LoD
The logic of Descriptions
(HP2T)

LoD – The Logic of descriptions

- **Intuition**
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

LoD – Why a logic of Descriptions?

- The **Logic of Descriptions** (LoD) allows us to represent etypes and dtypes and the properties which correlate them. It allows to reason about the etypes and dtypes which are populated in a LoE EG.
- It allows to construct a new **complex etype** based on the **properties** of the elements of an existing etype. For instance a foreigner may be described as a person who does not speak the local language, or a pet as an animal which lives in the house
- It allows to construct a new **complex composite etype** from simpler etypes, starting from the basic ones, as defined in LoE. For instance, it allows to define a parent as the union of father and mother.
- It allows to **constraint** the extension of etypes (via a **description**). For instance, it allows to say that a woman and a man are disjoint etypes, and that they are more specific than the etype person.
- It allows to **define** a new type by given a name (via a **definition**) to a previously constructed complex etype
- It allows to **reason** about how the meaning of complex as a function of the meaning of basic etypes.
- It allows to **extend** LoE graphs with the information about their etypes and to reason about these extended LoE Graphs (as part of the **LoDE** logic)

LoD – Highlights

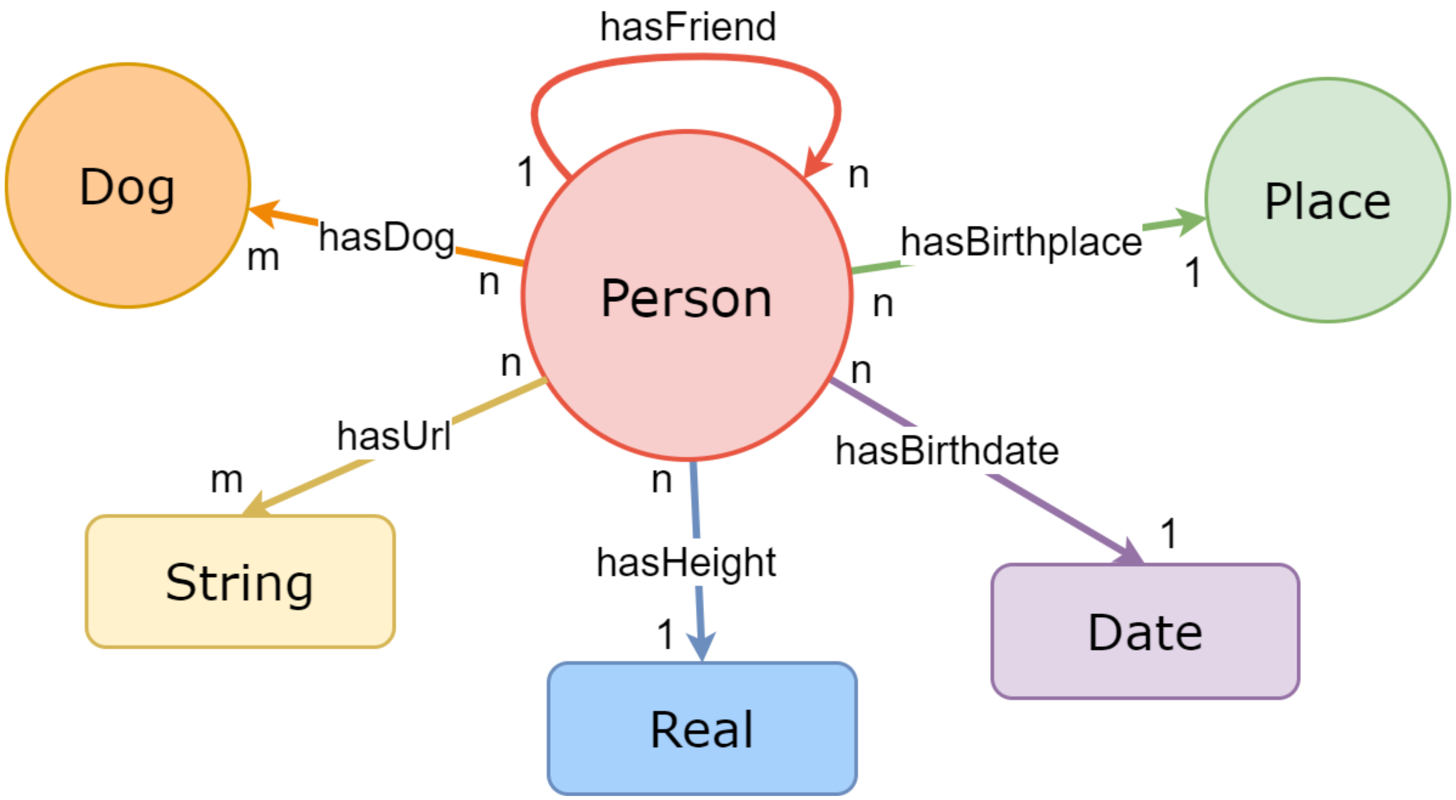
- LoD is the logic encoding the general knowledge about entities (how they are defined and how they can be described).
- LoD represents etypes and the properties and relations of etypes. It does not represent specific entities. Knowledge is at the type level, not at the entity level
- LoD is a world logic with a graph linguistic/ analogic representation
- Any LoE EG is built with reference to a LoD ETG. ETGs can be thought of specifiying the schema of Egs.
- LoD is conceptually similar to the Terminology Box (Tbox) of Description Logics (DL). The moves is from DBs to KGs.

LoD – which percepts

In LoD we have the following ETG elements:

- An **entity type** (etype) is a class of entities (corresponding to the etype to which an entity belongs in a LoE EG).
- A **datatype** (dtype) is a class of (data) values (corresponding to the dtype to which a value belongs in a LoE EG).
- An **Object Property** describes a relation between two etypes (not beween two entities, as in LoE).
- A **Data Property**, also called **Attribute**, describes a characteristic of an etype (not of an entity as in LoE).

An example of LoD ETG



Which percepts?

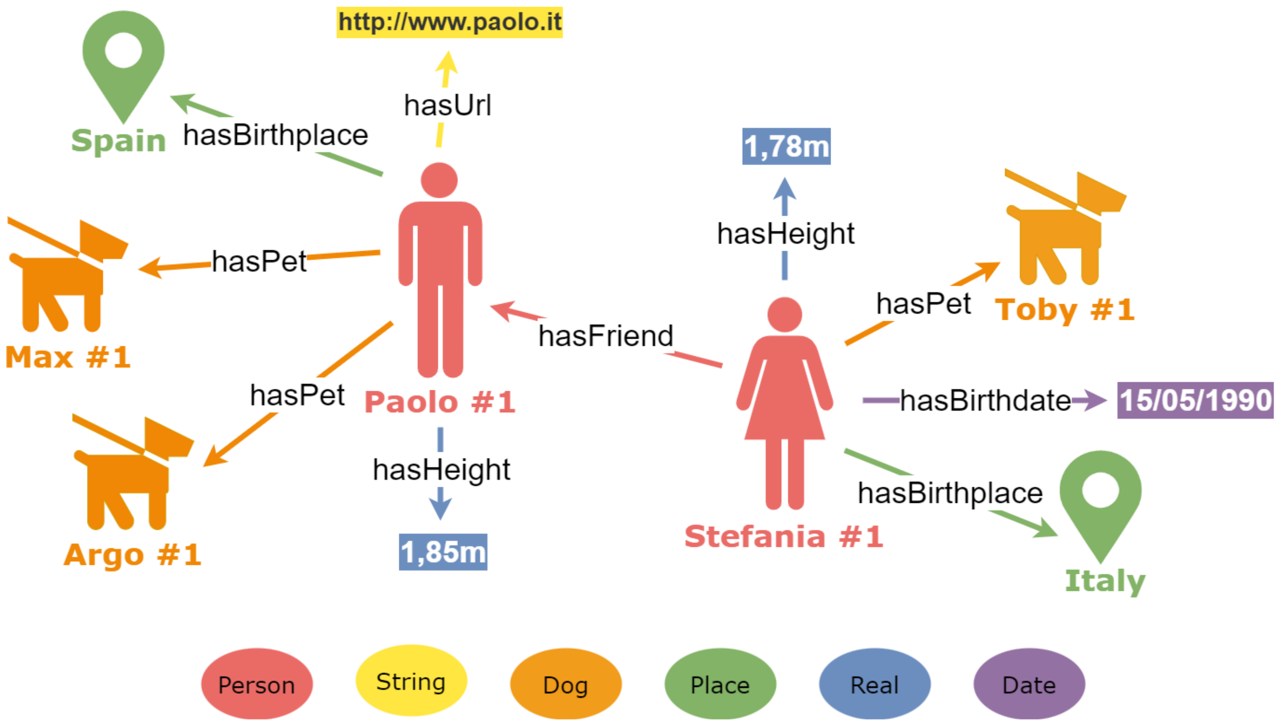
Which facts?

