



LoDE - The logic of Entity Bases (EBs) (HP2T)

LoDE – The Logic of Entity Bases

- **Intuition**
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions

LoDE – Why a logic of Entity Bases?

- The **Logic of Entities (LoDE)** is the logic formalizing the reasoning implemented inside Entity Bases, the most expressive approach to the formalization of knowledge in reasoning systems
- LoDE formalizes reasoning with Knowledge Graphs (KGs), taking into account the information encoded in the words used to build KGs.
- With respect to LoE, LoDE presents two main improvements
 - It allows to represent the meaning of words as encoded in Genus-Differentia LoD definitions
 - It allows to describe the data and object properties of etypes as encoded in LoD descriptions

LoDE – Highlights

- LoDE is a world logic, with both a graph linguistic / analogic representation.
- LoDE is the most expressive world logic which allows for a LoE-like KG representation of the information it encodes (modulo the disjointness information encoded in the differentia disjointness assertions which is usually not represented in KGs).
- It is conceptually similar in spirit to a earlier logic obtained from the union of the LoE Assertion Box (ABOX), what we call the EG, and the TBox terminologies of Description Logics (DL). The move is from DBs to KGs.

LoDE - Components

- The language used to name the genus and differentia which allow to define LoD concepts (LoD concept definitions)
- The language used to name and describe complex etypes (LoD etype descriptions)
- The knowledge used to constrain the properties of the etypes (LoD assertions)
- The names of the entities, and their etypes which are the case in the world (LoE entity names)
- The (data and object) properties which describe how entities correlate in the world (LoE assertions)
- Extended LoE Ask/Tell reasoning capabilities about entities and their properties

Entity Bases

The original use of the term knowledge base was to describe one of the two sub-systems of an **expert system** (a **knowledge-based system**).

As from [*] **knowledge-based system** consists of

- a **knowledge-Base (KB)** representing ***facts***** about the world and
- ways of **reasoning** about those ***facts*** to deduce new facts or highlight inconsistencies.

We call an **Entity Base (EB)** a KB where assertions are stored as an **Entity Graph (EG)**.

* Hayes-Roth, Frederick; Donald Waterman; Douglas Lenat (1983). Building Expert Systems. Addison-Wesley.

** That is, ***assertions denoting facts***

EB – Informal definition

$$EB = \langle L, T, A \rangle$$

where:

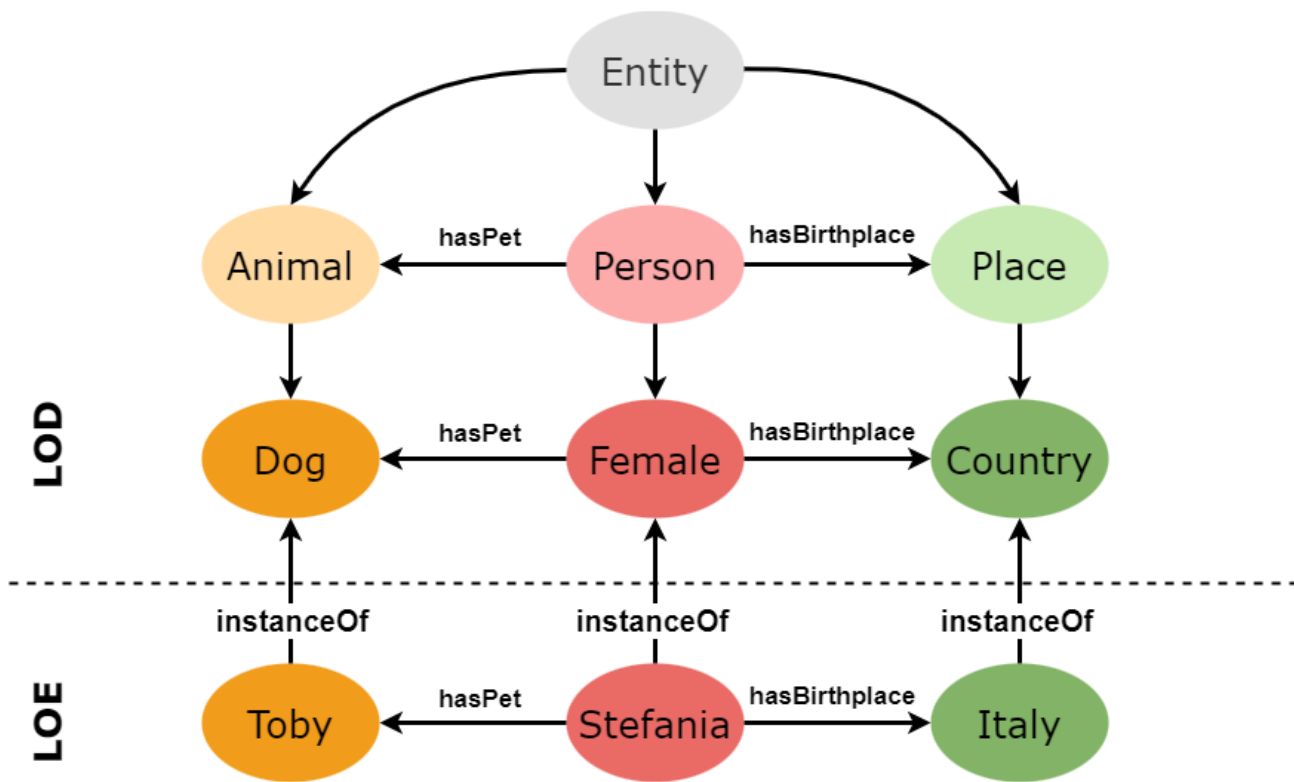
- **L** (for **L**exicon) is a terminology which formalizes the meaning of words into a lexicon that we use to describe the world. We formalize **L** in LOD.
- **T** (for **T**eleontology) is a terminology which formalizes the background (commonsense, scientific, ...) knowledge of the world. We formalize **T** in LOD.
- **A** (for **A**ssertions) is an assertional theory which models our knowledge of the world as we perceive it, as we are told about it, as we infer about it. We formalize **A** in LOE.

