UNIVERSITY OF TRENTO - Italy

Dipartimento di Ingegneria e Scienza dell'Informazione

Know dive

LoE The Logic of Entities

(HP2T)

Oct 13,2023

LoE – The Logic of entities

- **Intuition**
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Ask / Tell
- Key notions

LoE – Why a logic of entities?

- The **Logic of Entities (LoE)** allows us to describe entities as a function of the main class to which they belong, and their properties and relations.
- LoE formalizes and has the same level of expressiveness and reasoning capabilities as relational DBs and KGs which have entities as nodes (entity graphs).
- It automates, in a provably correct way, the kind of reasoning which is implemented in DBs and KGs. In particular, it allows for checking the facts which hold of entities when interacting among them.

LoE – Highlights

- All entities are assumed to have a name which uniquely identifies them, which may be unknown (multiple names for the same entity being allowed).
- It is world logic, with a graph linguistic / analogic representation.
- It is conceptually similar in spirit to an earlier logic called the Assertion Box (ABOX) of Description Logics (DL). The move is from DBs to KGs.

LoE – which percepts?

The **LoE world model** represents the following **Entity Graph** (**EG**) elements:

- An **Entity** is anything to which we give a name;
- An **Entity type** (**etype**) is the main class to which an entity belongs;
- A **Data Value (value)** is anything whose name and characteristics are predefined;
- A **Datatype** is a class of values;
- A **Data Property**, also called **Attributive Property** or also **Attribute***, describes an property type;
- An **Object Property** describes a relation type.

** Term derived from Relational DB terminology. It preserves the same intuition.*

Data types

Definition (Datatype, value). A data type is an etype where data and object properties are defined a priori. A data type is defined by providing: (i) the names of its values plus the interpretation function; (ii) data properties and their values; (iii) object properties and their values; and (iv) the notion of equality between values

Example (Natural Number). We have the following

1. Values: numerals, written 1,2,3, …, denoting natural numbers, 1,2,3,…
2. Data properties: odd, even, prime, multiple
3. Object properties: <, >, plus, minus, times, division
4. Equality: =

Example (Data type). Number, real, string, identifier, date, time, coordinates, weight, data (this being the most general datatype).

Observation (Data type, data value). Data types must be used "as is". The use of data types is a well established practice in Computer Science. LoE allows much richer data types (e.g., date, geodatatypes, mood data types) as needed to model the world. Value identification is guaranteed via equality.

Etypes - examples

Location, i.e., an example etype which models spatial containment of entities.

Locations do not change their position with respect to their coordinate reference systems. That is, coordinates are **identifying data properties**. They are keyfor deciding whether two locations (i.e., two entities belonging to the etype Location) are actually the same location.

There are many etypes which are special cases (sub-etypes) of Location, for instance: *Mountain*, *City*, *Street*, *Home*, and many others.

Other important etypes are:

Entity, the most general etype, that which contains all elements of the EG. Its most relevant property is that it has a **unique identifier**, thus imposing that all entities must have it. The identifier is the identifying data property. In most cases it does not exist;

Event, whose most characterizing properties are its start and end times. Which identifying data properties?

Person, with properties such as name, birth date, parents, phone number; and many others. Which identifying data properties?