Well-formedness conditions

An ETG, to be well-formed must satisfy the following conditions:

- Each node is associated one and ONLY one etype/ dtype.
- Each link is associated with one and only one data or object property.
- Data and object properties must have the correct etypes or datatypes (strong typing).
- No links are allowed starting from dtype nodes.

An example of EG for the previous ETG



Observation (ETG, EG). An ETG defines all the etypes, dtypes, object properties and attributes used in an EG. An EG is an expansion (notion formally defined later on) where each not and link is expanded into all its elements. Compare with the previous ETG.

LoD – which alphabet elements?

The **same** alphabet elements as percepts, that is:

- **Entity**, **etype**, **value, dtype, Attribute**, **Object Property** (as from ETG above).

plus language elements needed to build knowledge statements

- Etype specializations, using object and data properties
- Etype constructors, that is: intersection, union, complement.
- Defined etypes
- Equivalence/ subsumption /djsjointness assertions of etypes (Semantically: set equality, subset relation, disjointness)

LoD – The Logic of descriptions

- Intuition
- **Definition**
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

LoD – The Logic of Descriptions - definition

We formally define LoD as follows

$$\mathbf{LoD} = \langle \mathbf{ETG}, \models_{\mathbf{LoD}} \rangle$$

with

$$\mathbf{ETG} = \langle \mathbf{L_{LoD}}, \mathbf{D}, \mathbf{I_{LoD}} \rangle$$

When no confusion arises, we drop the subscripts.

LoD – The Logic of descriptions

- Intuition
- Definition
- **Domain**
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

Domain

Definition (Domain)

$$D = < \{C\}, \{R\} >$$

where:

- {u} is a set of **units** $u_1, \ldots, u_n$, where $u_1 \in U$, with U= {u} is the **universe** of D
- {C} is a set of **classes** $C_1, \ldots, C_m$ of units, for some m, with $C_i \subseteq U$
- {R} is a set of **binary relations** $R_1, \ldots, R_p$ between units, for some p, with $R_i \subseteq U \times U$

Observation (Domain of interpretation). The units in U are not part of the domain of interpretation of LoD. They are left implicit. The alphabet does not mention them.

Classes

Definition (Class)

$$\{C\} = E_T \cup D_T$$

where:

- $U = \{u\}$ is the **universe of interpretation**;
- $E = \{e\} \subseteq \{u\} = U$ is the **entity universe**;
- $V = \{v\} \subseteq \{u\} = U$ is the **value universe**;
- $\{e\}$ and $\{v\}$ are disjoint.
- $E_T = \{E_i\}$ is a **set of etypes** E_i, with $E_i = \{e\} \subseteq E$
- $D_T = \{D_i\}$ is a **set of dtypes** D_i, with $D_i = \{v\} \subseteq V$

Observation (LoD classes). The same classes as LoE, but with no reference to the entities e and values v in U.

Binary relations

Definition (Binary relation)

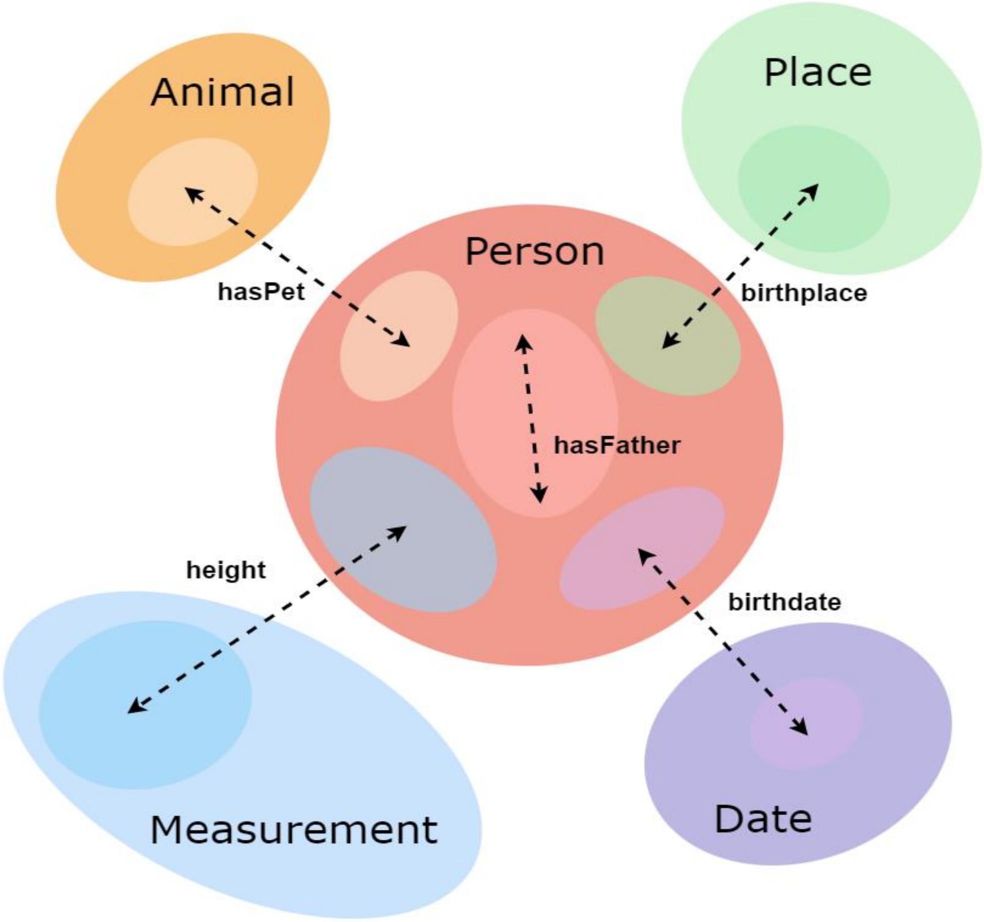
$$\{R\} = O_R \cup A_R$$

where:

- $O_R = \{O_i\}$ is a **set of object properties** O_i, with $O_i \subseteq E_k \times E_j$
- $A_R = \{A_i\}$ is a **set of attributes** A_i, with $A_i \subseteq E_k \times D_j$

Observation (Object and data relations). The same relations as LoE.