LOD and LOE provide, respectively, the means for reasoning about **T** and **A** independently.

LODE provide the means for reasoning about **A** based on the knowledge encoded in **T**.

An example of EB Components

A populated knowledge teleontology



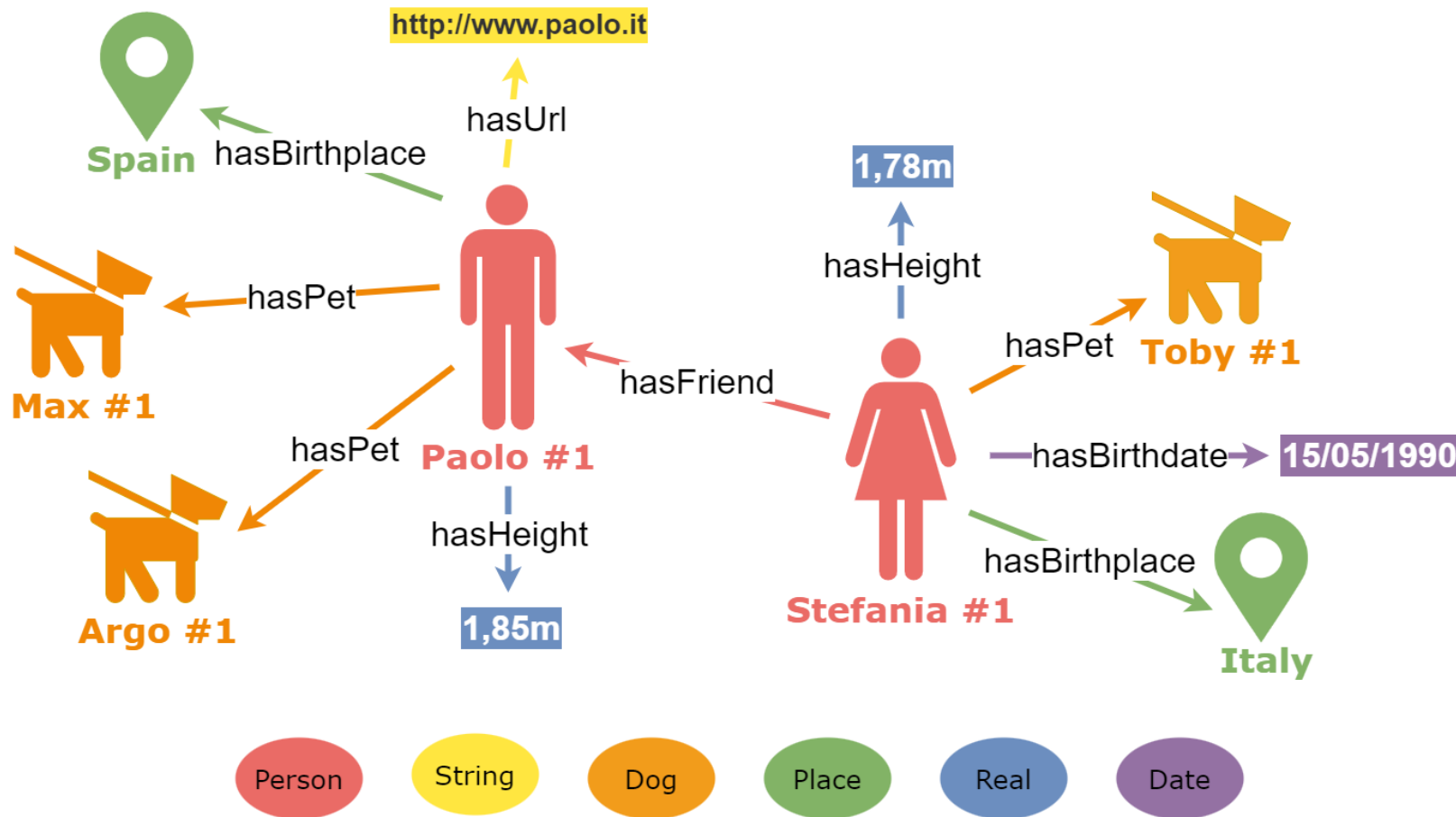
- Which LoD definitions?
- Which LoD descriptions?
- Which LoE assertions?

