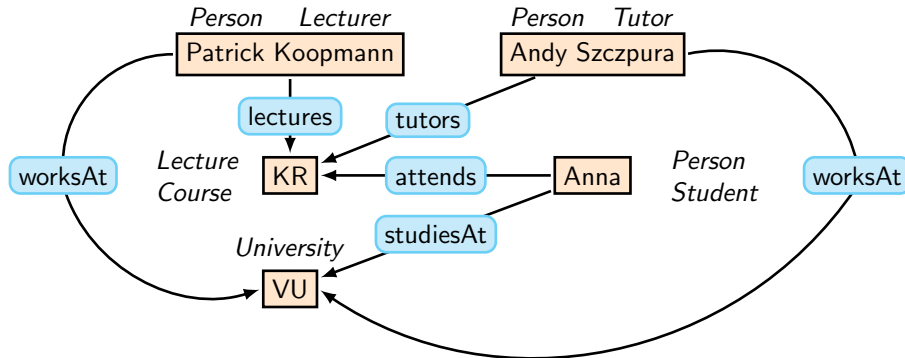
Conceptualizations

“An ontology is a formal, explicit specification of a shared **conceptualization**.” (Gruber 1994; Staab, Studer, 2009)

Conceptualizations (after Gruber 1994)

A **conceptualization** is given by:

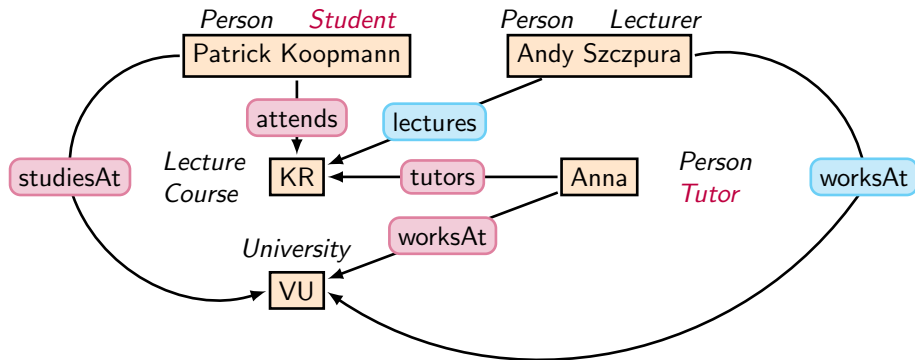
- ▶ a **domain of discourse**
- ▶ a set of **conceptual relations**, for example:
 - ▶ **concepts** (unary predicates),
 - ▶ **roles** or **relations** (binary predicates)



Conceptualizations (after Gruber 1994)

A **conceptualization** is given by:

- ▶ a **domain of discourse**
- ▶ a set of **conceptual relations**
- ▶ for different **states** of the world, and different **possible worlds**



Conceptualizations

We usually consider a simplified view based on *first-order structures*:

- ▶ possible worlds correspond to **interpretations**
- ▶ the domain of discourse is not fixed
- ▶ as **conceptual relations** we consider **concepts** (unary) and **roles** (binary)

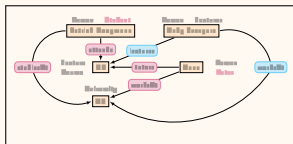
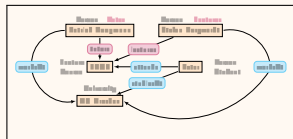
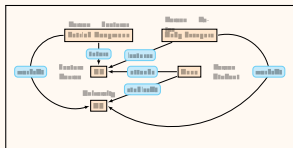
Formal, Explicit Specifications

“An ontology is a **formal, explicit specification** of a shared conceptualization.” (Gruber 1994; Staab, Studer, 2009)

Formal, Explicit Specifications

Extensional specification:

- ▶ explicitly state the elements of the conceptual relations



• • •

Formal, Explicit Specifications

Extensional specification:

- ▶ explicitly state the elements of the conceptual relations

Intensional specification:

- ▶ constrain concepts and roles in the different possible worlds
- ▶ *axiomatize*

Every lecture is a course.

A TA is someone who teaches a course.