An example of ETG – Venn diagram



An example of domain of ETG (continued)

E_T = {Entity, P, D, L, …}
D_T = {Data, Real, String, …}
{R} = {hF, hD, hH, hB, hL, hU, …}

from which we construct the following facts in the domain:

D = {P $\subseteq$ entity, Real $\subseteq$ data, hF(P, P), hD(P, D), hH(P, Real), …}

with, e.g., hF(P, P) standing for hF $\subseteq$ P x P

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- **Language – intuition**
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

Language

Intuition (Language). The LoD language is progressively and compositionally extended in three steps:

- **Step 1 – The language of etype percepts.** It is an etype construction language. Composes etypes and properties to generate new etypes (e.g., a "mother" as a "parent" who is a "female");
- **Step 2 – The language of composite etype percepts.** It is a composite etype construction language. Composes pairs of etypes to generate complex new etypes (e.g., the union of "mother" and "father");
- **Step 3 – The language of Descriptions (LoD language).** It extends the alphabet by introducing names for defined etypes, as constructed in steps 1 and 2 (e.g., the new etype "parent" defined as the union of the etypes "mother" and "father"). Circular definitions are allowed.

Observation (Language). The LoD language is constructed in step 3 by exploiting the language extensions of the first two steps.

Language - Observations

Observation (Percepts and facts). Step 1 and step 2 only allow for the extension of the types of percepts. i.e., etypes, which can be talked about. Step 3 allows for assertions about what is the case in the domain.

Observation (LoD facts). The LoD language (step 3) assertions do NOT necessarily describe single facts but complex composition of facts (e.g., the fact that cars are vehicles and that they have four vehicles).

Observation (LoD percepts). The LoD language (step 3) allows to **define new etypes**, that is, to introduce new terms which extend the language and which, therefore, generate in the domain of interpretation new simple percepts, i.e., etypes. This is the basic mechanism by which natural language and knowledge works. Thus, for instance:

- I can **define** a car as a street vehicle which is NOT a moto-bike, or a bus, or a track.
- I can **describe** a sport car as a car which have a certain shape and goes fast.

Language – Observations

Observation (Facts, percepts). Facts define what is the case in a model as a composition of percepts, as they occur in the reference domain (see lectures before). These percepts and facts are not decomposable. We call them **atomic percepts** and **atomic facts**.

Observation (Complex percepts and facts). We distinguish the percepts and facts denoted by the LoD language between (**atomic) percepts** and **facts** and **complex percepts** and **facts**, i.e., combinations of them.

Observation (Etype and composite complex percepts). We distinguish **complex percepts** into (complex) **etype percepts**, obtained by defining a new etype based on its properties, and (complex) **composite etype percepts**, obtained by composing etypes.

Observation 1 (Language and domains). The LoD language allows for the construction of percepts (step 1, 2) and facts (step 3) which are not perceived. It creates a linguistic mental representation reality which does not map into the "reality" of what is being perceived, i.e. the analogic mental representation.

Observation 2 (Language and domains). The possibility of giving a name to the newly defined etypes allows for the generation of an unbound number of etypes and facts.

Observation (Language heterogeneity). LoD enables an expressiveness which approximates the complexity of human natural language. The mapping with the perceived reality further extended via a new many-to-many relation.

Interpretation function – Observations

Observation (Three LoD sub-languages composed in a single language). Each of the three languages generates, via appropriate formation rules, more complex formulas using the formulas generated by the previous language as elements of the alphabet.

Observation (Three nested interpretation functions). Each language has its own interpretation function which uses the output of the interpretation function of the language one level below as its own input. We have nested interpretation functions.

Observation (Domain in extension via language constructs). The first two interpretation functions generate (formulas denoting) complex (etype and composite etype) percepts. The third generates complex assertions generating new facts.

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- **The language of etype percepts**
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

Step 1 – The language of etype percepts

Definition (Language of etype percepts, L_T)

$$L_T = \langle A_T, FR_T \rangle = \{p_T\}$$

where:

- L_T is a **language** naming atomic etype percepts
- A_T is an **alphabet**
- FR_T is a **set of formation rules**
- $\{p_T\}$ is the set of type percepts p_T (dtypes plus atomic and complex etypes) obtained by the exhaustive application of FR_T to A_T (the transitive closure $FR_T(A_T)$ of FR_T applied to A_T).

Alphabet

Definition (Alphabet A**)***

$$A_T = < \{T\}, \{P\} >$$

where:

- $\{T\} = \{E_i\} \cup \{D_i\}$ is a set of unary predicates standing for **etypes** and **dtypes**;
- $\{P\} = \{O_i\} \cup \{A_i\}$ is a set of binary **properties**, where O_i is an **object property**, also called a **role**, and A_i is an **attribute**.

Observation (Alphabet of percepts). Similarly to LoE, A_T is an aphabet which denotes percepts in the domain (but denoting a different set of percepts).

*The elements of the alphabet are written in *italic* to distinsguish them from percepts

Formation rules – BNF

< $p_{\top}$ > ::= <etype> | <dtype> | ⊤ | ⊥

< etype > ::= ∃<objProp> .<etype> |
∃<dataProp>.<dtype> |
∀<objProp> .<etype> |
∀<dataProp> .<dtype>

<etype> ::= E_1 | ...| E_n

<dtype> ::= D_1 | ...| D_n

<objProp> ::= O_1 | ...| O_n

<dataProp> ::= A_1 | ...| A_n

Observation (BNF). This BNF does allow the iterative application of the formation rules on etypes (dtypes cannot be changed). It allows for the generation of etype percepts of any depth.

Observation (BNF). Entities are not mentioned (not part of the language). They are referred implicitly via the existential quantifier and also, somehow via the universal quantifier.

Etype percepts – Example

- ⊤ - to be read **Top**
  - *(**Intuition:** the set of all entities)*
- ⊥ - to be read **Bottom**
  - *(**Intuition:** the empty set of entities)*
- Person
  - *(**Intuition:** the set of entities which are called called persons)*
- ∃hasFriend.Person
  - *(**Intuition:** the set of entities which have – at least – one friend who is a person)*
- ∀hasFriend.Person
  - *(**Intuition:** the set of entities whose friends are only persons, possily none)*
- Integer
  - *(**Intuition:** the set of Integers)*
- ∃hasApple.Integer
  - *(**Intuition:** the set of entities which have at least one apple)*
- ∀hasApple.Integer
  - *(**Intuition:** the set of entities which have only not halved apples, possibly none)*

Nested etype percepts – Example

- ∃talksTo.(∃hasFriend.Person)
  - ***Intuition:*** *the set of entities which talk – at least once – to entities which have – at least – one friend who is a person*
- ∀talksTo.(∃hasFriend.Person)
  - ***Intuition:*** *the set of entities which talk only to the set of entities which have – at least – one friend who is a person*
- ∃talksTo.((∀hasFriend.Person)
  - ***Intuition:*** *the set of entities which talk – at least once – to the set of entities whose friends are only persons, possily none*
- ∀talksTo.((∀hasFriend.Person)
  - ***Intuition:*** *the set of entities which talk only to the set of entities whose friends are only persons, possily none*

Interpretation of etype percepts

$I_T(\top) = U$, with U the universe of interpretation

$I_T(\bot) = \emptyset$, with $\emptyset$ the empty set

$I_T(E_i) = E_i$

$I_T(D_i) = D_i$

$I_T(\exists P.T) = \{d \in U \mid$ there is an $e \in U$ with $(d, e) \in I_T(P)$ and $e \in I_T(T)\}$

$I_T(\forall P.T) = \{d \in U \mid$ for all $e \in U$ if $(d, e) \in I(P)$ then $e \in I_T(T)\}$

where I_T is the interpretation function of L_T

Observation (Interpretation function). For an intensional view of the interpretation functions for etypes, dtypes, object properties and attributes, follow what done with LoE.

Observation (Interpretation of nested etypes). It is sufficient to interpret the application of the second external quantifier to the etype built via the application of the first quantifier.

Interpretation of etype percepts - Observations

Observation (Existential type). $\exists P.T$ is an **existential etype**. Its interpretation

$$I(\exists P.T) = \{d \in U \mid \text{there is an } e \in U \text{ with } (d, e) \in I(P) \text{ and } e \in I(T)\}$$

is the set of all units d for which there exists a unit e in the codomain of P or type T. $\exists P.T$ defines the etype which is in relation *P* with *T*, not necessarily only with *T* (there could be an e' not in I(*T*)).