Could we have an EG encoding the following assertions?

- Person (Lucia)
- HasBirthplace (person#1, Italy)
- Person(Stefania)
- HasPet(Stefania, Toby)
- HasPet(Stefania, Dog#2)

Observation. In an EB the goal is to provide answers about the properties of entities in the LOE EG. The LOD knowledge component is used «only» to make assertions informed by the LOD implicit knowledge.

An example of LoDE EG



All the properties and Etypes are flattened via (LoD) unfolding and then (LoDE) expansion.

The result is LoE EG where all the language implicit information is made explicit in the graph nodes and links.

Well-formedness conditions (the same as LoE)

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

- Each node is associated to one and only ONE entity or value;
- Each node is associated to 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 data value nodes.

LoDE – The Logic of Entity Bases

- Intuition
- **Definition**
- Domain
- Language
- Interpretation function
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions

LoDE – Definition

We formally define LoDE as follows

$$\text{LoDE} = \langle \text{EG}_{\text{LoDE}}, \models_{\text{LoDE}} \rangle$$

with

$$\text{EG}_{\text{LoDE}} = \langle L_{\text{LoDE}}, D_{\text{LoDE}}, I_{\text{LoDE}} \rangle$$

Below, any time no confusion arises, we drop the subscripts.

Observation (LoDE). LoDE is the Logic of the Entities together with their Definitions and Descriptions

LoDE – Definition (continued)

Observation (LoDE). $EG_{LoDE} = \langle L_{LoDE}, D_{LoDE}, I_{LoDE} \rangle$ is constructed by suitably composing

$$EG_{LoE} = \langle L_{LoE}, D_{LoE}, I_{LoE} \rangle$$

and

$$ETG_{LoD} = \langle L_{LoD}, D_{LoD}, I_{LoD} \rangle$$

under the following assumptions:

- $D_{LoDE} = D_{LoE}$
- The etypes and dtypes in L_{LoE} and L_{LoD} are the same, both linguistically and in how the interpretation function applies to them

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- **Domain**
- Language
- Interpretation function
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions

Domain (the same as LoE)


Definition (Domain)

$$D = \langle U, \{C\}, \{R\} \rangle$$

where:

- $U = \{u\}$ is called the **universe (of interpretation)** of D .
- $\{u\}$ is a set of **units** u_1, \dots, u_n , for some n
- $\{C\}$ is a set of **classes** C_1, \dots, C_m of units, for some m , with $C_i \subseteq U$
- $\{R\}$ is a set of **binary relations** R_1, \dots, R_p between units, for some p , with $R_i \subseteq U \times U$

Observation (EG, Binary relations). To comply to the graph notation we restrict ourselves to binary relations.



Universe of interpretation (the same as LoE)

Definition (Universe of interpretation)

$$U = E \cup V$$

where:

- $U = \{u\}$ is the **universe of interpretation**;
- $E = \{e\} \subseteq \{u\}$ is the **entity universe**;
- $V = \{v\} \subseteq \{u\}$ is the **value universe**;
- $\{e\}$ and $\{v\}$ are disjoint.

Observation (Entity universe, value universe). E and V can be further divided into smaller sets which can be optionally defined to be mutually disjoint.

Classes (the same as LoE – with one key point)

Definition (Class)

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

where:

- $E_T = \{E_i\}$ 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 (etype). For any etype E_i , $E_i \subseteq E$.

Observation (dtype). For any dtype D_i , $D_i \subseteq V$

Observation (LoE vs LoDE). With respect to LoE, in LoDE some classes are added being the interpretation of defined etypes

Binary relations (the same as LoE)

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). Dtypes can only be leaves of an EG. Any such leaf value can be queried to obtain the desired information

Facts (the same as LoE)