Every TA works at a university.

⋮

Formal, Explicit Specifications

Extensional specification:

- ▶ explicitly state the elements of the conceptual relations

Intensional specification:

- ▶ constrain concepts and roles in the different possible worlds
- ▶ *axiomatize*
- ▶ **formal** languages make these specifications *precise*

$$\forall x. (\text{Lecture}(x) \rightarrow \text{Course}(x))$$
$$\forall x. (\text{TA}(x) \leftrightarrow \exists y. (\text{teaches}(x, y) \wedge \text{Course}(y)))$$
$$\forall x. (\text{TA}(x) \rightarrow \exists y. (\text{worksAt}(x, y) \wedge \text{University}(y)))$$
$$\vdots$$

Formal, Explicit Specifications

Extensional specification:

- ▶ explicitly state the elements of the conceptual relations

Intensional specification:

- ▶ constrain concepts and roles in the different possible worlds
- ▶ *axiomatize*
- ▶ **formal** languages make these specifications *precise*
- ▶ dedicated languages make them **human readable**

Lecture **SubClassOf** Course

TA **EquivalentTo** teaches **some** Course

TA **SubClassOf** worksAt **some** University

...

Formal, Explicit Specifications

Extensional specification:

- ▶ explicitly state the elements of the conceptual relations

Intensional specification:

- ▶ constrain concepts and roles in the different possible worlds
- ▶ *axiomatize*
- ▶ *formal* languages make these specifications *precise*
- ▶ dedicated languages make them *human readable*

Lecture \sqsubseteq *Course*

TA $\equiv \exists \text{teaches.} \textit{Course}$

TA $\sqsubseteq \exists \text{worksAt.} \textit{University}$

\vdots

Formal, Explicit Specifications

“An ontology is a formal, explicit specification of a **shared** conceptualization.” (Gruber 1994; Staab, Studer, 2009)

Ontologies allow to share a fixed conceptualization with different parties

- ▶ AI system and user
- ▶ different user groups/companies
- ▶ different systems (communication via the web)

Ontologies in Practice

Search and Semantic Web

Schema.org

- ▶ Launched in 2011 by Bing, Google and Yahoo!
- ▶ Selection of schemas for metadata of web content
- ▶ Influences for example search results by Google's Knowledge Graph
- ▶ Relatively small:
 - ▶ 797 Concepts, 1,453 Roles