Example. ∃hasFriend.Person is the etype of all those entities who have *at least* a friend who is a person.

Interpretation of etype percepts - Observations

Observation (Universal etype). $\forall P.T$ is an **universal etype.** Its interpretation

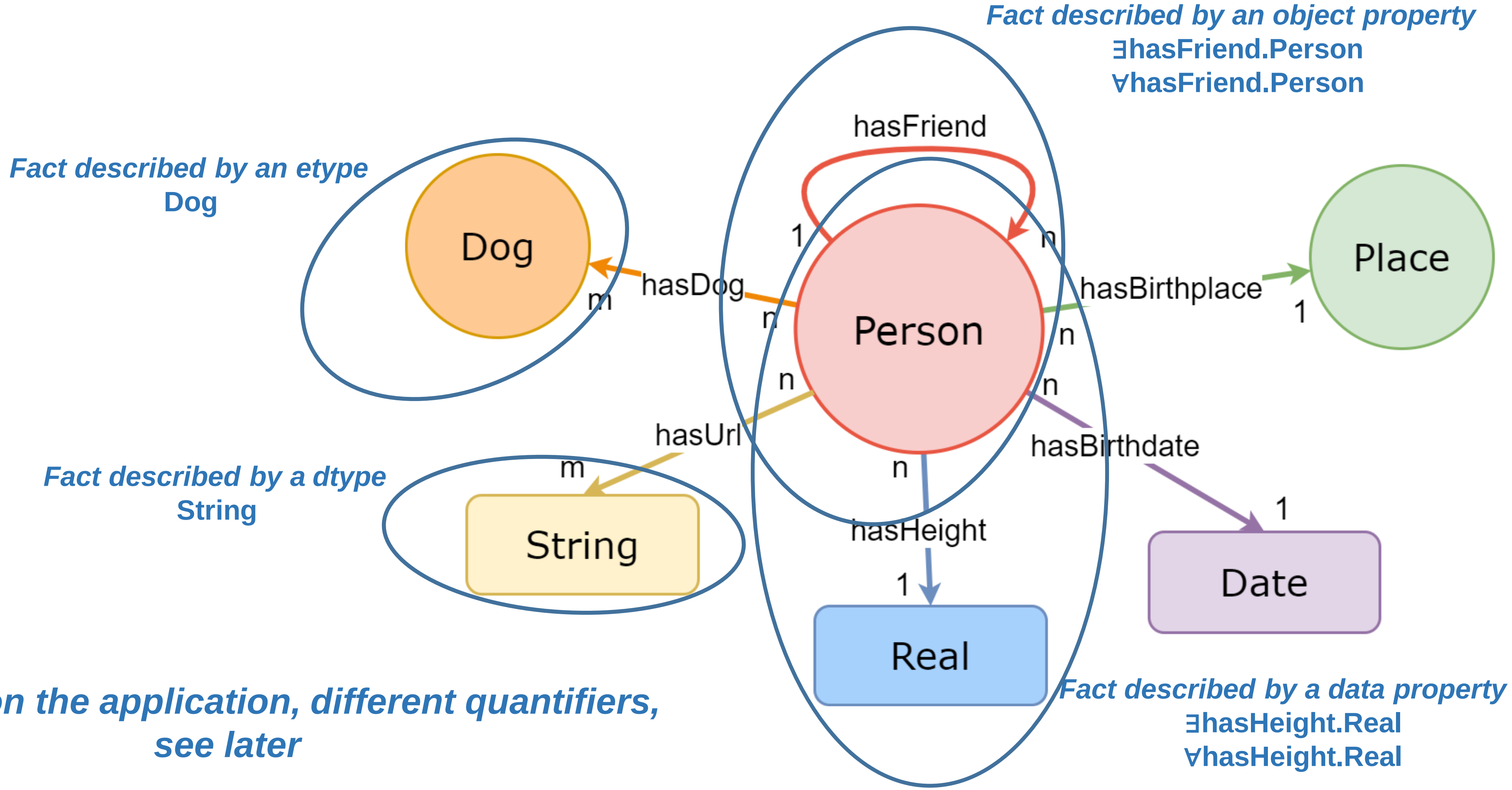
$$I(\forall P.T) = \{d \in U \mid \text{for all } e \in U \text{ if } (d, e) \in I(P) \text{ then } e \in I(T)\}$$

is the set of all units e for which all the units e in the codomain of P are of type T.

Example. ∀hasFriend.Person is the etype of all people whose friends are *only* persons.

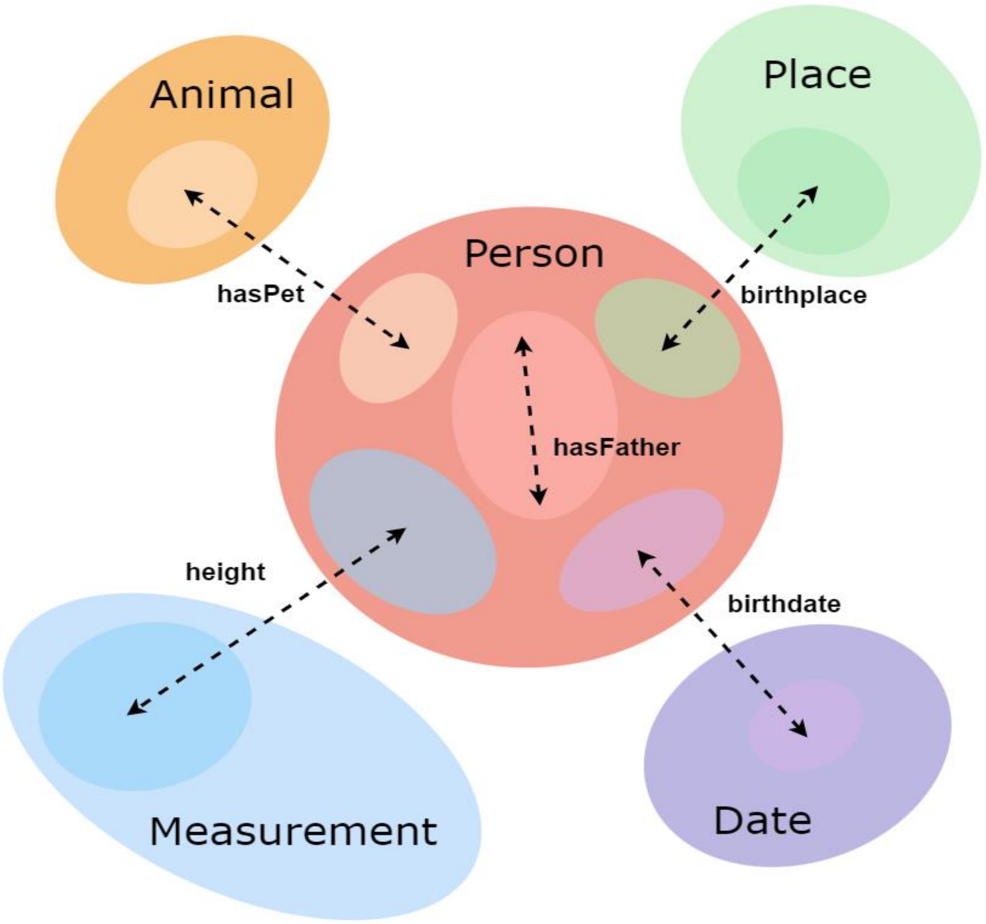
Proposition. $\forall P.T$ does not imply $\exists P.T$. $\forall P.T$ holds if $I(\forall P.T) = \emptyset$, while this is not the case with $\exists P.T$.