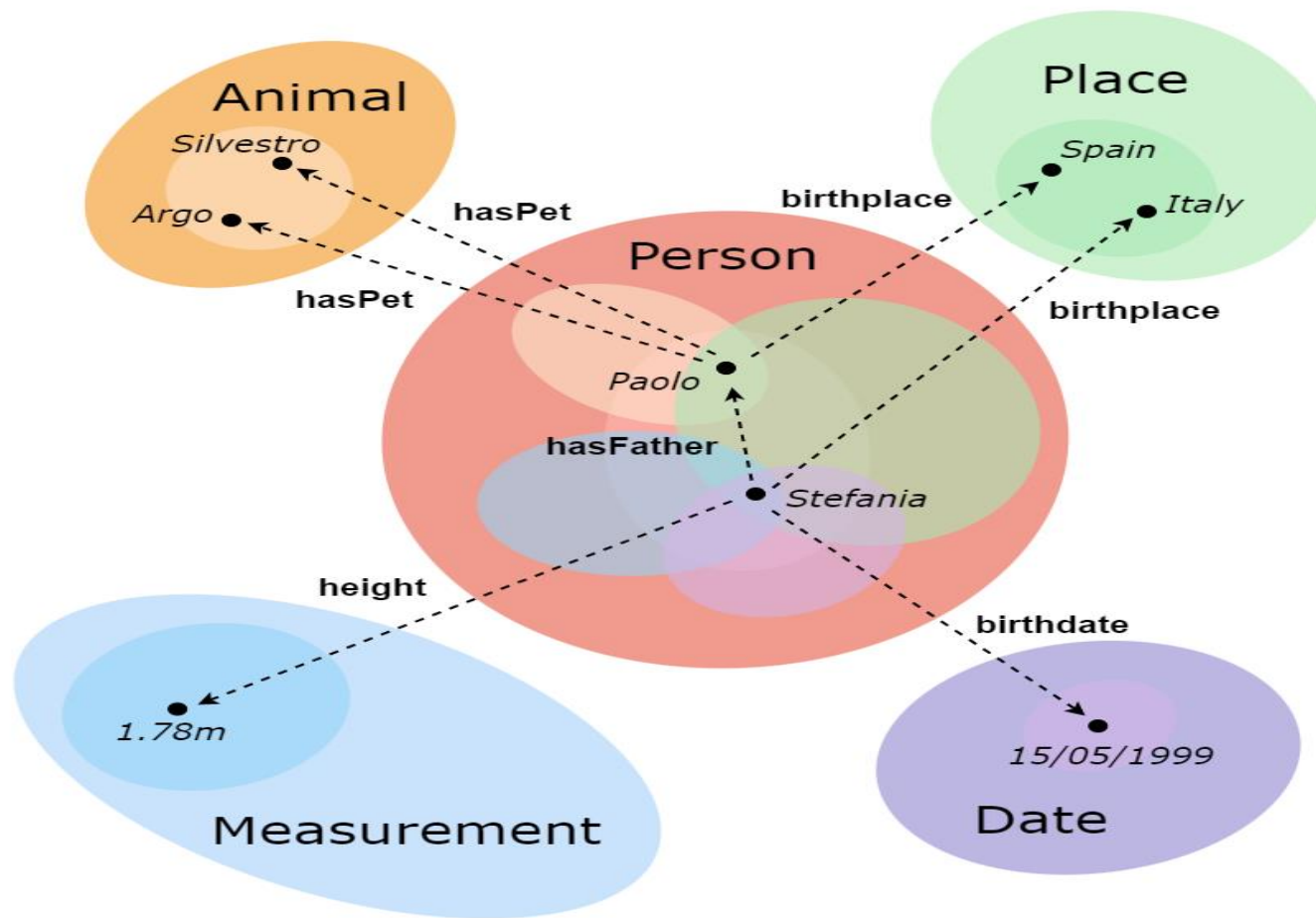
Observation. LoDE allows for the following facts:

- Every etype/ dtype and its argument is a fact
- Every relation R populated by its two arguments is a fact

Facts only have one of four possible forms:

- $e \in E$, that is, the entity e has etype E
- $v \in D$, that is, the value v has dtype D
- $\langle e_1, e_2 \rangle \in O_i$, that is, the object property O_i holds between the two entities e_1, e_2
- $\langle e, v \rangle \in A$, that is, the attribute A of entity e has value v

An example of EG – Venn diagram (the same as LoE)



LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- **Language**
- Interpretation function
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions

Language (the same as LoE)

Definition (Assertional language)

$$L_a = \langle A_a, FR_a \rangle = \{a\}$$

where:

- L_a is an **assertional language**
- A_a is an **alphabet** of assertions
- FR_a is a **set of formation rules**
- $\{a\}$ is the set of assertions obtained by the exhaustive application of FR_a to A_a (the transitive closure $FR_a(A_a)$ of FR_a applied to A_a).

Alphabet (the same as LoE)

Definition (Alphabet A)*

$$A_a = \langle E, \{T\}, \{P\} \rangle$$

where:

- $E = \{e\} \cup \{v\}$ is a set of (names of) **entities** e and of values v ;
- $\{T\} = \{E_i\} \cup \{D_i\}$ 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**.