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Inglourious Basterds</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
    Director: <span itemprop="name">Quentin Tarantino</span>
    (born <time itemprop="birthDate" datetime="1963-03-27">March 27, 1963</time>)
  </div>
  <span itemprop="genre">War Film</span>
  <a href="../movies/ingbast-theatrical-trailer.html" itemprop="trailer">Trailer</a>
</div>
```

SNOMED CT

- ▶ Clinical healthcare terminology
- ▶ 361,042 concepts and 242 roles
- ▶ Concepts for:
 - ▶ clinical findings,
 - ▶ symptoms
 - ▶ diagnoses
 - ▶ procedures
 - ▶ body structures
 - ▶ organisms and other etiologies,
 - ▶ substances,
 - ▶ pharmaceuticals,
 - ▶ devices,
 - ▶ specimens.
- ▶ Applications:
 - ▶ electronic health records
 - ▶ catalogues of clinical services
 - ▶ clinical decision support systems
 - ▶ ...

SNOMED CT

7 new concepts were added to SNOMED CT on March 9, 2020.

ConceptID	Description
840534001	COVID-19 vaccination
840544004	Suspected COVID-19
840535000	Antibody to SARS-CoV-2
840536004	Antigen of SARS-CoV-2
840539006	COVID-19
840546002	Exposure to SARS-CoV-2
840533007	SARS-CoV-2

These were used by doctors around the world (US, UK, Germany, Argentina, India, Israel, ...) to record their diagnoses.

Gene Ontology (GO)

- ▶ Developed by the Gene Ontology consortium since 1998
- ▶ Biological processes and their interactions
- ▶ 63.000 concepts and 300 roles
- ▶ Main application: biological research

$DNAMetabolicProcess \equiv MetabolicProcess \sqcap \exists hasParticipant.DNA$
 $MAPKCascade \sqsubseteq MetabolicProcess \sqcap \exists partOf.CellCommunication$

Biology and Medicine

NCBO BioPortal

- ▶ Source of ontologies on biology and medicine
- ▶ <https://bioportal.bioontology.org/>
- ▶ 975 ontologies (over 100 more than last year)
- ▶ 13,794,030 concepts
- ▶ 36,286 Roles
- ▶ Some examples:
 - ▶ National Cancer Institute Thesaurus (NCIT)
 - ▶ SNOMED Clinical Terms (SNOMED CT)
 - ▶ Gene Ontology (GO)
 - ▶ Semantic Web for Earth and Environment Technology Ontology (SWEET)
 - ▶ COVID-19 Ontology (COVID-19)

Common Core Ontologies

- ▶ Developed by non-profit R&D company CUBRC, since 2010
- ▶ Formalize generic notions found in many applications

Time Ontology:

“A day is a temporal interval. An hour occurs during a day. The relation ‘during’ is transitive.”

Day \sqsubseteq *OneDimensionalTemporalRegion*

Hour \sqsubseteq \exists *intervalDuring*.*Day*

intervalDuring \circ *intervalDuring* \sqsubseteq *intervalDuring*

Common Core Ontologies

- ▶ Developed by non-profit R&D company CUBRC, since 2010
- ▶ Formalize generic notions found in many applications

Agent Ontology:

“An agent is an organization or person that acts in some process”.

“A group of agents contains at least one agent and consists only of agents.”

$Agent \equiv (Organization \sqcup Person) \sqcap \exists agentIn.Process$

$GroupOfAgents \sqsubseteq \exists hasPart.Agent \sqcap \forall hasPart.Agent$

More Examples

- ▶ **GeoNames**
 - ▶ ontology to specify geographical information (countries, cities, rivers, borders, etc)
 - ▶ used for GIS (Geo Information Systems)
- ▶ **LKIF Legal Core Ontology**
 - ▶ collection of ontologies for the legal domain
- ▶ **General Ontology for Linguistic Description (GOLD)**
 - ▶ ontology about human language