Example – how LoD (names of) percepts represent ETG facts



Depending on the application, different quantifiers, see later

Interpretation function (Venn diagram)



Most often, in informal world models we assume both universal and existential quantifiers

The first does not necessarilty imply the second

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- **The language of composite etype percepts**
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

Step 2 – The language of composite etype percepts

Definition (The language of composite etype percepts, L_C**)**

$$L_C = < A_C, FR_C > = \{p_C\}$$

where:

- L_C is a **language** of composite etype percepts
- $A_C = \{p_T\}$, the **alphabet**, consists of all the percepts $p_T \in L_T$
- FR_C is a **set of formation rules**
- $\{p_C\}$ is the set of composite etype percepts a_C obtained by the exhaustive application of FR_C to A_C (the transitive closure $FR_C(A_C)$ of FR_C applied to A_C).

Formation rules – BNF

$$
\begin{aligned}
< p_C > \;\; ::= \; & < p_C > \sqcap < p_C > \; | \\
& < p_C > \sqcup < p_C > \; | \\
& \neg < p_C > \\
< p_C > \; ::= \; & < p_T >
\end{aligned}
$$

Notation (BNF). $< p_C >$ is a nonterminal symbol and it stands for a p_C percept. $< p_T >$ is an L_C terminal symbol and it stands for an L_T percept. See the BNF of L_T to see how to expand it to a LoD terminal symbol.

Observation (BNF). This BNF does allow the iterative application of the formation rules. It allows to generate percepts of any depth.

Composite etype percepts – Example

- Person ⊓ ∃hasFriend.Person
  - *(**Intuition:** the set of entities which are persons and have a friend which is a person)*
- Person ⊔ Dog
  - *(**Intuition:** the set of entities which are a person or a dog)*
- ¬∃hasFriend.Person
  - *(**Intuition:** the set of entities do not have a friend which is a person)*
- Person ⊓ ¬(∃hasFriend.Person)
  - *(**Intuition:** the set of entities which are persons and which do not have a friend which is a person)*

Composite etype percepts – Example etypes

Consider the following concept names:

Vehicle, Boat, Bicycle, Car, Device, Wheel, Engine, Axle, Rotation,

Water, Human, Driver, Adult, Child

Formalize the following natural language statements:

- Nothing (empty set): $\bot$
- Everything (All the interpretation domain): $\top$
- Humans which are drivers : Human $\sqcap$ Driver
- Humans and vehicles: Human $\sqcup$ Vehicle
- Vehicles which are not boats: Vehicle $\sqcap \neg$ Boat
- Wheels or engines which are used in cars: (Wheel $\sqcup$ Engine) $\sqcap$ Car
- Adults or children: Adult $\sqcup$ Child

Composite etype percepts – Example roles

Consider the previous concept names plus the following role names:

hasPart, poweredBy, capableOf, travelsOn, controls

Formalize in DL the following natural language statements:

1. Those vehicles that have wheels and are powered by an engine
2. Those vehicles that have wheels and are powered by a human
3. Those vehicles that travel on water
4. Those objects which have no wheels
5. Those objects which do not travel on water
6. Those devices that have an axle and are capable of rotation
7. Those humans who control a vehicle
8. The drivers of cars

Composite etype percepts – Example roles (cont)

1. Vehicle $\sqcap$ $\exists$hasPart.Wheel $\sqcap$ $\exists$poweredBy.Engine
2. Vehicle $\sqcap$ $\exists$hasPart.Wheel $\sqcap$ $\exists$poweredBy.Human
3. Vehicle $\sqcap$ $\exists$travelsOn.Water
4. $\forall$hasPart.$\neg$Wheel
5. $\forall$travelsOn.$\neg$Water
6. Device $\sqcap$ $\exists$hasPart.Axle $\sqcap$ $\exists$capableOf.Rotation
7. Human $\sqcap$ $\exists$controls.Vehicle
8. Driver $\sqcap$ $\exists$controls.Car

Interpretation of composite etype percepts

$$I_C(p_1 \sqcap p_2) = I_C(p_1) \cap I_C(p_2)$$
$$I_C(p_1 \sqcup p_2) = I_C(p_1) \cup I_C(p_2)$$
$$I_C(\neg p_1) = U \setminus I_C(p_1)$$
$$I_C(p_T) = I_T(p_T)$$
$$I_T(p_T) = \mathrm{p}_T$$

where:

- I_C is the interpretation function of L_C
- I_T is the intepretation function for L_T, the language of etype percepts.
- p_1, p_2 are composite etype percepts
- p_T (in *italic*) is (the name of an) etype percept denoting the domain percept p_T (not in italic)

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- **The language of descriptions**
- Entailment
- Tell
- Ask – Reasoning problems
- Key notions

Step 2 – The language of Descriptions

Definition (The language of descriptions, L_{LoD})

$$L_{LoD} = < A_{LoD}, FR_{LoD} > = \{a_{LoD}\}$$

where:

- L_{LoD} is a **language** of **assertions**
- $A_{LoD} = \{a_C\}$, the **alphabet**, consists of all the formulas $a_C \in L_C$
- FR_{LoD} is a **set of formation rules**
- $\{a_{LoD}\}$ is the set of assertions a_{LoD} which are obtained by the exhaustive application of FR_{LoD} to A_{LoD} (the transitive closure $FR_{LoD}(A_{LoD})$ of FR_{LoD} applied to A_{LoD}). Each assertion a_{LoD} is called a **(LoD) description**.

(LoD) Descriptions – BNF

$$< a_{\text{LoD}} > ::= < p_{\text{C}} > \sqsubseteq < p_{\text{C}} > \mid < p_{\text{C}} > \equiv < p_{\text{C}} >$$

where:

- $< p_{\text{C}} >$ is a composite etype percept.
- a_{LoD} is a LoD description, an assertion involving two composite etype percepts.

Terminology (LoD Description). A **LoD description** describes how the extensions of two composite etype percepts correlate. It is a constraint which reflects back into the component etypes. We call the first a **subsumption** (description) and the second an **equivalence** (description).

Terminology (Subsumption). $\sqsubseteq$ is a **subsumption relation**. $p_1 \sqsubseteq p_2$ is to be read as p_1 **is subsumed by** p_2, or, vice versa that p_2 **subsumes** p_1.

Terminology (Equivalence). $\equiv$ is an **equivalence relation.** We have

$$p_1 \equiv p_2 \text{ if and only if } p_1 \sqsubseteq p_2 \text{ and } p_2 \sqsubseteq p_1$$

(LoD) Definitions – BNF

$$< a_{\text{LoD}} > ::= < E > \sqsubseteq < p_{\text{C}} > \mid < E > \equiv < p_{\text{C}} >$$
$$< E > ::= E_1 \mid \ldots \mid E_n$$

where:

- $< E >$ is an atomic etype percept (an etype in L_T).
- $E_1 \mid \ldots \mid E_n$ are (names of) etype percpets
- p_{C} is a composite etype percept.
- a_{LoD} is a **LoD definition**.

Terminology (LoD definition). A LoD definition is a LoD description that describes the extension of an atomic etype. It constrains the extension of $< E >$.LoD definitions allow to introduce new etypes by defining their extension.

Terminology (Etype subsumption, etype equivalence). The first definition is an **etype subsumption**. The second is an **etype equivalence**. Equivalences allow to precisely define the extension of $< E >$.

Etype disjointness – Observation

Definition ((Atomic) etype disjointness): A special case of concept subsumption is **etype disjointness**, i.e.,

$$E_1 \sqsubseteq \neg E_2$$

where E_1 is an atomic etype and E_2 is an etype, possibly an atomic etype, also written

$$E_1 \perp E_2$$

If both E_1 and E_2 are atomic etypes, we have an **atomic etype disjointness.**

Observation (Etype disjointness): Strong etype disjointness, i.e.,

$$E_1 \equiv \neg E_2$$

implies that the union of the two etypes is the Universe of interpretation U = {u}.