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

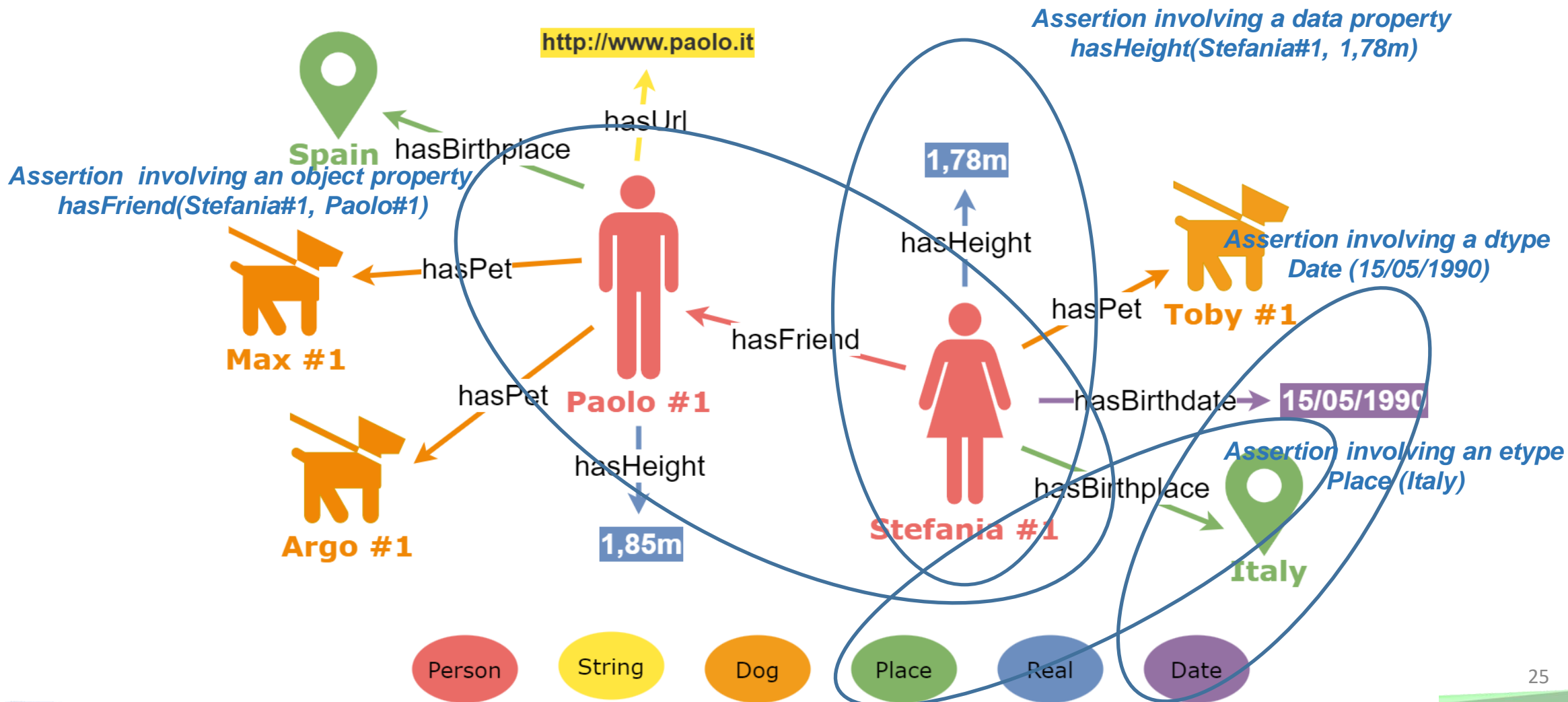


Formation Rules – BNF (almost the same as LoE)

<assertion>	::= <etype>(<nameEntity>)	
	<dtype>(<value>)	
	<objProp>(<nameEntity>, <nameEntity>)	
	<dataProp>(<nameEntity>, <value>)	
<etype>	::= E_1 ... E_n <deflabel> <descrlabel>	
<dtype>	::= D_1 ... D_n	
<objProp>	::= O_1 ... O_n	
<dataProp>	::= A_1 ... A_n	
<nameEntity>	::= e_1 ... e_n	
<value>	::= v_1 ... v_n	

Observation (LoE vs. LoDE). Differently from LoE, in LoDE we have not only primitive etypes, but also defined etypes and descriptions of etypes (in **bold**).

An example of LoDE assertions (the same as LoE)



LoDE assertions – as from BNF (same as facts)

Observation. LoDE allows for the following assertions:

- Every etype (**primitive, defined or described**) / dtype and its argument is an assertion.
- Every relation R and its two arguments is an assertion.

Assertions only have one of four possible forms:

- $E_T(e)$, meaning that the entity e is of etype E_T ,
- $D_T(v)$, meaning that the value v is of dtype D_T
- $O(e_1, e_2)$, meaning that the object property O holds between e_1 and e_2
- $A(e, v)$, meaning that the data property A of entity e has value v

Observation (LoE vs. LoDE). Differently from LoE, in LoDE we have not only primitive etypes, but also defined etypes and descriptions of etypes (in **bold**).

Formation rules – BNF

(a subset of the LoD definitions)

$$\langle a_{\text{LoD}} \rangle ::= \langle \text{deflabel} \rangle \mid \langle \text{descrlabel} \rangle \mid \langle \text{diffconstraint} \rangle$$
$$\langle \text{deflabel} \rangle ::= \langle E \rangle \equiv \langle p_C \rangle$$
$$\langle \text{descrlabel} \rangle ::= \langle E \rangle \equiv \langle p_C \rangle$$
$$\langle \text{diffconstraint} \rangle ::= \langle E \rangle \perp \langle E \rangle$$
$$\langle E \rangle ::= E_1 \mid \dots \mid E_n$$

Observation. As from the previous page, $\langle p_C \rangle$ can only be a conjunction, that is: a genus-differentia definition, or a LoD description.