- ▶ **Process Specification Language (PSL)**
 - ▶ used to model manufacturing processes
- ▶ **FoodOn**
 - ▶ Ontology about food
- ▶ **SWARMS**
 - ▶ Ontology about autonomous underwater vehicles (submarine robots)

Ontology Languages

There are very different ontology languages used in practice:

- ▶ **Higher order logic** (SUMO)
 - ▶ relatively rare
 - ▶ limited practicality
- ▶ **RDFS**
 - ▶ “**R**ich **D**ata **F**ormat **S**chema”
 - ▶ very common for knowledge graphs in **RDF**
 - ▶ limited expressivity
 - ▶ complicated semantics
 - ▶ covered in Masters course “Knowledge Representation in the Web”
- ▶ **Datalog**
 - ▶ Rule-based language
 - ▶ Common systems are RDFS, VLog and Nemo
- ▶ **OWL**
 - ▶ “**W**eb **O**ntology **L**anguage”
 - ▶ Most expressive decidable formalism in this list
 - ▶ Based on **description logics**

Ontologies in Practice

- ▶ Existing ontologies vary a lot in size and expressivity
 - ▶ some ontologies are just taxonomies
 - ▶ others allow for complex logical inferencing
- ▶ Many knowledge graphs are used without or with very simple ontologies
 - ▶ in Wikidata, a lot of relevant knowledge is already there
 - ▶ terms are often not used coherently enough to allow for logical reasoning
- ▶ In the context of this course, we are interested in the more expressive ontology formalisms

The Description Logic \mathcal{ALC}

Description Logics

Description logics (DLs)

- ▶ are **decidable** fragments of first-order logic
- ▶ are restricted to **unary** and **binary** predicates
- ▶ use a **special syntax** for formulas
- ▶ have the specification of ontologies as main use case

Vocabulary

Concepts / Classes / Categories:

Person, Student, Teacher, Room, Building, University, ...

Roles / Relations / Properties / Attributes:

attends, teaches, is part of, is a, belongs to, is employed by, ...

Objects / Individuals:

Patrick Koopmann, KR Lecture, Vrije Universiteit Amsterdam, Netherlands, ...

Vocabulary

Formally the syntax of DLs is based on the following infinite, disjoint sets:

concept names

$\mathbf{C} = \{A, B, \dots\}$

role names

$\mathbf{R} = \{r, s, \dots\}$

individual names

$\mathbf{I} = \{a, b, \dots\}$

- ▶ Together, they are called the **vocabulary**.
- ▶ Concept names A describe **sets**.

The concept name **Person** denotes the set of all persons.

- ▶ Role names describe binary relations.

belongsTo denotes the set of all pairs (x, y) where x **belongs to** y .

- ▶ Individual names describe individual objects.

vrijeUniversiteit denotes this university.

Semantics

- ▶ **Semantics** is used to specify the meaning of DL expressions
- ▶ As for classical logics, we use **interpretations** for this
- ▶ Recall propositional logic:
 - ▶ Vocabulary consists of propositional variables
 - ▶ Interpretations assign **true** or **false** to those
- ▶ For the DL vocabulary, we need to talk about individual objects and their relations

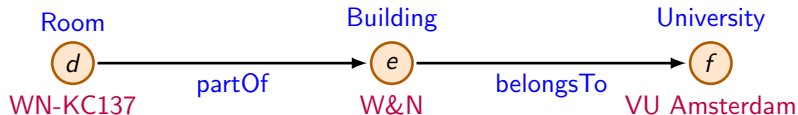
Interpretations

The semantics of DLs is based on (first-order) interpretations.

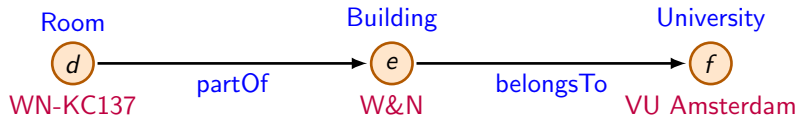
A DL **interpretation** is a tuple $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where

- ▶ $\Delta^{\mathcal{I}}$ is a **non-empty** set, called the **domain** of \mathcal{I} ,
- ▶ $\cdot^{\mathcal{I}}$ is the **interpretation function** that assigns meanings to names:
 - ▶ each $A \in \mathbf{C}$ is interpreted as a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$,
 - ▶ each $r \in \mathbf{R}$ is interpreted as a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$,
 - ▶ each $a \in \mathbf{I}$ is interpreted as an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

Interpretations can be represented as labeled graphs:



Example: An Interpretation



$\mathbf{C} = \{\text{Room}, \text{Building}, \text{University}, \text{Lecture}, \dots\}$

$\mathbf{R} = \{\text{partOf}, \text{belongsTo}, \dots\}$

$\mathbf{I} = \{\text{WN-KC137}, \text{W\&N}, \text{VUAmsterdam}, \dots\}$

$\Delta^{\mathcal{I}} = \{d, e, f\}$

$\text{Room}^{\mathcal{I}} = \{d\}$

$\text{partOf}^{\mathcal{I}} = \{(d, e)\}$

$\text{WN-KC137}^{\mathcal{I}} = d$

$\text{Building}^{\mathcal{I}} = \{e\}$

$\text{belongsTo}^{\mathcal{I}} = \{(e, f)\}$

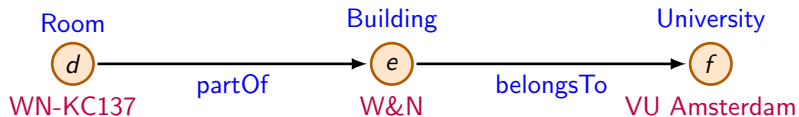
$\text{W\&N}^{\mathcal{I}} = e$

$\text{University}^{\mathcal{I}} = \{f\}$

$\text{VUAmsterdam}^{\mathcal{I}} = f$

$\text{Lecture}^{\mathcal{I}} = \emptyset$

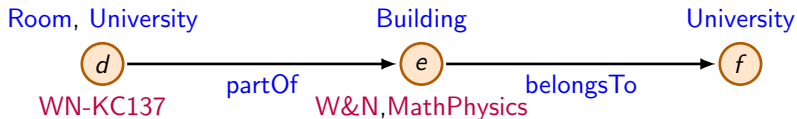
Important Facts about Interpretations



- ▶ The domain $\Delta^{\mathcal{I}}$ can be infinite.
- ▶ Elements of $\Delta^{\mathcal{I}}$ are called (domain) elements, individuals, or objects.
- ▶ $Room^{\mathcal{I}} = \{d\}$ is called the extension of *Room*
- ▶ The individual e is different from the individual name *W&N*.
- ▶ f is called belongsTo-successor of e .
- ▶ e is called belongsTo-predecessor of f .

Example: Another Interpretation

Interpretations can be completely arbitrary:



- ▶ Different individual names can be interpreted as the same individual, e.g. *W&N* and *Math&Physics* are interpreted as *e*.
- ▶ There can be *unnamed* individuals, e.g. *f*, which do not have a corresponding individual name.
- ▶ Rooms can be universities, which intuitively does not make sense.
- ▶ The *axioms* in an ontology restrict the set of interpretations, e.g. by stating that *rooms* cannot be lectures.

Concepts

Before we introduce **axioms**, we introduce complex **concepts**

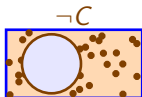
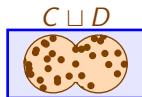
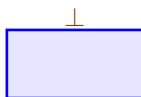
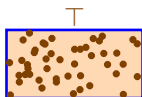
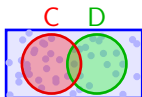
- ▶ also: *concept descriptions*, *compound concepts*, or just *concepts*
- ▶ describe **sets of objects** in an interpretation
- ▶ central elements in ontologies and ontology axioms
- ▶ the **semantics** of concepts is captured by the **interpretation function**:
 - ▶ $\cdot^{\mathcal{I}}$ is extended to map concepts C to subsets $C^{\mathcal{I}}$ of the domain $\Delta^{\mathcal{I}}$

Concepts

We define \mathcal{ALC} concepts C and their semantics $C^{\mathcal{I}}$ **inductively**.

- ▶ every concept name A is a concept with semantics $A^{\mathcal{I}}$
- ▶ if C, D are concepts, then the following are also concepts:

Name:	top	bottom	conjunction	disjunction	complement
Syntax:	\top	\perp	$C \sqcap D$	$C \sqcup D$	$\neg C$
Semantics:	$\Delta^{\mathcal{I}}$	\emptyset	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$



Concepts

We define \mathcal{ALC} concepts C and their semantics $C^{\mathcal{I}}$ **inductively**.

- ▶ every concept name A is a concept with semantics $A^{\mathcal{I}}$