Observation (Etype disjointness) Etype disjointness states that two etypes (e.g., “car” and “bus”) are disjoint. Disjointness definitions are key eleements in language lexicons.

Etype subsumption – Examples

1. Boats have no wheels
2. Cars do not travel on water
3. Drivers are adults who control cars
4. Humans are not vehicles
5. Wheels are not humans
6. Humans are either adults or children
7. Adults are not children

Etype subsumption – Examples

1. Boat $\sqsubseteq$ $\forall$hasPart.$\neg$Wheel
2. Car $\sqsubseteq$ $\forall$travelsOn.$\neg$Water
3. Driver $\sqsubseteq$ Adult $\sqcap$ $\exists$controls.Car
4. Human $\sqsubseteq$ $\neg$ Vehicle
5. Wheel $\sqsubseteq$ $\neg$ Human
6. Human $\sqsubseteq$ Adult $\sqcup$ Child
7. Adult $\sqsubseteq$ $\neg$Child

Etype equivalence – Examples

1. Cars are exactly those vehicles that have wheels and are powered by an engine
2. Bicycles are exactly those vehicles that have wheels and are powered by a human
3. Boats are exactly those vehicles that travel on water
4. Wheels are exactly those devices that have an axle and are capable of rotation
5. Drivers are exactly those humans who control a vehicle

Etype equivalence – Examples

1. Car ≡ Vehicle ⊓ ∃hasPart.Wheel ⊓ ∃poweredBy.Engine
2. Bicyle ≡ Vehicle ⊓ ∃hasPart.Wheel ⊓ ∃poweredBy.Human
3. Boat ≡ Vehicle ⊓ ∃travelsOn.Water
4. Wheel ≡ Device ⊓ ∃hasPart.Axle ⊓ ∃capableOf.Rotation
5. Driver ≡ Human ⊓ ∃controls.Vehicle

Interpretation of LoD descriptions

$$I(p_1 \sqsubseteq p_2) = I_c(p_1) \subseteq I_c(p_2)$$

$$I(p_1 \equiv p_2) = I_c(p_1) = I_c(p_2)$$

$$= I_c(p_1) \subseteq I_c(p_2) \text{ and } I_c(p_2) \subseteq I_c(p_1)$$

where:

- I is the interpretation function for L_{LoD}
- I_c is the intepretation function for L_c, the language of composite etype percepts

Interpretation of LoD definitions

$$I(E \sqsubseteq p_2) = I_c(E) \subseteq I_c(p_2)$$

$$I(E \equiv p_2) = I_c(E) = I_c(p_2)$$

$$= I_c(E) \subseteq I_c(p_2) \text{ and } I_c(p_2) \subseteq I_c(E)$$

$$I_c(E) = I_T(E) = E$$

where:

- I is the interpretation function for L_{LoD}
- I_c is the intepretation function for L_c, the language of composite etype percepts
- I_T is the intepretation function for L_T, the language of composite etype percepts
- E is an etype, a subset of the universe of interpretation

Assertions and facts

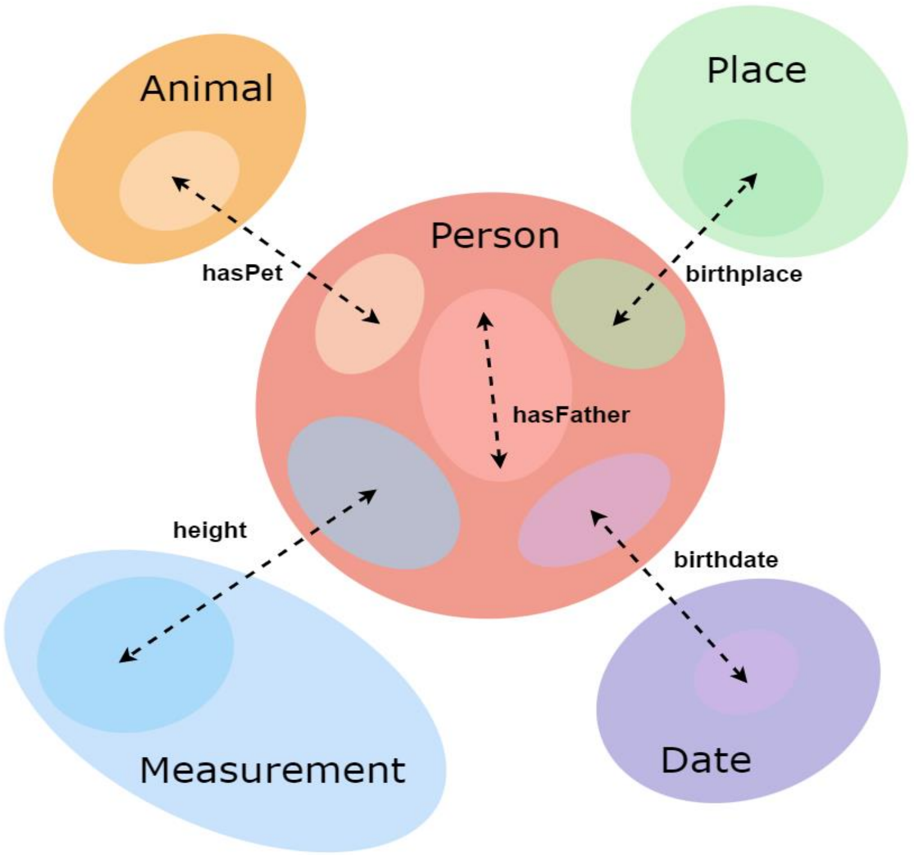
Observation (LoD assertions). LoD assertions (descriptions and definitions) describe the following facts:

- A composite etype p_i is a subset of a composite etype p_j
- A composite etype p_i has the same extension of another composite etype p_j
- A (new atomic) etype E is a subset of a composite etype p
- A (new atomic) etype E has the same extension as a composite etype p.

Assertions and facts only have one of four possible forms:

- The assertion $p_i \sqsubseteq p_j$ describing the fact $p_i \subseteq p_j$
- The assertion $p_i \equiv p_j$ describing the fact $p_i = p_j$
- The assertion $E \sqsubseteq p$ describing the fact $E \subseteq p$
- The assertion $E \equiv p$ describing the fact $E = p$

Interpretation function (Venn diagram)



Which percepts?

Which facts?

It depends!

LoD descriptions and definitions structure the intended model

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- **Entailment**
- Tell
- Ask – Reasoning problems
- Key notions

Entailment

$$M \models p_1 \sqsubseteq p_2 \quad \text{iff} \quad I(p_1) \subseteq I(p_2)$$

$$M \models p_1 \equiv p_2 \quad \text{iff} \quad I(p_1) = I(p_2)$$

$$\text{iff} \quad I(p_1) \subseteq I(p_2) \text{ and } I(p_2) \subseteq I(p_1)$$

with p_1, $p_1 \in L_{LoD}$.

Entailment (extended language)

$M \models p_1 \sqsubseteq p_2$ iff $I(p_1) \subseteq I(p_2)$

$M \models p_1 \equiv p_2$ iff $I(p_1) = I(p_2)$

iff $I(p_1) \subseteq I(p_2)$ and $I(p_2) \subseteq I(p_1)$

$M \models p_1 \sqsupseteq p_2$ iff $I(p_2) \subseteq I(p_1)$

$M \models p_1 \perp p_2$ iff $I(p_1) \cap I(p_2) \subseteq \emptyset$

with

- $p_1, p_2 \in L_{LoD}$;
- $p_1 \sqsupseteq p_2$ a notational variant of $p_2 \sqsubseteq p_1$;
- $p_1 \perp p_2$ a notational variant of $p_1 \sqsubseteq \neg p_2$.