Formation rules – BNF

(a subset of the LoD composite percepts)

$$\langle p_C \rangle ::= \langle p_C \rangle \sqcap \langle p_C \rangle \mid$$

$$\langle p_C \rangle \sqcup \langle p_C \rangle \mid$$

$$\neg \langle p_C \rangle$$

$$\langle p_C \rangle ::= \langle p_T \rangle$$



Formation rules – BNF (the same as the LoD percepts)

$$\begin{aligned} \langle p_T \rangle &::= \langle \text{etype} \rangle \mid \langle \text{dtype} \rangle \quad \mid \top \mid \perp \\ \langle \text{etype} \rangle &::= \exists \langle \text{objProp} \rangle . \langle \text{etype} \rangle \mid \\ &\quad \exists \langle \text{dataProp} \rangle . \langle \text{dtype} \rangle \mid \\ &\quad \forall \langle \text{objProp} \rangle . \langle \text{etype} \rangle \mid \\ &\quad \forall \langle \text{dataProp} \rangle . \langle \text{dtype} \rangle \\ \langle \text{etype} \rangle &::= E_1 \mid \dots \mid E_n \\ \langle \text{dtype} \rangle &::= D_1 \mid \dots \mid D_n \\ \langle \text{objProp} \rangle &::= O_1 \mid \dots \mid O_n \\ \langle \text{dataProp} \rangle &::= A_1 \mid \dots \mid A_n \end{aligned}$$

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- **Interpretation function**
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions



LoDE – Interpretation function

Definition (Interpretation function I_{LoDE}):

$$I_{\text{LoDE}} = \langle I_{\text{LoD}}, I_{\text{LoE}} \rangle$$

Observation: I_{LoD} is first applied to compute the extension the of defined etypes, which are then used to compute the semantics of assertions (via I_{LoE}).

Interpretation of assertions (same as LoE)

Definition (Interpretation function, alphabet). The LoE interpretation function I is defined as

$$I_{\text{LoE}} : L \rightarrow D$$

where I is defined as

$$I_{\text{LoE}} = \langle I_U, I_T, I_P \rangle$$

where:

- $I_U = \langle I_e, I_v \rangle$, the universe interpretation function, is the pair of the entity and the value interpretation function;
- $I_T = \langle I_E, I_D \rangle$, the Type interpretation function, is the pair of the etype and the dtypes interpretation function;
- $I_P = \langle I_O, I_A \rangle$, the Property interpretation function, is the pair of the object property and the attribute interpretation function

with:

$$I_e : \{e\} \rightarrow \{e\}$$

$$I_E : \{E_i\} \rightarrow \{E_i\}$$

$$I_O : \{O_i\} \rightarrow \{O_i\}$$

$$I_v : \{v\} \rightarrow \{v\}$$

$$I_D : \{D_i\} \rightarrow \{D_i\}$$

$$I_A : \{A_i\} \rightarrow \{A_i\}$$

Interpretation of assertions (same as LoE)

Definition (LoE interpretation function, assertion). The interpretation of assertions into facts is defined as follows:

$$\begin{aligned} I_{\text{LoE}}(E_i(e)) &= I_E(E_i)(I_e(e)) &= e \in E_i \\ I_{\text{LoE}}(D_i(v)) &= I_D(D_i)(I_v(v)) &= v \in D_i \\ I_{\text{LoE}}(O_i(e_1, e_2)) &= I_O(O_i)(I_e(e_1), I_e(e_2)) &= \langle e_1, e_2 \rangle \in O_i \\ I_{\text{LoE}}(A_i(e, v)) &= I_A(A_i)(I_e(e), I_v(v)) &= \langle e, v \rangle \in A_i \end{aligned}$$

Observation. Notice how the interpretation of assertions first applies the interpretation function (i.e., I_{LoE}) which then, **compositionally**, applies the proper component specific interpretation function mimicking, step by step how syntax composes (e.g., in the first assertion, I_E and I_e).

Notation. The words of the alphabet are in “*italic*”, percepts are in “roman”

Interpretation of etype percepts (same as LoD)

$I_{\top}(T) = U$, with U the universe of interpretation

$I_{\top}(\perp) = \emptyset$, with \emptyset the empty set

$I_{\top}(E_i) = E_i$

$I_{\top}(D_i) = D_i$

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

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

where I_{\top} is the interpretation function of L_{\top} , the LoD language of primitive etypes and dtypes.

Interpretation of composite etype percepts (a subset of LoD)

$$I_C(p_1 \sqcap p_2) = I_C(p_1) \cap I_C(p_2)$$

$$I_C(p_T) = I_T(p_T)$$

$$I_T(p_T) = p_T$$

where:

- I_C is the interpretation function of L_C , the LoD language for composite types
- I_T is the interpretation function for L_T , the LoD language of primitive etypes.
- 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*)

Interpretation of LoD descriptions

$$I_{\text{LoD}}(E \equiv p_2) = I_C(E) = I_C(p_2)$$

$$I_{\text{LoD}}(E_1 \perp E_2) = I_C(E_1) \cap I_C(E_2) = \emptyset$$

where:

- I_{LoD} is the interpretation function for L_{LoD}
- I_C is the interpretation function for L_C , the language of composite etypes

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- **Entailment**
- Etype expansion
- Tell
- Ask – Reasoning problems
- Key notions

Entailment relation

Definition (Entailment \models)

$$M \models a \iff I(a) \in M$$

with $a \in L_a$.

Observation: The same of LoE, but (!) on an EG hugely extended with defined etypes. Entailment cannot be computed by just checking whether the interpretation of an assertion is in the model.