- ▶ if C, D are concepts, then the following are also concepts:

Name:	top	bottom	conjunction	disjunction	complement
Syntax:	\top	\perp	$C \sqcap D$	$C \sqcup D$	$\neg C$
Semantics:	$\Delta^{\mathcal{I}}$	\emptyset	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$

- ▶ These correspond to the operators in propositional logic.

Concepts

We define \mathcal{ALC} concepts C and their semantics $C^{\mathcal{I}}$ **inductively**.

- ▶ every concept name A is a concept with semantics $A^{\mathcal{I}}$
- ▶ if C, D are concepts, then the following are also concepts:

Name:	top	bottom	conjunction	disjunction	complement
Syntax:	\top	\perp	$C \sqcap D$	$C \sqcup D$	$\neg C$
Semantics:	$\Delta^{\mathcal{I}}$	\emptyset	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$

- ▶ These correspond to the operators in propositional logic.

Examples for complex concepts:

$\text{Room} \sqcap \text{Lecture} \quad \neg \text{Building}$

Concepts using Roles

Additionally, we can describe **outgoing role connections** by specifying the concepts of the role successors.

- If C is a concept and r is a role, then the following are also concepts:

Name: existential restriction

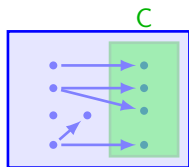
Syntax: $\exists r.C$

Semantics: $\{d \mid \exists e.(d, e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$

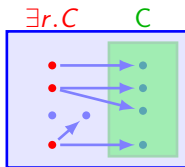
Name: value restriction

Syntax: $\forall r.C$

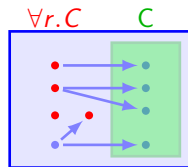
Semantics: $\{d \mid \forall e.(d, e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$



Some
 r -successor
in C : $\exists r.C$



All
 r -successors
in C : $\forall r.C$



Concepts using Roles

Additionally, we can describe **outgoing role connections** by specifying the concepts of the role successors.

- If C is a concept and r is a role, then the following are also concepts:

Name:	existential restriction	value restriction
Syntax:	$\exists r.C$	$\forall r.C$
Semantics:	$\{d \mid \exists e.(d, e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$	$\{d \mid \forall e.(d, e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$

Examples:

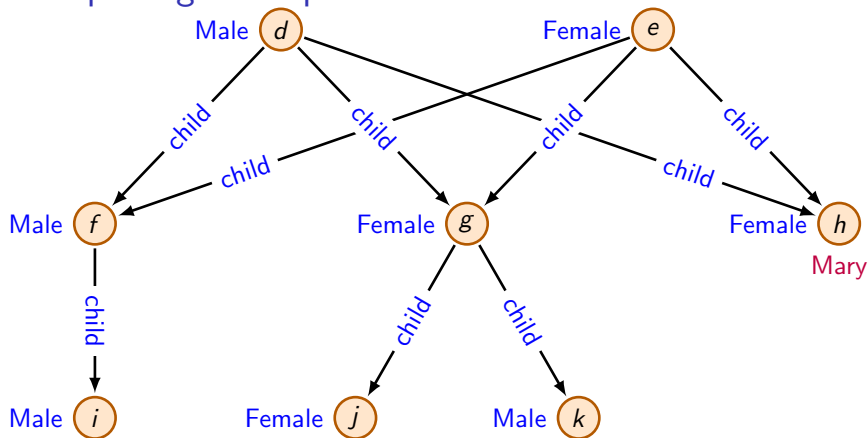
$\exists \textit{hasChild}.\textit{Girl}$ $\exists \textit{hasChild}.\exists \textit{hasChild}.\top$ $\forall \textit{eats}.\textit{PlantBased}$ $\exists \textit{belongsTo}.\top$

Syntax of \mathcal{ALC} Concepts

\mathcal{ALC} is the description logic that allows the concept constructors \top , \perp , \sqcap , \sqcup , \neg , \exists , and \forall to build concepts.

Other DLs may use different constructors.

Example: Interpreting Concepts



$$\begin{aligned}
 (\exists \textit{child}.\top)^{\mathcal{I}} &= \\
 (\textit{Female} \sqcap \exists \textit{child}.\top)^{\mathcal{I}} &= \\
 (\neg \exists \textit{child}.\top)^{\mathcal{I}} &=
 \end{aligned}$$

$$\begin{aligned}
 (\exists \textit{child.Female})^{\mathcal{I}} &= \\
 (\forall \textit{child.Female})^{\mathcal{I}} &= \\
 (\exists \textit{child}.\perp)^{\mathcal{I}} &=
 \end{aligned}$$

From Concepts to Ontologies

- ▶ Interpretations are arbitrary
 - ▶ We usually do not know which interpretation is the **correct one**
- ▶ Only some will be models of our conceptualization
- ▶ To specify what is a model, we use **axioms**
- ▶ Axioms