Data types vs. Entity types – example

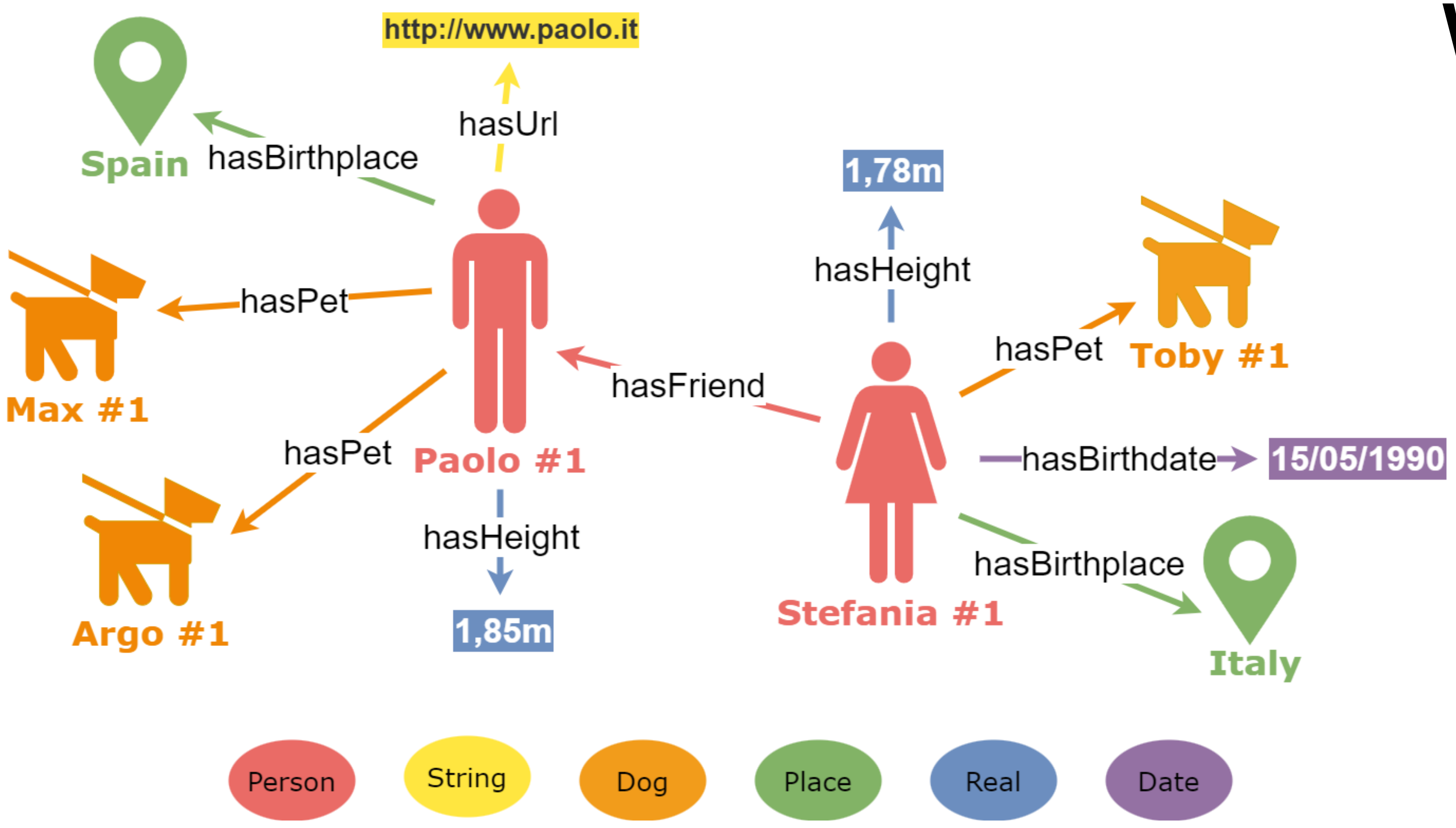
Example (Natural Number). We have the following

1. Values: numerals, written 1,2,3, …, denoting natural numbers, 1,2,3,…
2. Data properties: odd, even, prime, multiple
3. Object properties: plus, minus, times, division
4. Equality: =

Example (Person). We have the following *user-defined* example:

1. Entities: Sofia, Anonymus, Marco, Fausto, Pietro … *denoting* Sofia#3, #1211, Marco#7,Fausto#1, …
2. Data properties: name, height, age, gender, coordinates, …
3. Object properties: near, friendOf, hasFriend, talksTo, …
4. Equality: FaceRecognition, SocialSecurityNumber, …

An example of LoE EG



Which percepts?

Which facts?

Well-formedness conditions

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.

LoE – The Logic of entities

- Intuition
- **Definition**
- Domain
- Language
- Interpretation function
- Entailment
- Ask / Tell
- Key notions

LoE – The Logic of entities

Definition (LoE)

$$\text{LoE} = \langle \text{EG}, \models_{\text{LoE}} \rangle$$

with

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

When no confusion arises, we drop the subscripts.