Observation (Entailment language): The entailment (query) language can be extended / changed as long as it is possible to perform the translation into the underlying logic language in **polynomial time**. The underlying algorithm remains the same.

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- **Tell**
- Ask – Reasoning problems
- Key notions

Tell – Model building

Intuition (Model building). Assume we have a logic LoD as from above. Then we can build the intended model M, via a sequence of one or more Tell operations, by asserting a theory $T_a = \{a\}$ which constructs (via the interpretation function) a LoD representation

$$R = \langle T_a, M \rangle$$

with

$$M = \{f\} \subseteq D$$
$$T_a = \{a\} \subseteq L_a$$

where M is the intended **model** of T_a.

Terminology (Tell, TellW, TellA). The basic operation of declaring that a certain LoD assertion is an axiom is a TellA. When this axiom is actually a definition of a new etype, then the Tell operation is both a TellW and a TellA.

Terminology (TBox). In LoD the assertional theor $T_a = \{a\}$ generated via one or more TellT operations is called a **TBox** (see later for a detailed account).

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- **Ask – Reasoning problems**
- Key notions

Ask – Reasoning problems - Observations

Observation (AskC). Differently from LoE, model checking in LoD cannot be reduced to checking whether the assertion of the input query is present, modulo synonyms, in the TBox. Infact, because of the properties of sets (e.g., the DeMorgan laws), the same set can be constructed using an unbound number of set-theoretic expressions. Model checking in LoD is a search problem, implemented by reasoning **search algorithms**.

Observation (AskS). Similarly to LoE, the language of LoD allows only for positive assertions. AskS is useless as it always returns a positive answer. All the LoD reasoning problems are AskC problems.

Notation (AskC, TBox). Historically, from a notation point of view all the LoD model checking problems are formulated as

$$T_a \models a$$

where T_a, called the **TBox**, is the assertional theory, built via a sequence of Tell operations – see above, which is used to build the intended model.

Ask – Reasoning problems

Observation (LoD reasoning problems). The four LoD core reasoning problems are:

- T |= C, **Satisfiability with respect to a TBox T**
- T |= C ⊑ D, **Subsumption with respect to a TBox T**
- T |= C ≡ D, **Equivalence with respect to a TBox T**
- T |= C ⊥ D, **Disjointness with respect to a TBox T**

where T can also be empty.

Observation (LoD reasoning problems). The problems listed above are the ones which historically have been identified as the most important. They all relate to the need of making explicit the semantics of the words used in the definitions of the schemas of DBs (and, more recently, KGs).

Observation (Empty TBox). When the TBox is empty LoD reasoning reduces to proving basic set-theoretic assertions.

Satisfiability

Definition (Satisfiability with respect to a TBox T). An assertion C is **satisfiable** with respect to a TBox T if there exists an interpretation function I of T such that I(C) is not empty.

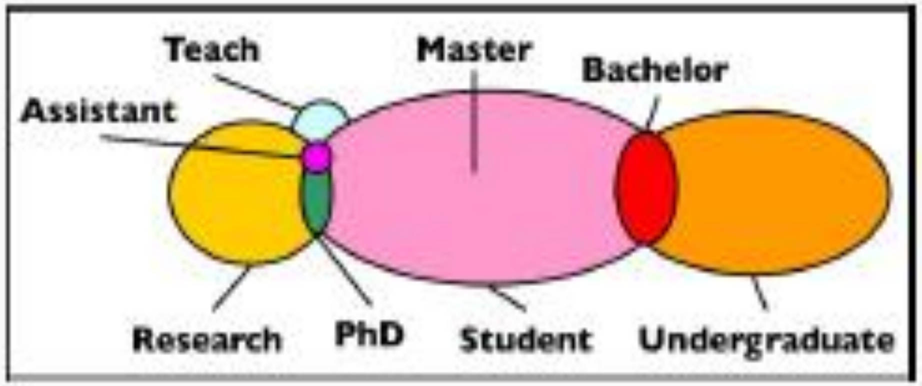
In this case we also say that such an interpretation function I is a **model** of C, with respect to T.

Observation (Satisfiability with respect to a TBox T). As previously discussed this is an AskC problem. However this problem is somehow related, fro an intuitive point of view with AskS (form which its name). In fact it deals with the problem that allowing for the set theoretic negation, i.e., disjointness one can create empty sets. The key point is that we may have a model of the world where a set is empty.

Satisfiability – via Venn diagrams

Consider the Tbox

$$T = \begin{cases} Undergraduate \sqsubseteq \neg Teach \\ Bachelor \equiv Student \sqcap Undergraduate \\ Master \equiv Student \sqcap \neg Undergraduate \\ PhD \equiv Master \sqcap Research \\ Assistant \equiv PhD \sqcap Teach \end{cases}$$



How to generate all possible diagrams?

Bachelor ⊓ *PhD* satisfiable?

NOTE: the diagram on the right is only one of the many possible ones.

To prove the goal one would have to generate **all** the possible diagrams till the satisfying interpretation function has been found, or all of them have been tried out. In the latter case the input formula is not satisfiable.

Satisfiability – via search

Example 1

Is Bachelor ⊓ PhD satisfied by $\mathcal{T}$? (**No**)

The problem can be formalized as:

$$\mathcal{T} \models Bachelor \sqcap PhD$$

Proof:

Bachelor ⊓ PhD
≡ (Student ⊓ Undergraduate) ⊓ (Master ⊓ Research)
≡ (Student ⊓ Undergraduate) ⊓ ((Student ⊓ ¬ Undergraduate) ⊓ Research)
≡ Student ⊓ **Undergraduate** ⊓ ¬ **Undergraduate** ⊓ Research
≡ Student ⊓ ⊥ ⊓ Research

Observation (Satisfiability via search). Note how in this proof, and all the following, the reasoning strategy is to use definitions to expand the goal till one arrives to ⊥ or to terms which can be no longer expanded. Disjunctions, generate alternative expansions, and therefore search for the "correct" one.

Subsumption

Definition (Subsumption with respect to a TBox T). An assertion C is subsumed by an assertion D with respect to a TBox T if

$$I(C) \subseteq I(D)$$

for every interpretation function I. In this case we write

$$C \sqsubseteq_T D$$

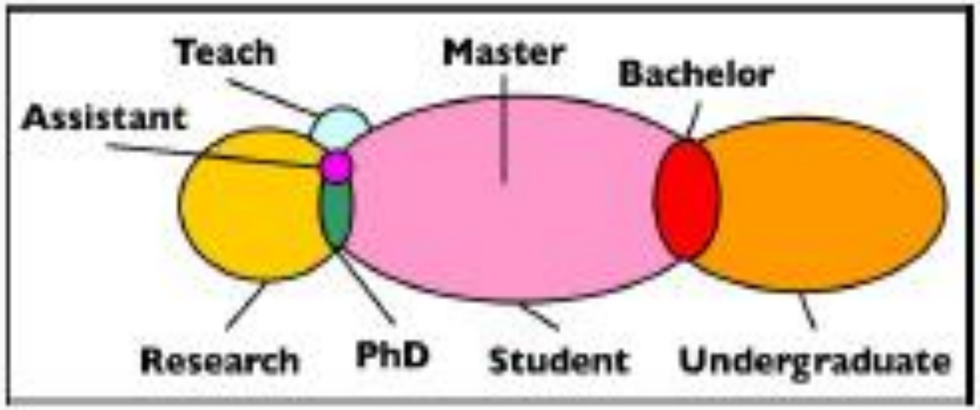
or also

$$T \models C \sqsubseteq D$$

Subsumption – via Venn diagrams

Consider the Tbox

$$T = \begin{cases} Undergraduate \sqsubseteq \neg Teach \\ Bachelor \equiv Student \sqcap Undergraduate \\ Master \equiv Student \sqcap \neg Undergraduate \\ PhD \equiv Master \sqcap Research \\ Assistant \equiv PhD \sqcap Teach \end{cases}$$



T |= *PhD* ⊑ *Student*

It should be checked for all models of T

Subsumption – via search

Example 2

Is PhD $\sqsubseteq$ Student satisfiable? (Yes)

The problem can be formalized as:

$$\mathcal{T} \models PhD \sqsubseteq Student$$

Proof:

PhD

$\equiv$ Master $\sqcap$ Research

$\equiv$ (Student $\sqcap$ $\neg$ Undergraduate) $\sqcap$ Research

$\sqsubseteq$ Student

Equivalence

Definition(Equivalence with respect to a TBox T). Two assertions C and D are equivalent with respect to a TBox T if

$$I(C)= I(D)$$

for every interpretation function I. In this case we write

$$C \equiv_T D$$

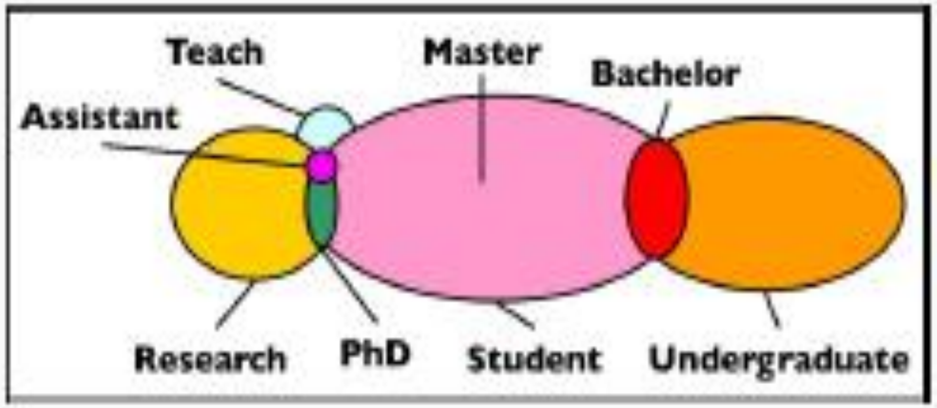
or also

$$T \models C \equiv D$$

Equivalence – via Venn diagrams

Consider the Tbox

$$
\mathcal{T} = \begin{cases}
Undergraduate \sqsubseteq \neg Teach \\
Bachelor \equiv Student \sqcap Undergraduate \\
Master \equiv Student \sqcap \neg Undergraduate \\
PhD \equiv Master \sqcap Research \\
Assistant \equiv PhD \sqcap Teach
\end{cases}
$$



T |= *Student ≡ Bachelor ⊔ Master*

It should be checked
for all models of T

Equivalence – via search

Example 3

$$\text{Is Student} \equiv \text{Bachelor} \sqcup \text{Master consistent with } \mathcal{T}\text{? (Yes)}$$

The problem can be formalized as:

$$\mathcal{T} \models Student \equiv Bachelor \sqcup Master$$

Proof:

$\text{Bachelor} \sqcup \text{Master}$

$\equiv (\text{Student} \sqcap \text{Undergraduate}) \sqcup (\text{Student} \sqcap \neg\, \text{Undergraduate})$

$\equiv \text{Student} \sqcup (\text{Undergraduate} \sqcap \neg\, \text{Undergraduate})$

$\equiv \text{Student} \sqcup \top$

$\equiv \text{Student}$

NOTE: the symbol T in the line before the last shold be substituted with $\bot$

Disjointness

Definition (Disjointness with respect to a TBox T). Two assertions C and D are disjoint with respect to T if

$$I(C) \cap I(D) = \emptyset$$

for every interpretation function I. In this case we write

$$C \perp_T D$$

or also

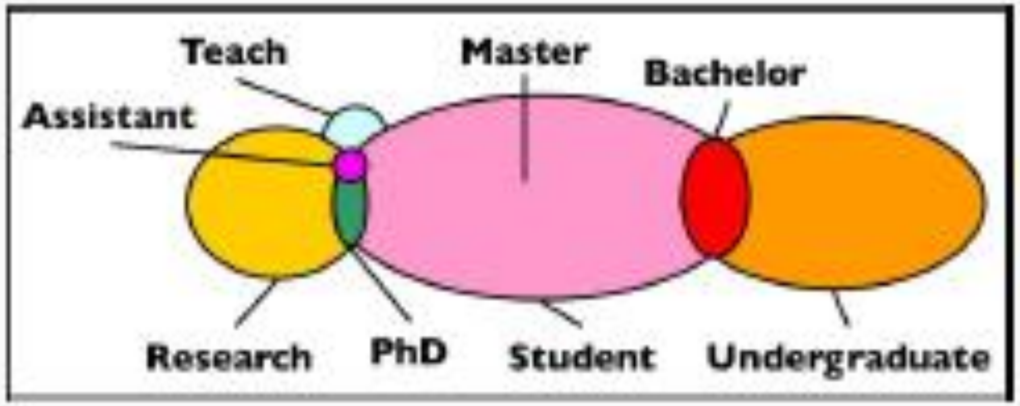
$$T \models C \perp D$$

Disjointness – via Venn diagrams

Consider the Tbox

$$\mathcal{T} = \begin{cases} Undergraduate \sqsubseteq \neg Teach \\ Bachelor \equiv Student \sqcap Undergraduate \\ Master \equiv Student \sqcap \neg Undergraduate \\ PhD \equiv Master \sqcap Research \\ Assistant \equiv PhD \sqcap Teach \end{cases}$$

$T \models Undergraduate \sqcap Assistant \sqsubseteq \bot$



It should be checked
for all models of T

Disjointness – via search

Example 4

Is Undergraduate $\sqcap$ Assistant $\sqsubseteq \bot$ consistent with $\mathcal{T}$? (Yes)

The problem can be formalized as:

$$\mathcal{T} \models \mathit{Undergraduate} \sqcap \mathit{Assistant} \sqsubseteq \bot$$

Proof:

Undergraduate $\sqcap$ Assistant
$\sqsubseteq \neg$ Teach $\sqcap$ Assistant
$\equiv \neg$ Teach $\sqcap$ (PhD $\sqcap$ Teach)
$\equiv \bot \sqcap$ PhD
$\equiv \bot$

Reasoning problems (Reduction)

Proposition (Reduction to satisfiability). All the problems reduce to satisfiability. In fact, we have the following eqiuvalences:

- **Equivalence**: $C \equiv_T D$ if and only if $C \sqsubseteq_T D$ and $D \sqsubseteq_T C$;
- **Subsumption**: $C \sqsubseteq_T D$ if and only if $C \sqcap \neg D$ is *unsatisfiable* with respect to T;
- **Disjointness:** $C \perp_T D$ if and only if $C \sqcap D$ is *unsatisfiable* with respect to T.

Proposition. LoD satisfiability can be reduced to propositional satisfiability (see later, the logic LoP).

LoD – The Logic of descriptions

- Intuition
- Definition
- Domain
- Language – intuition
- The language of etype percepts
- The language of composite etype percepts
- The language of descriptions
- Entailment
- Tell
- Ask – Reasoning problems
- **Key notions**

Key notions

- LoD as a logic of etypes
- Etype graphs (ETGs)
- Language of etype percepts
- Nested etype percepts
- Language of composite etype percepts
- LoD assertions
- LoD facts
- Language of descriptions
- Descriptions
- Definitions
- Entailment
- Tell – TBoxes
- (Strongly) definitional TBox
- Acyclic TBox
- Terminology
- Ask – reasoning problems
- Satisfiability
- Subsumption
- Equivalence
- Disjointness
- Reasoning problem reducibility

LoD
The logic of Descriptions
(HP2T)