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- **Etype expansion**
- Tell
- Ask – Reasoning problems
- Key notions

Etype expansion

Definition (Etype expansion) Let T be an **unfolded** LoDE definitional TBox and A a LoDE ABox. Let C be an etype defined in T (after unfolding) as

$$C \equiv C_1 \sqcap \dots \sqcap C_n$$

where C_1, \dots, C_n are conjuncts containing only primitive etypes.

Let e be an entity occurring in A . Assume $C(e)$ occurs in A . Then the **expansion** $\text{exp}(C(e))$ of $C(e)$ in A in T is defined as follows:

$$\text{exp}(C(e)) = \{C(e), \text{exp}(C_1(e)), \dots, \text{exp}(C_n(e))\}$$

Observation (expansion). Expansion is defined recursively. However the process is guaranteed to terminate because the Tbox is acyclic by hypothesis (see lectures on KG theories).

Etype expansion (continued)

Definition (Etype expansion $\exp(C_i(e))$) Let $C_i(e)$, be an assertion in A which is a conjunct in the expansion of C . Then the **expansion $\exp(C_i(e))$** of $C_i(e)$ is defined as follows

- If C_1 is an etype C . Then

$$\exp(C(e)) = C(e)$$

- If C_1 is of the form $\exists P.C$. Then

$$\exp(\exists P.C(e)) = \{P(e, e_2), C(e_2)\}$$

with e_2 new in A .

- If C_1 is of the form $\forall P.C$, with $P(e, e_2)$. Then

$$\exp(\forall P.C(e)) = C(e_2)$$

Etype expansion (observations)

Observation (etyp expansion). The expansion of a defined etyp applied to an individual allows to expand it into multiple independent assertions, one per conjunct.

Observation (etyp conjunct expansion). The expansion of conjunct consisting of an etyp applied to an individual is the type itself.

Observation (existential conjunct expansion). The expansion of conjunct consisting of an existential quantification allows to create a new link to an individual which is anonymous. For instance, the expansion of $\exists \text{HasChild.Person}(e1)$ is $\text{HasChild}(e1, \text{anonymous}\#1)$ with $\text{anonymous}\#1$ not occurring in the EG .

Observation (universal conjunct expansion). The expansion of conjunct consisting of a universal quantification allows to know the etyp of the individual in the codomain of the of the property. For instance, the expansio of $\forall \text{HasChild.Person}(e1)$ is $\text{Person}(e2)$ assuming that the link $\text{HasChild}(e1, e2)$ occurs in the EG.

Expansion (main result)

Definition (ABox expansion wrt a TBox). Apply etype expansion to all occurrences of all etypes occurring in the ABox and defined in the terminology T' obtained by unfolding the TBox T . The resulting ABox A' is called **expansion** of A with respect to T .

Theorem. Let T be a terminology and A and an ABox. Let A' be the result of expanding A based on the terminology T' obtained by unfolding T . Then M is a model of $T \cup A$ if and only if M is a model of A' .

$$M \models T \cup A \iff M \models A'$$

Expansion of an unfolded TBox (example 3 (cont))

- T=
- Bachelor $\equiv \text{Student} \sqcap \text{Undergrad}$
 - Master $\equiv \text{Student} \sqcap \neg \text{Undergrad}$
 - PhD $\equiv \text{Student} \sqcap \neg \text{Undergrad} \sqcap \text{Research}$
 - Assistant $\equiv \text{Student} \sqcap \neg \text{Undergrad} \sqcap \text{Research} \sqcap \text{Teach}$

A = {PhD(Rui)}

C = PhD

$\text{Exp}(\text{PhD}(\text{Rui})) = \{\text{PhD}(\text{Rui}), \text{Student}(\text{Rui}), \neg \text{Undergrad}(\text{Rui}), \text{Research}(\text{Rui}), \text{Master}(\text{Rui})\}$

Expansion - example

$T = \{ \text{Mother} \equiv \text{Female} \sqcap \exists \text{HasChild}.\text{Person}, \text{Female} \equiv \text{Person} \sqcap \text{Nice} \}$

$A = \{ \text{Mother}(\text{Anna}) \}$

The expansion A' of A with respect to T is:

$A' = \{ \text{Mother}(\text{Anna}), \text{Female}(\text{Anna}), \text{HasChild}(\text{Anna}, \text{Anonymous\#1}), \text{Person}(\text{Anna}), \text{Person}(\text{Anonymous\#1}), \text{Nice}(\text{Anna}) \}$

Expansion - example

$T = \{ \text{FemaleMother} \equiv \text{Female} \sqcap \forall \text{HasChild.female}, \text{Female} \equiv \text{Person} \sqcap \text{Nice} \}$

$A = \{ \text{FemaleMother}(\text{Anna}), \text{HasChild}(\text{Anna}, \text{Mary}) \}$

The expansion A' of A with respect to T is:

$A' = \{ \text{FemaleMother}(\text{Anna}), \text{Female}(\text{Anna}), \text{Female}(\text{Mary}), \text{Person}(\text{Anna}), \text{Nice}(\text{Anna}), \text{Person}(\text{Mary}), \text{Nice}(\text{Mary}) \}$

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Etype expansion
- **Tell**
- Ask – Reasoning problems
- Key notions

Tell – Model building

Intuition (Model building). As from LoE and LoD, we construct a model by defining an assertional LoDE theory. A LoDE theory is defined as (see above)

$$EB = \langle L, T, A \rangle$$

We construct EB as follows:

- we construct **L** (for **L**exicon) as a TBox of LoD concepts, that is, as a set of Genus-Differentia definitions;
- we construct **T** (for **T**eleontology) as a TBox of etype definitions, that is, as a set of LoD descriptions;
- we construct **A** (for **A**ssertions) as an assertional theory consisting of LoE assertions, where all the etypes used are defined either as LoD concepts of **L** or as LoD etype definitions of **T**.

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Etype expansion
- Tell
- **Ask – Reasoning problems**
- Key notions

LoDE reasoning

Theorem. Let A' be an ABox expansion of the ABox A based on the terminology T' obtained by unfolding the TBox T . Let $a' \in \text{exp}(a)$, with $a \in A$, $a' \in A'$, being two assertions. Then

$$T \cup A \models a \iff A' \models a' \iff a' \in A'$$

Observation. The theorem above is the main result which also summarizes how expansion works. It can be summarized as follows:

- T is a LoD teleontology
- A is an LoE EG
- T is unfolded in T' and then T' is exploited to expand A into A'
- A' is an extended EG where the original nodes and links of the EG corresponding to A have been extended with the conjuncts resulting from the expansion of A with respect to T'

Observation. The theorem above is a corollary of the main result of expansion, see above. 50



Etype expansion (observations)

Observation (Scope of an expansion): A LoDE EG may contain both primitive and defined etypes. Which ones to keep is application and query dependent.

Observation (Size of an expansion): The expansion of an assertion involving a primitive etype does not change anything. The expansion of defined terms increases the number of nodes and also of links of the EG.

Observation (Purpose of an expansion). For each assertion in the EG, the level of detail at which an EG is expressed can be increased by expansion or decreased by its inverse operation, thus allowing for the fine tuning of the EG to the user questions.

LoDE Reasoning problems (the same as LoE)

Instance checking, Checking whether an assertion is entailed by a Model, i.e. checking whether

$$M \models E(e)$$

$$M \models P(e_1, e_2)$$

with $M = I(T)$.

LoDE Reasoning problems (the same as LoE)

Instance retrieval Given an etype (or object/ data property), retrieve all the entities (or pairs entity, entity/data) which satisfy the etype (object/data property)

$$M \models E$$

$$M \models P$$

with $M = I(T)$

Correctness and completeness

Proposition (Language correctness and completeness). L_{LODE} is correct with respect to D_{LODE} . L_{LODE} is complete with respect to D_{LODE} if I_{LODE} is surjective.

Evidence (Language correctness and completeness). From the definition of L_{LODE} and I_{LODE} for any assertion there is a corresponding fact in the domain, not necessarily only one. We have completeness when the alphabet contains symbols for all the percepts in the domain.

Proposition (Theory correctness and completeness). A theory $T_a = \{a\}$ such that, if $a \in T_a$ then $I(a) \in M$ is correct with respect to M . T_a is complete only if it is maximal.

Evidence (Theory correctness and completeness). By construction.

How to use LoDE (observations)

Observation (How to use LoDE – representation). LoDE is the logical formalization of all those AI / KR applications where knowledge is represented by three elements:

- A KG (formalized as a LoE EG) which models the specific application (e.g., the EG of a car dealer). The KG is application dependent. A new KG must be defined for any different application.
- A set of background knowledge LoD descriptions (formalized as LoD etypes) which define the relations which are left implicit (e.g., that a station-wagon has more space in the back, a car has four wheels). LoD descriptions are usually large reused but they are usually reused inside the same domain (e.g., Health, transportation, climate, ...)
- Lexicons, i.e., sets of words and how their meanings are related (formalized as LoD concepts, including disjointness statements) which are used to to define the KG (e.g., that a car is a vehicle). Lexicons are for the most part (modulo domain languages, see lecture on KG theories) are for the most part reused across domains and applications).

How to use LoDE (observations)

Observation (How to use LoDE – reasoning steps). Once built, via the Tell operation the LoDE EB one can use it as follows:

- Precompile all the language unfolding, using also the disjointness information
- Precompile all the knowledge unfoldings.
- The unfoldings should be indexed by the key words the use
- Given a query (instance checking or retrieval) just expand the smallest possible sub-EG
- Perform Q/A by «standard» Q/A technology

Observation (How to use LoDE – LoP reasoning steps). The approach above can be expanded to full reasoning. See next lectures

LoDE – The Logic of Entity Bases

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Etype expansion
- Tell
- Ask – Reasoning problems
- **Key notions**

Key notions

- Knowledge Base, expert system
- Entity Base
- Populated teleontology
- Terminology unfolding
- Etype expansion
- LoDE Instance checking
- LoDE Instance Retrieval



LoDE - The logic of Entity Bases (EBs) (HP2T)

Sept 13, 2023