LoE – The Logic of entities

- Intuition
- Definition
- **Domain**
- Language
- Interpretation function
- Entailment
- Ask / Tell
- Key notions

Domain

Definition (Domain)

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

where:

- U = {u} is called the **universe (of interpretation)** of D.
- {u} is a set of **units** $u_1, \ldots, u_n$, for some n
- {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 (EG, Binary relations). To comply to the graph notation we restrict ourselves to binary relations.

Universe of interpretation

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.

Universe of interpretation – entities and values

Observation (Entities). The notion of entity defined here captures the notion of entity informally introduced in the model theory. An entity is a thing that we perceive.

Example (Etype). Entity, person, artifact, vehicle, car, bus, woman, man, …, professor, father, student, parent. Some are mutually djsjoint, some are not, some are subsets of others.

Observation (Values of dtypes). Values are not real world entities. They do not exist in the world as we perceive it. They are **abstract entities,** so-called **mind constructs,** that are defined to capture certain characteristics of entities. They allow to **compare** certain characteristics of entities, thanks to their predefined definition of equality and properties. Some are mutually djsjoint, some are not, some are subsets of others.

Example 1 (values of dtypes).

- Numbers allow to compare quantities (1, 2, 3, 4, … are all entities, called numerals)
- Unities of measure (e.g., weight, height) when associated to numbers, allow to compare, specific aspects of entities
- Qualitative dtypes with qualitative values (hair color, beauty, empathy, clarity)
- Qualitative dtypes with a quantitative definition (e.g., color, mood)

Example 2 (Dtype). Real, long, Integer are subsets of number. They are all disjoint. Identifier is a subset of string. They are all subsets of the most general datatype, that is, data

Classes

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 (etype, dtype). Not all subsets of E (of V) are etypes (dtypes). To become so, they must be identified as such, i.e., they must be given a name and defined to be so.

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

Facts

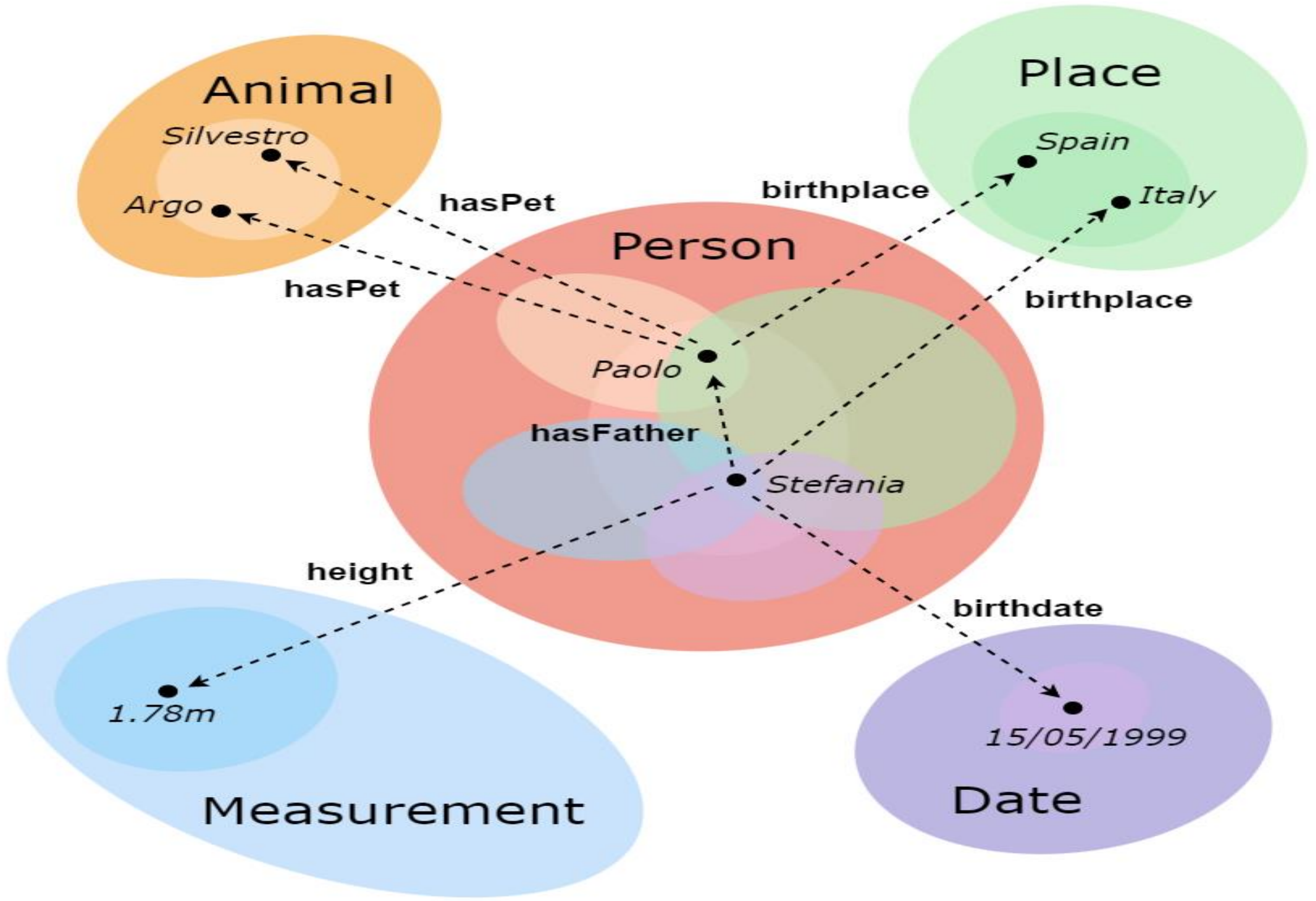
Observation. LOE allows for the following facts:

- 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
- $(e_1, e_2> \in O_i$, that is, the object property O_i holds between the two entities e_1, e_2
- $<e, v> \in O$, that is, the attribute A of entity e has value v

An example of EG – Venn diagram



An example of domain of EG (continued)

We have the following example EG

E = {#1, #2, #3, #4, #5, #6, #7, #8, #9, 1,85m, 1,78m, 15/05/1990, http://www.paolo.it}

E_T = {Entity, Person}

D_T = {Data, Real, Boolean, String, Date, Length}

{R} = {hF, hD, hH, hB, hL, hU}

from which we construct the following facts in the domain

D = {#1 ∊ Entity, 1,85∊ Real, 1,85 ∊ Data, hF(#1, #1), …}

LoE – The Logic of entities

- Intuition
- Definition
- Domain
- **Language**
- Interpretation function
- Entailment
- Ask / Tell
- Key notions

Language

Definition (Assertional language)

$$L_a = < A_a, FR_a > = \{a\}$$

where L_a is an **assertional language**, A_a is an **alphabet** of assertions and FR_a is a **set of formation rules.**

Alphabet

Definition (Alphabet A**)***

$$A_a = < E, \{T\}, \{P\} >$$

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 distinsguish them from percepts

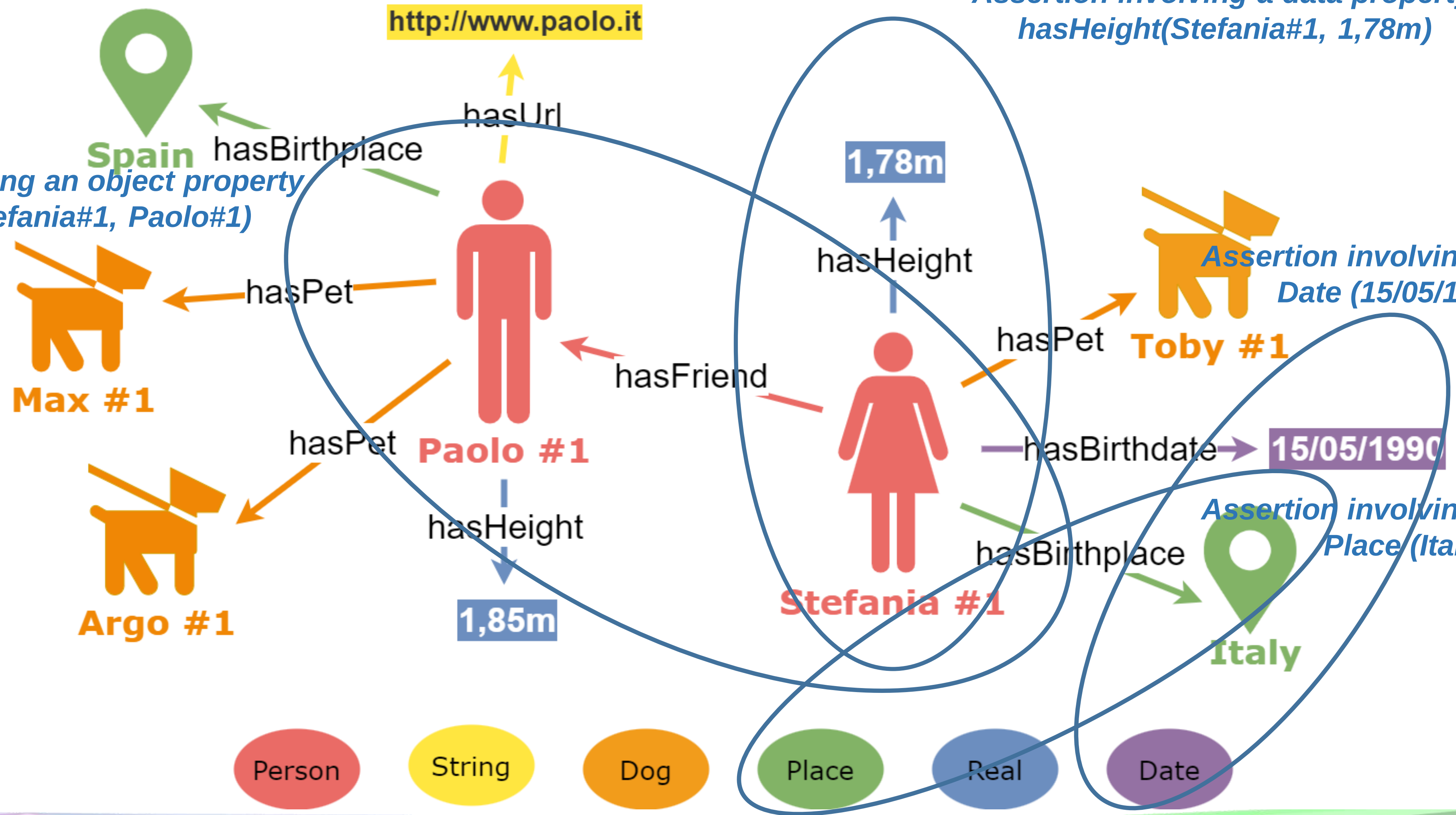
Formation Rules – BNF

```
<assertion>      ::= <etype>(<nameEntity>)                         |
                     <dtype>(<value>)                              |
                     <objProp>(<nameEntity>, <nameEntity>)  |
                     <dataProp>(<nameEntity>, <value>)
```

<etype>	::= E_1 | ...| E_n
<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 (BNF). This BNF generates assertions which are triples. It can be used to generate entity graph assertions as well as as a set of textual assertions.

An example of EG – assertions



Assertion involving a data property hasHeight(Stefania#1, 1,78m)

Assertion involving an object property hasFriend(Stefania#1, Paolo#1)

Assertion involving a dtype Date (15/05/1990)

Assertion involving an etype Place (Italy)

Assertions – as from BNF (same as facts)

Observation. LOE allows for the following assertions:

- Every etype/ 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

LoE – Theory (example –as from above)

Alphabet

- **Set of entities** = {Paolo, Sofia, Stefania, Argo, Max, Toby, Balto, Spain, Italy, Balto Spain, Italy}
- **Set of values** = {1,85m, 1,78m, 15/05/1990, http://www.paolo.it}
- **Set of etypes** = {Person, Dog, Place}
- **Set of dtypes**= {Measurement, Date}
- **Set of object properties** = {hasFriend, hasDog, hasBirthPlace}
- **Set of data properties** = {hasBirthdate, hasUrl, hasHeight}

Assertions = {Person(Paolo), hasBirthplace(Paolo, Spain)
hasHeight(Paolo, 1,85m), hasDog(Paolo, Argo),
hasUrl(Paolo, http://www.paolo.it), hasDog(Paolo, Max)}

LoE expressiveness – observation

Observation 1 (LOE expressivity). The language of LoE is mapped one-to-one to the structure of domain interpretation.

Observation 2 (LOE expressivity). In LoE, given the low expressiveness of the language, the properties of etypes and dtypes can be exploited to make assertions but they cannot be reasoned about (similar to Relational DBs).

Example 1 (weak expressivity: professor and person). The fact that Fausto is a person cannot be derived from the fact that he is a professor. There is no way to state the fact subsumption that all professors are persons (called a subsumption).

Example 2 (weak expressivity: professor and person). One could think of solving the problem above by adding to each professor the fact that (s)he is also a person. This would still not allow to assert the general statement, instance-independent, that “a professor is also a person”.

LoE – The Logic of entities

- Intuition
- Definition
- Domain
- Language
- **Interpretation function**
- Entailment
- Ask / Tell
- Key notions

Interpretation function

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

$$I : L \rightarrow D$$

where I is defined as

$$I =< I_E, I_T, I_P >$$

where:

- $I_E = <I_e, I_v>$, the Entity intepretation function, is the pair of the entity and the value interpretaton function;
- $I_T = <I_E, I_D>$, the Type intepretation function, is the pair of the etype and the dtypes interpretaton function;
- $I_P = <I_O, I_A>$, the Property intepretation function, is the pair of the object property and the attribute interpretaton function

with:

$$I_e : \{e\} \rightarrow \{\mathrm{e}\} \qquad I_v : \{v\} \rightarrow \{\mathrm{v}\}$$

$$I_E : \{E_i\} \rightarrow \{\mathrm{E_i}\} \qquad I_D : \{D_i\} \rightarrow \{\mathrm{D_i}\}$$

$$I_O : \{O_i\} \rightarrow \{\mathrm{O_i}\} \qquad I_A : \{A_i\} \rightarrow \{\mathrm{A_i}\}$$

Interpretation function (continued)

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

$$I(E_i\ (e)) = I_E(E_i)\ (I_e(e)) = e \in E_i$$
$$I(D_i\ (v)) = I_D(D_i)\ (I_v(v)) = v \in D_i$$
$$I(O_i\ (e_1,e_2)) = I_O(O_i)\ (I_e(e_1), I_e(e_2)) = \langle e_1, e_2 \rangle \in O_i$$
$$I(A_i\ (e,v)) = I_A(A_i)\ (I_e(e),\ I_v(v)) = \langle e, v \rangle \in A_i$$

Observation. Notice how the interpretation of assertions first applies the interpretation function (i.e., I) 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).

Observation. The definition I =< I_E, I_T, I_P > is the **intensional** view of the interpretation function. The defition of how it applies to assertions (this page) is its **extensional** view.

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

Interpretation function – Example (continued)

We have the following

I(Paolo)	= #1
I(hasFriend(Paolo, Sofia))	= hf(#2, #1)
I(Person(Stefania))	= P(#3)
I(hasDog(Stefania, Toby))	= hD(#3, #6)
I(hasDog(Paolo, Argo))	= hD(#1, #4)
I(hasDog(Paolo, Max))	= hD(#1, #5)

Correctness and completeness

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

Evidence (Language correctness and completeness). From the definition of L_{LOE} and I_{LOE} for any assertion there is a corresponding fact in the domain, not necessarily only one assertion per fact. 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. For any correct assertion, the compositionality of the interpretation function guarantees that the correct percept is computed.

Interpretation function - compositionality

Observation (Semantic compositionality). By semantic compositionality we mean the fact that the interpretation function interprets the elements of the alphabet one by one and builds the resulting final fact by composing the resulting percepts by mimicking how the formation rules compose.

In this way, either an input assertion is not well-formed, in which case it makes no sense to ask what it means, or the resulting fact will be the right interpretation of the input assertion, by construction.

This is how interpretation functions are built, for all logical languages sentences, not only for assertional languages and assertions.

It guarantees correctness, and it is also easily understandable by humans, as this is also how the semantics of natural languages are built.

LoE – The Logic of entities

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- **Entailment**
- Ask / Tell
- Key notions

Entailment relation

Definition (LoE entailment). Let $T_a \subseteq L_a$ be a theory and M ∈ D a model of W. Then

$$M \models T_a \text{ if and only if } I_a(a) \in M, \text{ for all } a \in T_a$$

$$M \models \{a\} \text{ if and only if } I_a(a) \in M$$

Observation (LOE entailment). The LoE entailment is the basic world entailment (see lecture on world model entailment).

LoE – The Logic of entities

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- **Ask / Tell**
- Key notions

Tell – model assertion

Definition (LoE TellT). We have the following

$$\text{TellT}(R, T) = R'$$

with

$$T_a' = T_a \cup T,$$
$$M' = M \cup \{I_a(a)\}, \text{ for all } a \in T$$

Observation 1 (LoE Tell). LoE allows for **theory extension** but not necessarily for **alphabet extension**.

Ask – Instance checking

Definition (LoE Instance checking) LoE instance checking checks whether an assertion is entailed by a model. We have the following four possible cases

M ?= $E(e)$	returns yes	if $M \models E(e)$,	no otherwise
M ?= $D(v)$	returns yes	if $M \models D(v)$,	no otherwise
M ?= $O(e_1, e_2)$	returns yes	if $M \models O(e_1, e_2)$,	no otherwise
M ?= $A(e, v)$	returns yes	if $M \models A(e, v)$,	no otherwise

Observation 1 (LoE Instance checking). It is a specific form of AskC.

Ask – Instance checking – close / open world

Observation (Instance checking). Consider the following two variations of instance checking

M ?= E(e) returns **"yes"** if M |= E(e), it returns **"no"** otherwise

M ?= E(e) returns "**E(e) is true**" if M |= E(e), it returns **"E(e) is false"** otherwise

Open World Assumption (OWA): The first version, the LoE instance checking, returns «I don't know» when entailment does not hold.
Closed World Assumption /CWA): The second version, returns that «E(e) is false» when entailment does not hold.

The second transforms **partial knowledge** (a fact not being in the model) into **negative knowledge**. What is not known is assumed to be false.

OWA is mainly applied in the Web, CWA is mainly applied corporate DBs.

Ask – Instance retrieval

Definition (LoE Instance retrieval). Given an etype (or object/ data property), retrieve all the entities (or pair entity, entity/data) which satisfy the etype (or object property/attribute) in input

M ?= E

M ?= D

M ?= O

M ?= A

Observation (Instance retrieval). It generates all the possible assertions using the entities and values of the alphabet and applies model checking to the resulting assertion. It returns the assertions for which model checking returns "yes".

Observation (Instance retrieval). A multiple call to Instance checking. Similar to Q/A in relational DBs.

LoE – The Logic of entities

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- Ask / Tell
- **Key notions**

Key notions

- LoE as a logic of entities
- Entity graphs (EGs)
- Entities and values
- Etypes and dtypes
- Object properties
- Data properties (alias attributes)
- LoE alphabet
- LoE formation rules
- LoE assertions
- LoE expressiveness
- Interpretation function
- Intensional definition of an interpretation function
- Extensional definition of an interpretation function
- Entailment relation
- LoE TellT
- LoE Instance checking
- OWA
- CWA
- LoE Instance retrieval

LoE
The Logic of Entities

(HP2T)

Oct 13,2023