 - ▶ express **constraints** on interpretations
 - ▶ relate concepts to other concepts
 - ▶ relate concepts to individuals
- ▶ An **ontology** is then a collection of axioms.

From Concepts to Ontologies

- ▶ DL ontologies are **sets of axioms**
- ▶ Axioms put **constraints** on interpretations
- ▶ We distinguish two types:
 - ▶ **terminological** axioms put **concepts** in relation
 - ▶ **assertions** relate **individual names** with concepts and roles
- ▶ They respectively form the **TBox** and the **ABox** of an ontology
- ▶ An interpretation satisfying such an axiom/ontology is a **model**
 - ▶ \mathcal{I} satisfies α : $\mathcal{I} \models \alpha$

Terminological Axioms

If C and D are concepts, then the following is a (terminological) axiom:

Name: *general concept inclusion (GCI)* *equivalence axiom*

Syntax: $C \sqsubseteq D$

$C \equiv D$

Semantics: $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

$C^{\mathcal{I}} \equiv D^{\mathcal{I}}$

$Lecture \sqsubseteq Course$ $Room \sqsubseteq \neg Lecture$

$Room \sqsubseteq Structure \sqcap \exists partOf. Building$

$TA \equiv Person \sqcap \exists teaches. Course$

$Lecturer \equiv Person \sqcap \exists teaches. Lecture$

Assertions

Assertions (also called **facts**) are axioms about named individuals.

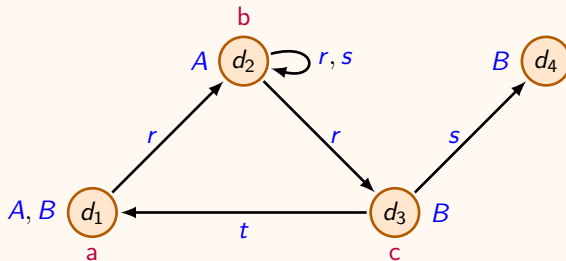
Given $a, b \in \mathbf{I}$, a concept C , and $r \in \mathbf{R}$, the following are assertions:

Name:	concept assertion	role assertion
Syntax:	$a : C$	$(a, b) : r$
Semantics:	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

WN-KC137 : *Room* (*W&N*, *VUAmsterdam*) : *belongsTo*

Exercise: Axioms

Let's look at the following interpretation \mathcal{I} :



Which of the following axioms does it satisfy:

- | | | |
|----------------------|---|--|
| 1. $(a, b) : r$ | 4. $A \sqsubseteq \exists r.A$ | 7. $\exists r.\top \sqsubseteq A$ |
| 2. $c : A$ | 5. $A \sqsubseteq \forall r.(A \sqcup B)$ | 8. $\exists r.\perp \sqsubseteq B$ |
| 3. $c : \exists s.B$ | 6. $A \equiv \forall r.(A \sqcup B)$ | 9. $\exists r.A \sqsubseteq \forall s.A$ |

Ontologies

An **ontology** is a set $\mathcal{O} = \mathcal{A} \cup \mathcal{T}$, where

- ▶ \mathcal{A} is an **ABox**, a finite set of assertions,
- ▶ \mathcal{T} is a **TBox**, a finite set of GCIs,

An interpretation is a **model** of \mathcal{O} (written $\mathcal{I} \models \mathcal{O}$) if it is a model of all axioms in \mathcal{O} .

- ▶ The ABox contains facts about named individuals (**data**), the TBox contains (**terminological**) **knowledge** that applies to all individuals.

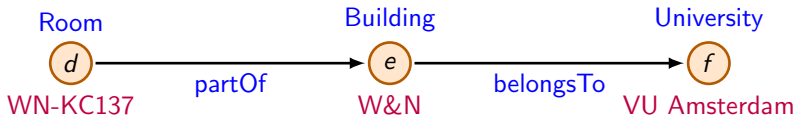
Example: An Ontology

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{WN-KC137 : Room, (WN-KC137, W\&N) : partOf\}$

$\mathcal{T} = \{Room \sqsubseteq \neg University, Room \sqsubseteq \exists partOf. Building, \\ Building \sqsubseteq \neg University, Building \sqsubseteq \neg Room\}$

This ontology has many models, for example the following:



Example: An Ontology

$\mathcal{O} = \mathcal{A} \cup \mathcal{T}$ with

$\mathcal{A} = \{WN-KC137 : Room, (WN-KC137, W\&N) : partOf\}$

$\mathcal{T} = \{Room \sqsubseteq \neg University, Room \sqsubseteq \exists partOf. Building, \\ Building \sqsubseteq \neg University, Building \sqsubseteq \neg Room\}$

The following interpretation is not a model of the ontology:

