LoI

The Logic on Interaction

(HP2T)

LoI – The Logic of Interaction

- **Intuition**
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

LoI – Why a logic of Interaction?

- Historically the first logic which was "invented" (by Aristotele, the logic of syllogisms);
- The mainstream name for LoI is **FOL** (for **First Order Logic**);
- It is the logic which is closest to natural language. If LoP is the logic of reasoning (formalizing and exploiting, we can say, "the language of thought"), then LoI is the logic of Interaction (formalizing and exploiting, we can say, "the language of interaction");
- It is the logic used in mainstream philosophical logic and mathematical logic;
- In philosophical logic, LoI is the logic used to formalize the liar paradox;
- In mathematical logic, LoI is the logic used to formalize mathematical reasoning which lead to Goedel's completeness theorem and Goedel's very famous incompleteness theorem;
- In CS, LoI was used extensively in the early days, later given up for more computational-friendly logical formalisms (e.g., LoP, Temporal Logics);
- In AI, LoI is the key logic used (in its nonmonotonic version) for the modeling of commonsense knowledge and reasoning, because of its "proximity" to natural language.

LoI – Highlights

- It maintains all the LoP propositional connectives;
- It extends enormously the expressiveness of LoP atomic propositions which, in LoI, are allowed to represent n-ary properties (called predicates here) and also functions, thus allowing for the explicit representation of LoDE assertions, and beyond;
- It allows for the representation of free variables meaning by this some generic symbols which, differently from constants, representing generic individuals not constrained by specific properties(e.g., what you would mention using words such that "somebody", "something", "sometimes", "somewhere", "anybody", …);
- It allows for full universal and existential quantification, very much in the same way as we use them in natural languages;
- Because of quantification, it allows to represent and **reason finitely about infinite domains**, this being the reason why it allows the statement and reasoning about paradoxes and the incompleteness of (human?) reasoning.

Expressivity in LoP – example

Example (Properties of entities)

Problem: How to express in LoP the following statements?

- Mary is a person
- John is a person
- Mary is mortal
- John and Mary are friends

Solution: As primitive propositions!

- p='Mary-is-a-person' (same as 'A')
- p='John-is-a-person' (same as 'B')
- p='Mary-is-mortal' (same as 'C')
- p='John-mary-are-friends' (same as 'D')

Difficulty: Atomic propositions are black boxes which do not capture the internal structure of those facts which constitute their intended model. You cannot develop the mapping incrementally.

Expressivity of LoI – example

Example (Properties of entities)

Problem: How to express in LoI the following statements?

- Mary is a person
- John is a person
- Mary is mortal
- friend(Mary, John)

Solution: As atomic propositions which are not primitive! See also LoE.

- Person(Mary)
- Person(John)
- Mortal(Mary)
- friend(Mary, John)

Fact: All problems of LoP are solved. Atomic propositions are transparent with respect to the structure of the facts of their intended model. By LoI reasoning we can infer that Mary is a person and mortal and that both John and Mary are mortal. LoI allows for a direct encoding of LoDE assertions into LoI atomic formulas. Note that LoE is a sub-language of LoI.

Expressivity of LoP – example

Example 2 (General statements about entities)

Problem: How to express in LoP the following statements?

- All people are mortal
- There is (at least) a person who is a spy

Solution: One proposition for each entity!

- Mary-is-mortal ∧ John-is-mortal ∧ Chris-is-mortal ∧ . . . (same as $A \wedge B \wedge C \wedge \ldots$)
- Mary-is-a-spy ∨ John-is-a-spy ∨ Chris-is-a-spy ∨ . . . (same as $D \vee E \vee F \vee \ldots$)

Difficulty: lots of them

1. A formula which is as long as there are entities
2. To write this formula we need to know how many entities there are
3. General statements are more abstract than simple enumerations
4. General statements work also with infinitely many entities
5. General statements allow to reason finitely about infinite sets.

Expressivity of LoI – example

Example (General statements about entities)

Problem: How to express in LoI the following statements?

- All people are mortal
- There is (at least) a person who is a spy

Solution: One proposition for each entity! Also with infinite domains.

- Forall people . Mortal(people)
- Exists person . Spy(person)

Fact: All problems of LoP are solved. When you say "Forall" or "Exists" you are completely abstracted from the specific entities and names of entities. LoI allows for a direct encoding of universal or existential LoD statements, and more. LoE is a subset language of LoI. The LoD language can be rewritten in the LoI language with a one-to-one mapping.

Expressivity of LoP - example

Example (Functional dependencies)

Problem: How to express in PL the following statement?

- the father of Luca is Italian

Solution: One proposition for each possible father entity!

- mario-is-father-of-luca ⊃ mario-is-italian
- michele-is-father-of-Luca ⊃ michele-is-italian
- …

Difficulty: lots of them

1. We need as many formulas as there are entities
2. We need need to know how many entities there are
3. Functional statements are more abstract than simple enumerations
4. Functional statements are independent of the names of entities (nothing changes if an entity changes name)
5. Functional statements work also with infinitely many entities.

Expressivity of LoI – example

Example (Functional dependencies)

Problem: How to express in FOL the following statement?

- the father of Luca is Italian

Solution: Exploit the generative power of functions

- Italian(fatherOf(Luca))

Fact: All the problems of LoP are solved. We can talk about specific entities without knowing their names. We can nest function symbols as many time as we want to generate as many entities as we need (up to infinity).

Fact: The information above is not representable in LoDE.

Expressivity of LoP – example

Example (Infinite domains of interpretation)

Problem: How to express in PL the following statement?

- Every Natural number is either even or odd

Solution: An infinitely long formula

- $(odd1 \vee even1) \vee (odd2 \vee even2) \vee (odd3 \vee even3) \vee \ldots$

Difficulty: just impossible to do in practice.

Expressivity of LoI – example

Example (Infinite domains of interpretation)

Problem: How to express in FOL the following statement?

- Every Natural number is either even or odd

Solution: An finitely long formula representing a statement about an infinite set

- forall number . (odd(number) or even(number))

Fact: The problem is solved. First order logic allows to make finitely long statements about infinite quantities and to reason finitely about them.

Inference in LoP – example

Example (Inference in Lop)

Problem: Consider a LoI theory where we know the following facts:

- Person(Mary)
- forall person.Mortal(person)

How to derive the fact that Mary is mortal?

Solution: You need to expand (see above) the universal quantification and you have to rewrite the first statement to have exactly the same proposition in both cases, e.g., Mortal-Mary.

Difficulty: The situation does not scale to more complex statements. Consider for instance having, instead of the second statement, the following fact:

forall entity. Person(entity) inplies Mortal(entity)

How do you correlate entity and person in the quantification? You need reasoning at translation time. That is, you need the rules of LoI while performing translation.

Inference in LoI – example

Example (Inference in LoI)

Problem: Consider a LoI theory where we know the following facts:

- Person(Mary)
- forall person.Mortal(person)

How to derive the fact that Mary is mortal?

Solution: LoI allows us to deduce the proposition Mortal(Mary) by substituting Mary to person (because of the universal quantifier) in the second fact

Fact: Mary is a person, but all person are mortal. Therefore also Mary is mortal.

Fact: The above reasoning would work also in the case of an infinite domain (e.g., natural numbers).

LoI – The Logic of Interaction

- Intuition
- **Definition**
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

LoI – Definition

We formally define LoI as follows

$$\textbf{LoI} = \langle\, L_{LoI}\,,\ \models_{LoI}\,\rangle$$

Observation (L_{LoI}, L_{LoP}). The language of LoI can be seen as an extension the language of LoP where atomic propositions are extended to represent arbitrarily complex formulas, allowing in particular for universally or existentially quantified formulas.

LoI – The Logic of Interaction

- Intuition
- Definition
- **Domain**
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Key notions

Domain of interpretation – definition (the same as in models)

Definition (Domain of interpretation) A **Domain (of interpretation)** D is defined as

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 elements, for some m, with $C_i \subseteq U$
- {R} is a set of n-ary **relations** $R_1^n, \ldots, R_p^n$ among elements, for some p, with $R_i^n \subseteq U \times \ldots \times U$
- $u_1, \ldots, u_n, C_1, \ldots, C_m, R_1^n, \ldots R_p^n$ are **percepts.**

Observation (LoI domain of interpretation) LoI allows for the most expressive domain.

Domain of interpretation – intuition (the same as in models)

Observation (Diversity of percepts). From a set theoretic point of view we have three different types of percepts:

- (i) units: u
- (ii) classes: C
- (iii) relations: R^n

where:

- Units depict entities
- Classes depict sets on entities
- Relations depict properties of sets of entities
- Relations depict relations of sets of entities

Types of facts (the same as in models)

Definition (Fact). A **fact f** has one of the following five forms

- *Unit memberOf Class:* $u_i \in C_j$,
- *Tuple of Units memberOf relation:* $< u_1, \ldots, u_n > \in R^n$,
- *Class subsetOf Class:* $C_i \subseteq C_j$,
- *Relation subsetOf relation:* $R_i^n \subseteq R_j^n$
- *Relation subsetOf tuple of classes and viceversa:*
  - $R^n \subseteq C_1 \times \ldots \times C_n$
  - $C_1 \times \ldots \times C_n \subseteq R^n$

with: $u_i \in U$, $C_i \subseteq U$, $R^n \subseteq U \times \ldots \times U$.

Facts and domains (the same as in models)

Observation 1 (percept, domain, fact). As defined before we have that a domain **D** is defined as a set of percepts, that is **D = {p}**. From the definition of fact we can also see **D** as the set of facts, that is

D = {f}

where a fact **f** ∈ **{f}** is built from percepts by applying any of the equations used in the construction of fact.

Observation 2 (percept, domain, possible fact). The construction of a **domain** depends on two modeling choices:

- The selection of the set of **percepts** {p}
- The selection of the set of **possible facts** {f}. Domains are usually assumed to contain all **possible facts** which can be composed from a selected set of percepts {p}.

where by "possible fact" we mean a fact (relation among percepts) which may eventually be what is the case

Facts, models and domains (the same as in models)

Proposition (Domain and facts). A **Domain** D is a set of facts {f}.

$$D = \{f\}$$

Proposition (Model). Given a domain D, a model M in D is a subset of D.

$$M = \{f\} \subseteq D$$

Observation (Domain, model). A domain is the set of all facts that we are willing to consider. A model is just the subset of facts that we define when depicting what is the case in the current situation.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- **Language**
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Language

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

- **Step 1 – The language of terms**. Starting from an alphabet, consisting of constants and variables, the language of terms allows to build terms, by using function symbols which allow to build simple descriptions (constants) as well as complex descriptions denoting the units in the universe of interpretation U.
- **Step 2 – The language of atomic propositions.** The terms built in step one are used as arguments to predicate symbols. The result is the construction of atomic propositions which are true or false, similarly to what happens in LoP, with a critical step in the case of free variables.
- **Step 3 – The language of (complex) propositions.** Atomic propositions are then composed into (complex) propositions by using the LoP connectives plus two new connectives, representing existential and universal quantification. These connectives allow to quantify over the terms and variables which are arguments of predicate symbols, as from step 2.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - **The language of atomic propositions**
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

The language of terms

Definition (Language of terms, T)

$$T = <A_T, FR_T> = \{t\}$$

where:

- $T = \{p_T\}$ is a **language** of terms
- A_T is an **alphabet** for term generation
- FR_T is a **set of formation rules**
- $\{t\}$ is the set of terms 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).

The alphabet of terms

Definition (Alphabet A_T of term symbols)*

$$A_T = < E, X, F >$$

where:

- $E = \{c\} = \{e\} \cup \{v\}$ is a set of (names of) **entities** e and of **values** v, also called **constants**;
- $X = \{x\} =$ is a set of term variables;
- $F = \{f\}$ is a set of n-ary **function symbols.**

Terminology (Constant, entity, value). For now on, following the terminology used in LoI, when no confusion arises, we talk generically of constants meaning both entities and values.

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

The alphabet of terms – observations

Observation (Alphabet of terms). The symbols used for defining terms are as follows:

- a set $c_1, c_2, \ldots$ of **constant symbols**, also called **constants**
- a set $x_1, x_2, \ldots$ of **variable symbols**, also called **variables**
- a set $f_1, f_2, \ldots$ of **functional symbols**, also called **functions**, each associated with an arity (i.e., the number of input arguments)

Example (Arithmetics). Constants of arithmetics are the natural numbers 0, 1, 2, 3, ... Examples of function symbols are: *+, -, succ.* No constraints on the name of variables.

Example (Real world representation). Examples of constants (entities) used in knowledge graphs are Aldo, Fausto, Trento but also the natural numbers (because of datatypes). Examples of function symbols are: *friendOf, MotherOf, heightOf.* No constraints on the name of variables.

Example (KG real world representation). Examples of constants (entities) used in knowledge graphs are Aldo, Fausto, Trento but also the natural numbers (because of datatypes). Functions and variables are not allowed in KGs.

The Alphabet of arithmetics – example

symbols	type	arity	intuitive interpretation
0	constant	0	the smallest natural number
succ(arg1)	function	1	a function which returns the successor of its input
+(arg1, arg2)	function	2	a function which returns the sum of the inputs
*(arg1, arg2)	Function	2	a function which returns the product of the inputs
x	variable	0	A variable which can be substituted with any term

The alphabet of a real world description – example

Constant	Variable	Function (arity)
Aldo	anyone	mark(3)
Bruno	someone	friendOf(1)
Carlo	a person	bestFriend(1)
MathLogic	the person	motherOf (1)
DataBase	x	
0, 1, . . . , 10	y	

The alphabet of a KG real world description – example

Constant	**~~Variable~~**	**Function (arity)**
Aldo	~~anyone~~	~~mark(3)~~
Bruno	~~someone~~	friendOf(1)
Carlo	~~a person~~	bestFriend(1)
MathLogic	~~the person~~	motherOf (1)
DataBase	~~x~~	
0, 1, . . . , 10	~~y~~	

Term formation rules – BNF

< term > ::= <function> (< term >, ..., < term >)

< term> ::= <constant>

< term> ::= <variable>

<function> ::= f_1 | ...| f_n

<constant> ::= c_1 | ...| c_n

<variable> ::= x_1 | ...| x_n

Observation (BNF). The number of terms must be the same as the arity *n* of the function. The limit case is arity n=1, as constants and variables are functions of arity 0.

Observation (BNF). This BNF does allow the iterative application of the formation rules on terms allowing therefore for the generation of terms of any depth.

The language of atomic propositions

Definition (Language of atomic propositions, L_{PT})

$$L_{PT} = \langle A_{PT}, FR_{PT} \rangle = \{p_T\}$$

where:

- $L_{PT} = \{p_T\}$ is a **language** of atomic propositions
- A_{PT} is an **alphabet** for atomic proposition generation
- FR_{PT} is a **set of formation rules**
- $\{p_T\}$ is the set of atomic propositions p_T obtained by the exhaustive application of FR_{PT} to A_{PT} (the transitive closure $FR_{PT}(A_{PT})$ of FR_{PT} applied to A_{PT}).

The alphabet of atomic propositions

Definition (Alphabet A_a)*

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

where:

- T consists of all the terms $t \in T$;
- $\{T\}$ is a set of unary predicates
- $\{P\} = \{O_i\} \cup \{A_i\}$ is a set of n-ary **predicates**, where O_i is an **object property**, also called a **role**, and A_i is an **attribute**.

Observation (unary predicates). etypes and **dtypes**, as from LoDE, are a subset of unary predicates.

Observation (n-ary predicates). Object properties and **attributes (roles)**, as from LoDE, are a subset of n-ary predicates.

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

The alphabet of atomic propositions – observations

Observation (Alphabet of atomic propositions). The symbols used for defining atomic propositions are as follows:

- a set $t_1, t_2, \ldots$ of **terms**, containing, as a particular case, **constants** $c_1, c_2, \ldots$ and **variables**;
- a set $p_1, p_2, \ldots$ of **unary and n-ary predicate symbols**, also called **predicates**, each associated with an arity (i.e., the number of input arguments)

Example (Arithmetics). Terms of arithmetics are the natural numbers 0, 1, 2, 3, +(x,2), plus complex terms -(300,+(x,2)), succ(z), ... Examples of predicates are: **=**, >, <, *odd, even.* No constraints on the name of variables.

Example (Real world representation). Examples of terms are Aldo, Fausto, Trento, *friendOf(Aldo,fausto,Trento), talksTo(fausto,Aldo,Trento, lunch).*

Example (KG real world representation). The only terms allowed by KGs are entities. Examples of predicates are: *friendOf, hasFriend, motherOf, hasMother.*

The Alphabet of arithmetic atomic propositions – example

symbols	type	arity	intuitive interpretation
0	constant	0	the smallest natural number
succ(arg1)	function	1	a function which returns the successor of its input
+(arg1, arg2)	function	2	a function which returns the sum of the inputs
*(arg1, arg2)	function	2	a function which returns the product of the inputs
<(arg1, arg2)	predicate	2	a relation between numbers
≤ (arg1, arg2)	predicate	2	a relation between numbers

The alphabet of a real world description – example

Constant	**Variable**	**Function(arity)**	**Predicate(arity)**
Aldo	anyone	mark(3)	attend(2)
Bruno	someone	friendOf(1)	friendsOf(5)
Carlo	person	bestFriend(1)	student(1)
MathLogic	x	motherOf (1)	course(1)
0, 1, . . . , 10	y		LessThan(2)

Observation (Alphabet of a real world description). Any informal notion can be formalized in many different ways. For instance, friendship can be formalized as a unary function, as a binary predicate or as a n-ary function or predicate, taking into account extra information, for instance: age of friends, location, period of the year, obtaining for instance friendsOf(person, age, location, year, period).

The alphabet of a KG real world representation – example

Constant	~~**Variable**~~	~~**Function(arity)**~~	**Predicate(arity)**
Aldo	~~anyone~~	~~mark(2)~~	attend(2)
Bruno	~~someone~~	~~friendOf(1)~~	~~friendsOf(5)~~
Carlo	~~person~~	~~bestFriend(1)~~	student(1)
MathLogic	~~x~~	~~motherOf (1)~~	course(1)
DataBase	~~y~~		LessThan(2)
0, 1, . . . , 10			~~mark(3)~~
			friendOf(2)
			bestFriend(2)
			motherOf (2)

Observation (Alphabet of a KG real world description). The KG language limits what can be stated as part of the language. The choice of KGs is a choice between language expressivity and computational efficiency.

The alphabet of atomic propositions – observations

Observation (Constant). Constants represent the entities in the domain of interpretation, the objects to be reasoned about. Not all the entities need to have a constant representing them.

Observation (Variable). Any variable can be used to represent "dynamically" the entities in the domain. Variables can be instantiated to terms, in the form of constants, of other variables and also of complex terms.

Observation (Function). Functions take in input entities and generate "by computation" the description of a new single entity. The entities generated by functions may or may not be represented by a constant. The application of functions can be iterated for any number of times.

Observation (Constant, Function). Constants and variables are functions of arity zero, that is, functions which have themselves in input as well as in output.

Observation (Predicate). Predicates take in input description of entities and describe a specific aspect of a state of affairs, that is, whether a certain property holds among the input entities. Out of them it is possible to compute propositions and whether they are **True** of **False**.

Observation (Equality Predicate). Term equality is modeled using equality, that is the symbol "**=**".

Observation (Primitive proposition). Primitive propositions are predicates of arity zero, that is, predicates where the description of the input entities is encoded in the name. Primitive **LoI propositions** are **LoP propositions**.

Atomic proposition formation rules – BNF

< atomic proposition > ::= <predicate> (< term >, ... < term >)

Observation (BNF). The number of terms must be the same as the arity of the predicate. The limit case is arity 0, in which case the atomic proposition reduces to a LoP proposition.

Observation (BNF). This BNF does not allow the iterative application of the formation rules on propositions. There cannot be nesting of atomic propositions (from which, the name).

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
    - The language of atomic propositions
    - **The language of propositions**
- Interpretation function
    - Atomic closed formulas
    - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

The language of propositions

Definition (The language of propositions, L_{LoI}**)**

$$L_{LoI} = L_P = < L_{PT}, FR_P > = \{p\}$$

where:

- $L_{LoI} = L_P = \{p_T\}$ is a **language** of propositions
- $L_{PT} = \{p_T\}$, the **alphabet**, is a language of atomic propositions $p_T \in L_{PT}$
- FR_P is a **set of formation rules**
- $\{p\}$ is the set of propositions p obtained by the exhaustive application of FR_P to L_{PT} (the transitive closure $FR_P(L_{PT})$ of FR_P applied to L_{PT}).

Formation rules – logical symbols

Definition (Logical symbols). The set of formation rules FR_P is based on a set of so-called **logical symbols** defined as follows:

- **Propositional connectives** $\wedge, \vee, \supset, \neg, \equiv, \oplus$
- **Quantifiers** $\forall, \exists$
- An infinite set of **variable** symbols $x_1, x_2, \ldots$

Observation (Logical symbols). The LoI logical symbols extend the LoP logical symbols by allowing to express general statements via quantifiers.

Observation (Variables). Variables are "mute" place-holders needed to represent general statements about what is the case in the model. They are logical symbols which modify the structure of terms

Formation rules – BNF

$$
\begin{aligned}
< P > \ ::= \ & \text{<atomic proposition>} \mid \\
& \neg < P > \mid \\
& < P > \wedge < P > \mid \\
& < P > \vee < P > \mid \\
& < P > \supset < P > \mid \\
& < P > \equiv < P > \mid \\
& < P > \oplus < P > \mid \\
& \exists x. < P > \mid \\
& \forall x. < P >
\end{aligned}
$$

Formation rules – observations

Observation (LoI well-formed formulas). We call LoI **well-formed formulas** the formulas generated via FR_P. They have the following general form:

- if A and B are formulas then $\bot$, $A \land B$, $A \supset B$, $A \lor B$, $\neg A$, $A \equiv B$, $A \oplus B$, are formulas
- if A is a formula and x is a variable, then $\forall x.A$ and $\exists x.A$ are formulas

Notation (BNF). < atomic proposition > is a nonterminal. See the BNF of L_{PT} to see how to expand it to a terminal terms.

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

Well-formed-formulas – example

Aldo and Bruno are the same person	$Aldo = Bruno$
Carlo is a person and MathLogic is a course	$person(Carlo) \wedge course(MathLogic)$
Courses are attended only by students	$\forall x\, ((attend(x, y) \supset course(y)) \supset student(x))$
Every course is attended by somebody	$\forall x\, (course(x) \supset \exists y.attends(y, x))$
Every student attends something	$\forall x\, (student(x) \supset \exists y.attends(x, y))$
A student who attends all the courses	$\exists x\, (student(x) \wedge \forall y\, (course(y) \supset attend(x, y)))$
Aldo's best friend attends the same courses attended by Aldo	$\forall x\, (attend(Aldo, x) \supset attend(bestFriend(Aldo), x))$
Best-friend is symmetric	$\forall x\, (bestFriend(bestFriend(x)) = x)$
Aldo and his best friend have the same mark in MathLogic	$mark(bestFriend(Aldo), MathLogic) = mark(Aldo, Mathlogic)$

Free occurrence of variable

Definition (Free occurrence) A variable x is said **occur free in a formula** if one of the following facts holds

- an occurrence of x in a term t_k is **free** in $P(t_1, \dots, t_k, \dots, t_n)$
- any **free** occurrence of x in a formula ϕ or ψ is also free in $\phi \land \psi, \psi \lor \phi, \psi \supset \phi,$ and $\neg\phi$
- a free occurrence of x in a formula ϕ is **free** in $\forall y.\phi$ and $\exists y.\phi$ if y is distinct from x

A variable occurrence is **bound** if it is not **free**.

Definition (Ground/Closed formula) A formula ϕ is **ground** if it does not contain any variable. A formula is **closed** if it does not contain free occurrences of variables, **open** otherwise.

Observation (Free / bound occurrence of a variable). Intuitively, given a formula, a free occurrence of a variable x is an occurrence of x which is not bounded by a (universal or existential) quantifier.

Observation (Occurrence of a variable in an atomic formula). An occurrence of a variable is always free in an atomic variable. An atomic formula is either ground or open

Observation (ground formula). A ground formula is also closed, but not vice versa.

Free occurrence of variable – example

Example (Free occurrence of a variable)

- $Friends(Bob, y)$: y is free
- $\forall y.Friends(Bob, y)$: y is bound
- $Sum(x, 3) = 12$: x is free
- $\exists x.(Sum(x, 3) = 12)$: x is bound
- $\exists x.(Sum(x, y) = 12)$: x is bound, y is free
- $\exists x.(Sum(x, \boldsymbol{y}) = 12 \supset \exists y.(Diff(12, \boldsymbol{x}) = y$: The first occurrence of y is free, the second is bound; the first occurrence of x is bound, the second is free
- $P(\boldsymbol{x}) \supset \forall x.Q(x)$ the first occurrence of x i is free, the second is bound

Observation (free occurrence of a variable). Inside a formula, a variable may occur both free and bound, with multiple occurrences each.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- **Interpretation function**
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Interpretation function – observations

Observation (interpretation function). Interpretation functions apply (as it is aways the case) to atomic formulas. In the definition of the interpretation function we need to distinguish two cases.

- **Atomic closed formulas**, for instance
  - $f(a, b) = c$
  - P(c,d)
- **Atomic open formulas**, for instance
  - $f(a, x) = c$
  - $P(y, z)$

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - **Atomic closed formulas**
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Interpretation function

Definition (Interpretation function). The **LoI interpretation function I** is defined as

$$I : L \rightarrow D$$

with

$$I =< I_c, I_f, I_P >$$

where

- I_c, the constant interpretation function, is such that $I_c \in D$
- I_f, the function interpretation function, is such that $I_f(f): D^n \rightarrow D$, with f an n-ary function. That is, $I_f(f) \subseteq D^{n+1}$;
- I_P, the predicate interpretation function, is such that $I_P(P) \subseteq D^n$, with P an n-ary predicate;

with, by definition, $D^n = D \times \ldots \times D$.

Interpretation function – example

Example (Interpretation). Let the **alphabet** be defined as follows.

- *Constants*: Alice, Bob, Carol, Robert
- *Functions*: mother-of (with arity equal to 1)
- *Predicates*: friends (with arity equal to 2)

Interpretation function – example (continued)

Let the **domain of interpretation** be the Natural numbers.

$$D = \{1, 2, 3, 4, \ldots\}.$$

Let the **interpretation function** be as follows:

$$I(alice) = 1$$
$$I(bob) = 2$$
$$I(carol) = 3$$
$$I(robert) = 2$$

$$I(motherOf) = \begin{cases} MotherOf(1) = 3 \\ MotherOf(2) = 1 \\ MotherOf(3) = 4 \\ MotherOf(n) = n + 1 \text{ for } n > 3 \end{cases}$$

$$I(friends) = \begin{matrix} (1,2) & (2,1) & (3,4) \\ (4,3) & (4,2) & (2,4) \\ (4,1) & (1,4) & (4,4) \end{matrix}$$

Interpretation function – example (continued)

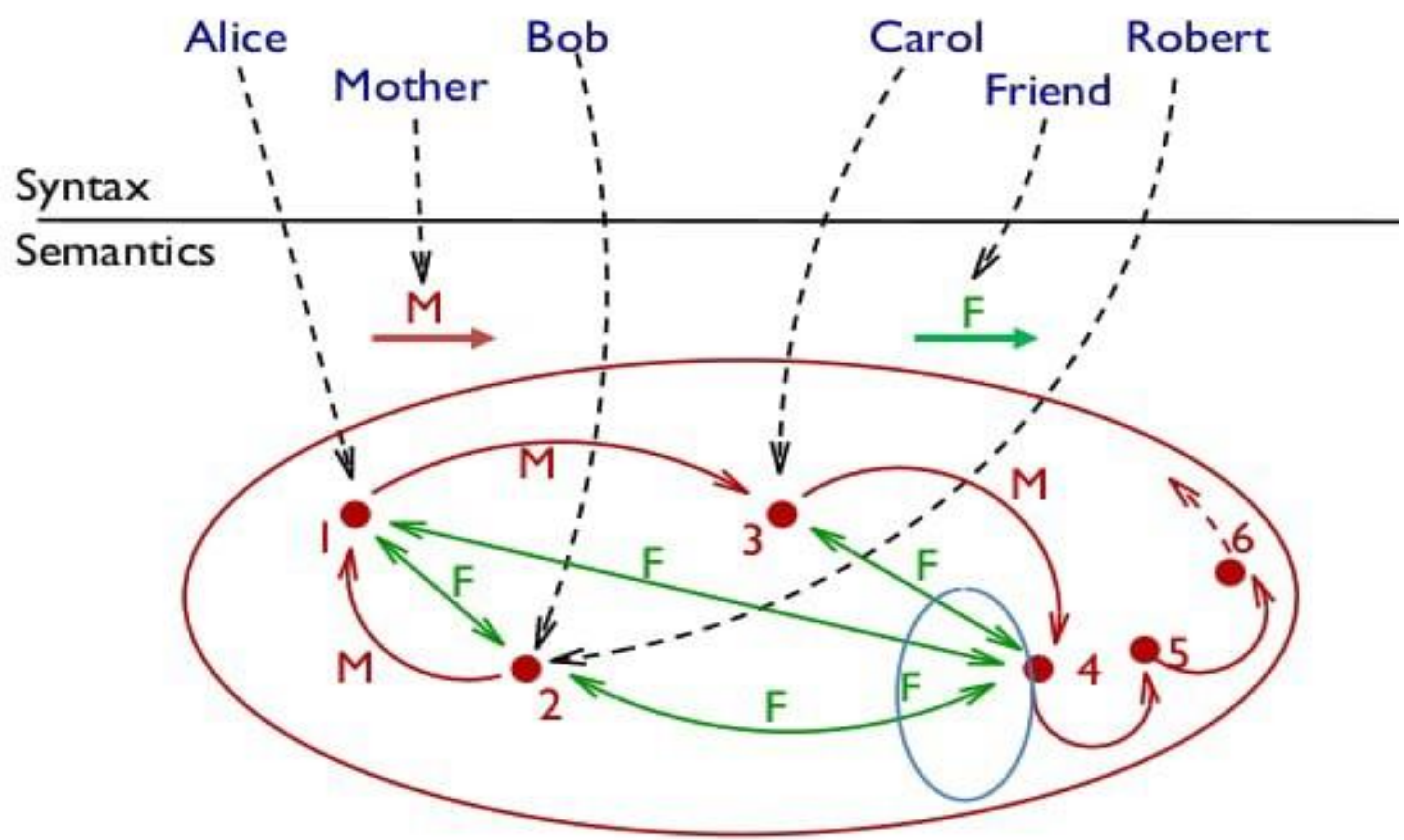

Figure: Language, Domain and Interpretation function

Interpretation of a ground formula – observations

Observation (Interpretation of a term). The interpretation of a term is an entity in the domain. We graphically represent entities in the domain using identifiers which are different for different entities.

Observation (Interpretation of a constant). We may have synonymity (e.g., Bob, Robert) but not polysemy.

Observation (Interpretation of a function symbol). Function symbols are used to generate complex term descriptions. n-ary function symbols (in the example above, the unary function "Mother") denote the space of all possible terms which can be constructed. N-ary function symbols are graphically represented as (n+1)-ary tuples.

Observation (Interpretation of a term). As from the BNF, functions allow to generate terms of any depth. A term is the name of an element of the domain. Therefore, with infinite domains, functions can be used to generate infinite constants. This is the main reason of the semi-decidability of LoI.

Observation (Interpretation of a predicate symbol). n-ary predicates generate atomic formulas. They denote the space of all possible facts (i.e., n-ary relations applied to primitive terms) which hold in the model. The fact generated by an n-ary predicate is graphically represented by a n-tuple.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - **Atomic open formulas**
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Interpretation of an atomic open formula – observations

Observation (Interpretation of a variable). A variable, as the name says, is allowed to change. A main reason of existence is in fact as an argument of quantifiers. As such, a variable **cannot** be interpreted as an element of the domain. At the same time, to have meaning, an atomic formula (and therefore all the variable occurring in it) **must** be assigned an interpretation in the domain.

Example (Interpretation of a variable). Think of the following natural language sentences

- Someone went there (but I do not know who he was)
- I saw something (but I could not recognize it)

The meaning of "someone" and "something" are unspecified but their meaning can be bound to a specific value when more information is available (as in "That something was a car ").

Observation (Variable assignment). Similarly to natural language, we interpret a free variable as something whose interpretation is unknown but that could be known by taking, as value, a possible element of the domain. Which element of the domain? No element can be discarded a priori. The context (i.e., the full formula/ theory) will allow to ascertain the good ones (e.g., because the guarantee satisfiability). This process is achieved by **variable assignments**.

Variable assignment

Definition (Variable Assignment). Let *L* be a first order language and D its domain of interpretation. Let $A \in L$ be a first order formula and let

$$Var\ (A) = \{x_1, ..., x_n\}$$

the set of variables occurring in *A*. An assignment *a* is a function from the set of variables to the domain of interpretation D.

$$a : Var\ (A) \rightarrow D.$$

We write $a[x/d]$ to mean the assignment that coincides with *a* on all the variables but *x*, which is associated to *d*, with $d \in$ D.

Variable assignment – example

Example (Variable assignment) Let *L* be a first order language with constants = {a,b,c}. Let D = {0,2,3} be the domain of interpretation of *L*. Let the interpretation function I be such that

- I(a)=0,
- I(b)=2,
- I(c)=3.

Let us consider the following example formulas:

- B(a): then there are no possible assignments;
- B(x): then the possible assignments are three: a1=[0], a2=[2], a3=[3];
- A(x1, x2): then the possible assignments are nine ($9 = 3^2$): a1=[0,0], a2=[0,2], a3=[0,3], a4=[2,0], a5=[2,2], a6=[2,3], a7=[3,0], a8=[3,2], a9=[3,3].

Interpretation of an atomic formula wrt an assignment

Definition (Interpretation of a term w.r.t an assignment). The interpretation of a term t w.r.t the assignment a, in symbols $I(t)[a]$ is recursively defined as follows:

- $I(c_i)[a] = I(c_i)$
- $I(x_i)[a] = a(x_i)$
- $I(f(t_1, \ldots, t_n))[a] = I(f)(I(t_1)[a], \ldots, I(t_n)[a])$

Definition (Interpretation of an atomic formula w.r.t an assignment). The interpretation of an atomic formula $P(t_1, \ldots t_n)$ w.r.t the assignment a, in symbols $I(P(t_1, \ldots, t_n))[a]$ is defined as follows:

$$I(P(t_1, \ldots, t_n))[a] = I(P)(I(t_1)[a], \ldots, I(t_n)[a]))$$

Interpretation – example (continued)

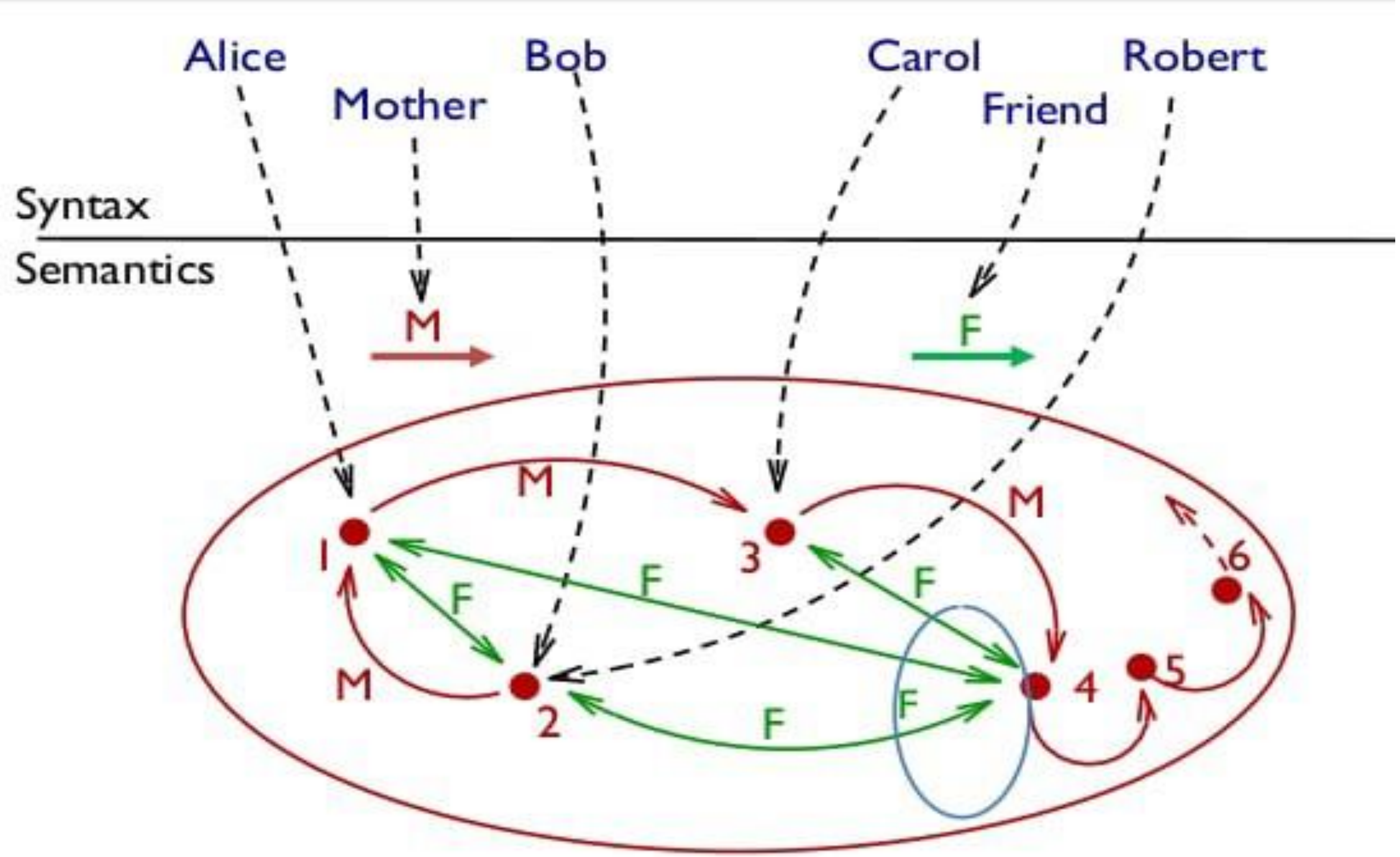

Figure: Language, Domain and Interpretation function

Interpretation function – example (continued)

Let the **domain of interpretation** be the Natural numbers.

$$D = \{1, 2, 3, 4, \ldots\}.$$

Let the **interpretation function** be as follows:

$$I(alice) = 1$$
$$I(bob) = 2$$
$$I(carol) = 3$$
$$I(robert) = 2$$

$$I(motherOf) = \begin{cases} MotherOf(1) = 3 \\ MotherOf(2) = 1 \\ MotherOf(3) = 4 \\ MotherOf(n) = n + 1 \text{ for } n > 3 \end{cases}$$

$$I(friends) = \begin{matrix} (1,2) & (2,1) & (3,4) \\ (4,3) & (4,2) & (2,4) \\ (4,1) & (1,4) & (4,4) \end{matrix}$$

Interpretation of an atomic open formula (continued)

Example.

- I(mother-of(Alice))[a[x/4]]] = 3
- I(mother-of(x))[a[x/4]] = 5

Observation (size of assignment). For any formula, if there are ***n*** variables occurring in it, then there are n^m possible assignments, with ***m*** the cardinality of D assuming D is finite. The intuition underlying the combinatorial explosion of the number of possible assignments is that, in absence of information, all the possible combinations must be considered.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- **Entailment**
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Entailment /model w.r.t an assignment

Definition (entailment wrt an assignment). An interpretation I **entails** ϕ **wrt an assignment** a if:

$$I \models \phi[a]$$

Definition (Model wrt an assignment). An interpretation I is a **model** of ϕ **wrt an assignment** a if it entails ϕ.

Entailment w.r.t. an assignment

Definition (Entailment w.r.t. an assignment). An interpretation I satisfies a formula ϕ w.r.t. the assignment *a* according to the following rules:

$\boldsymbol{I \models t_1 = t_2[a]}$	if and only if	$\boldsymbol{I(t_1)[a] = I(t_2)[a]}$
$\boldsymbol{I \models P(t_1, \ldots, t_n)[a]}$	if and only if	$\boldsymbol{< I(t_1)[a], \ldots, I(t_n)[a] > \in I(P)}$
$I \models (\phi \wedge \psi)[a]$	if and only if	$I \models \phi[a]$ and $I \models \psi[a]$
$I \models (\phi \vee \psi)[a]$	if and only if	$I \models \phi[a]$ or $I \models \psi[a]$
$I \models (\phi \supset \psi)[a]$	if and only if	$I \not\models \phi[a]$ or $I \models \psi[a]$
$I \models \neg \phi[a]$	if and only if	$I \not\models \phi[a]$
$I \models (\phi \equiv \psi)[a]$	if and only if	$I \models \phi[a]$ if and only if $I = \psi[a]$
$I \models (\phi \oplus \psi)[a]$	if and only if	$I \models \phi[a]$ if and only if $I \not\models \psi[a]$
$\boldsymbol{I \models \exists x\, \varphi[a]}$	if and only if	**there is a** $\boldsymbol{d \in D}$ **such that** $\boldsymbol{I \models \varphi[a[x/d]]}$
$\boldsymbol{I \models \forall x\, \varphi[a]}$	if and only if	**for all** $\boldsymbol{d \in D, I \models \varphi[a[x/d]]}$

Entailment w.r.t. an assignment – observations

Proposition (Interpretation and model). A model of a formula (theory) is an interpretation which entails a formula (theory). Same as LoP.

Notation (Model and interpretation). Being models interpretations, we write and say that intepretations entail formulas. Same as LoP.

Observation (From LoP to LoI entailment wrt an assignment). The notion of entailment wrt an assignment extends LoP Entailment via the four components written in bold. The last two handle the new existential and universal connectives, the first introduces the interpretation of equality, to be used whenever one needs to reason about equality. The second is the LoI reinterpretation of the LoP interpretation of atomic functions, see the next item below.

Observation (LoP atomic proposition vs LoI atomic proposition). The key novelty is that, in LoI, atomic propositions, because of predicates, are transparent to the underlying intended model. In fact they allow for a direct encoding of a LoDE EG into the syntax of LoI and, thanks to vaiables, they allow for quantification over the underlying LoDE entities and values.

Observation ($< I(t_1)[a], \ldots, I(t_n)[a] > \in I(P)$). This statement must be read as the process by which a proposition is built as a judgement about (the falsity or truth of) the underlying intended model, exactly as in LoP. The judgement itself changes its value as a function of the terms in input.

Entailment w.r.t an assignment – observations

Observation (atomic formula). An atomic formula is true iff its arguments are in the extension of the predicate.

Observation (Equality). Equality is treated in the same way as any other predicate. It is first computed in the underlying model (e.g., a LoDE graph) and then a judgement is made on the result.

Observation (propositional connectives). The same as in LoP.

Observation (Assignment). Assignments play a key role in atomic, existential and universal formulas. Given an assignment, the idea is to check whether that assignment satisfies the desired requirement, which is different in the three cases (i.e., fixed in atomic formulas, at least one in existential formulas, all cases in universal formulas).

Observation (Assignment and universal / existential quantification). Assignment is key with quantified formulas as it allows to enumerate the values over which they need to be evaluated. Notice that in the case of infinite domains, computing the entailment may not terminate, from which the semi-decidability of LoI.

Observation (Assignment and universal / existential quantification). The entailment of a formula with nested quantification, generates n^m possible combinations with ***n*** the number of variables and ***m*** the cardinality of the domain (including also elements for which the constant naming them is unknown). This is infinite with infinite domains.

Entailment w.r.t an assignment – example

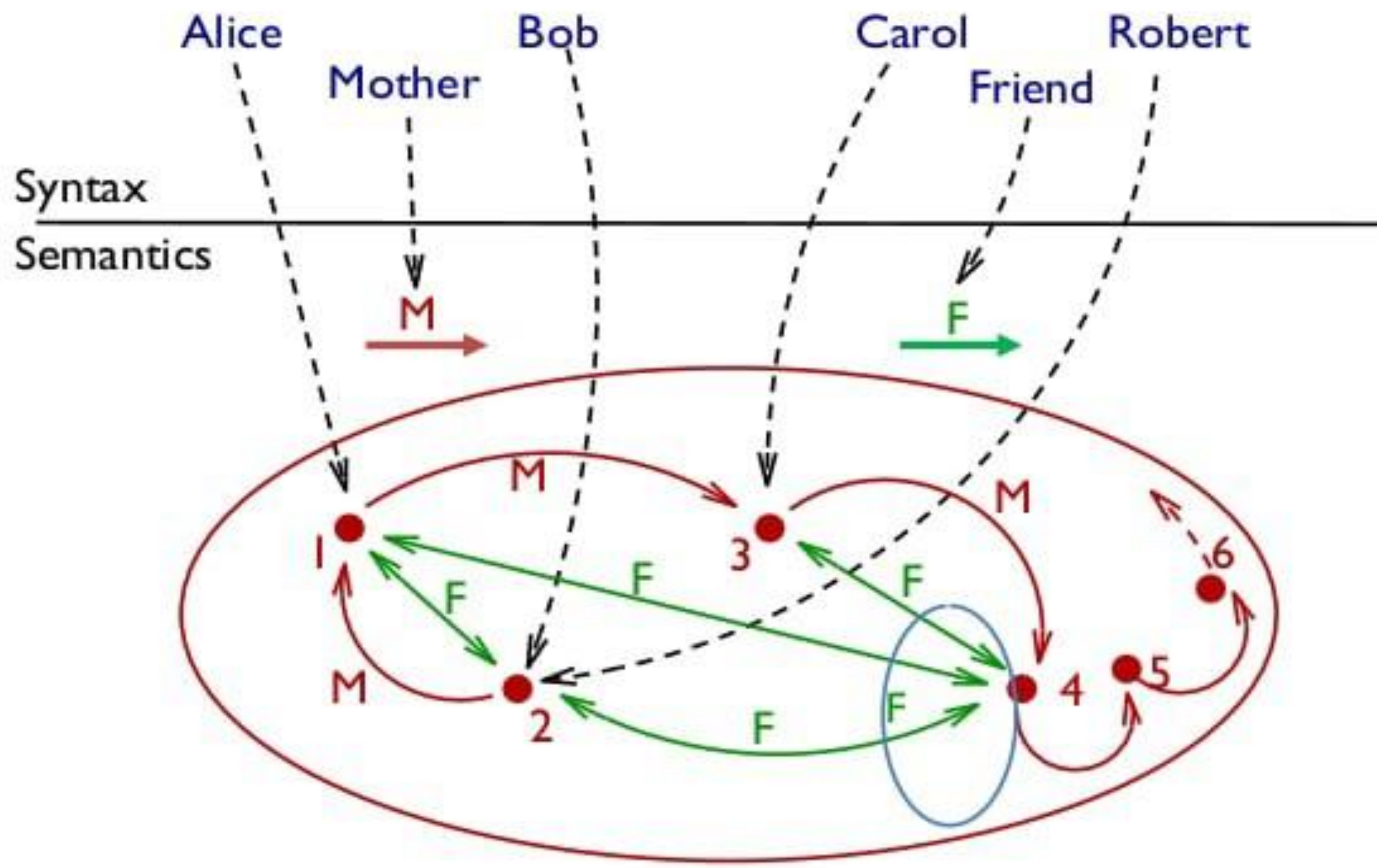

Figure 3: Language, Domain and Interpretation function

Interpretation function – example (continued)

Let the **domain of interpretation** be the Natural numbers.

$$D = \{1, 2, 3, 4, \ldots\}.$$

Let the **interpretation function** be as follows:

$$I(alice) = 1$$
$$I(bob) = 2$$
$$I(carol) = 3$$
$$I(robert) = 2$$

$$I(motherOf) = \begin{cases} MotherOf(1) = 3 \\ MotherOf(2) = 1 \\ MotherOf(3) = 4 \\ MotherOf(n) = n + 1 \text{ for } n > 3 \end{cases}$$

$$I(friends) = \begin{matrix} (1,2) & (2,1) & (3,4) \\ (4,3) & (4,2) & (2,4) \\ (4,1) & (1,4) & (4,4) \end{matrix}$$

Entailment w.r.t an assignment – example (continued)

I |= mother-of(Alice)[a] = 4

I |= (mother-of(x)[a[x/4]]) = 5

I |= (mother-of(x)[a[x/3]]) = 5

I |= friends(x, x) iff x := 4

I |= ∃*x* friends(x, y) iff y := 2, 1, 4, 3

I |= ∀*x* friends(x,y) iff y := 4

I |= friends(x,y) ∧ x=y iff (x, y):= (4,4)

I |= friends(x , y)

iff (x, y):= (1,2), (2,1), (4,1),
(1,4), (4,2), (2,4),
(4,3), (3,4), (4,4)

Entailment / model

Definition (Entailment). An interpretation I **entails** ϕ, in formulas,

$$I \models \phi$$

if there exists an assignment a such that

$$I \models \phi[a].$$

Definition (Model). An interpretation I is a **model** of ϕ if it entails ϕ.

Entailment with closed formulas

Proposition (Assignment invariance). Let $[a^1]$ and $[a^2]$ be two assignments. Then

$$I \models \phi[a^1] \quad \text{if and only if} \quad I \models \phi[a^2]$$

when $[a^1]$ and $[a^2]$ coincide on the variables free in ϕ.

Observation (Assignment invariance). $[a^1]$ and $[a^2]$ in the above proposition can be different on all the variables which are not free in ϕ. These are the only ones who have an impact on entailment, as there are the only ones where the choice is arbitrary.

Observation (Assignment invariance). The LoI assignment invariance property generalizes the LoP interpretation equivalence property. The intuition is that only what occurs in the goal formula is relevant to entailment.

Proposition (Closed formulas) With closed formulas, entailment does not depend on assignments. Therefore, notationally, we omit the assignment and write ϕ rather $\phi[a]$.

Observations (Closed formulas). The proposition for closed formulas is a corollary of the proposition on assignment invariance.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- **The meaning of logical connectives**
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Quantifiers

Example (Validity). The following formulas are:

$\forall x.\ \phi(x) \equiv \neg\exists x.\neg\phi(x)$	Valid
$\forall x.\exists x.\ \phi(x) \equiv \exists x.\ \phi(x)$	Valid
$\exists x.\forall x.\ \phi(x) \equiv \forall x.\phi(x)$	Valid
$\forall x.\ \phi(x) \equiv \exists x.\phi(x)$	not Valid (one implication yes)
$\forall x.\exists y.\ \phi(x, y) \equiv \exists y.\forall x.\ \phi(x, y)$	not Valid (one mplication yes)

Observation (non valid formulas). The two non valid formulas are valid with the implication left-to-right.

Quantifiers and propositional connectives

Example (Validity) The following formulas are:

$\forall x.(\phi(x) \wedge \psi(x)) \equiv \forall x.\ \phi(x) \wedge \forall x.\ \psi(x)$	Valid
$\exists x.\ (\phi(x) \vee \psi(x)) \equiv \exists x.\ \phi(x) \vee \exists x.\ \psi(x)$	Valid
$\forall x.(\phi(x) \vee \psi(x)) \equiv \forall x.\phi(x) \vee \forall x.\psi(x)$	not Valid (one implication yes)
$\exists x(\phi(x) \wedge \psi(x)) \equiv \exists x.\phi(x) \wedge \exists x.\psi(x)$	not Valid (one implication yes)

Observation (non valid formulas). The first non valid formula is valid with the implication right-to-left, the second with the implication left-to-right

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- The meaning of logical connectives
- **Tell**
- Ask – Reasoning problems
- Reasoning problems – correlations
- Key notions

Tell – Model building (the same as LoP)

Intuition (Model building). The model building is performed in three steps

- **(Step 1):** Define the LoI reference model, that is, the set of LoDE assertions which describe the facts which are true in the model
- **(Step 2):** Define the LoI language, that is, the set of atomic propositions and logical connectives which are used to judge what is true / false in the model
- **(Step 3):** Define the LoI theory, that is, the set of (atomic and complex) propositions which constrain what is the case in the model by:
  - (1) specifying the negative knowledge,
  - (2) completing the partial information encoded by the model, and
  - (3) putting further constraints on what is the case via complex propositions (usually crucially linked to the goal to be entailed).

Tell – Model building (step 1) (almost the same as LoP)

Intuition (Define the LoI Reference Model). The first step is articulated in five phases:

- **(Phase 1a)** Define the set of ground (LoE-like) assertions of the EG
- **(Phase 1b)** Define the set of universally quantified (LoD-like) language definitions
- **(Phase 1c)** Define the set of universally quantified (LoD-like) knowledge descriptions
- ~~**(Phase 1d)** Perform the LoD unfolding~~ (no need as syntax is already mapped)
- ~~**(Phase 1e)** Perform the LoDe expansion~~ (no need as syntax is already mapped)

~~**Observation (Define the LoI reference model).** Any of the first three steps is optional. Step 1d and Step 1e are performed only when needed. The key observation is that LoI propositions can be built by espressing judgements on all three LoDE components: ground facts about entities, facts about defined etypes, facts about language concepts.~~

Tell – Model building (step 2) (almost the same as LoP)

Intuition (Define the LoP/LoI Language). The second step is articulated in three phases:

- **(Phase 2a)** Select which LoDE assertions are going to be judged
- **(Phase 2b)** Select a uniform method for encoding a LoDE assertion a into a LoP assertion a'^{+}, a'^{-}. This in turn is composed of two steps
  - (1) How to encode a structured formula into an atomic formula, e.g., from *HasFriend(Stefania#1,Paolo#1)* to *HF-S.P*
  - (2) which of the possible positive or negative encodings a'^{+}, a'^{-} select and how to encode them in the proposition name, e.g., from *HF-S.P to HF-S.P0* and *HF-S.P1*
- **(Phase 2c)** Select the logical connectives, not necessarily used to write complex propositions

Intuition (Phase 2b). There is a std encoding which performs a 1-to-1 mapping (see later).

Observation (Differences from LoP to LoI). The process with LoP and LoI is exactly the same. The key difference is that the encoding of LoDE facts inot LoI propositions can be done one-to-one given that the LoDE language (and domain of interpretation) is a subset of the LoI language (and domain of interpretation).

Tell – Model building (step 3) (the same as LoP)

Intuition (Define the LoP Theory). The third step is articulated in three phases:

- **(Phase 3a).** Select the LoDE assertions which are going to be judged. This usually turns out to be a set of atomic or conjunctions of atomic propositions
- **(Phase 3b).** Select the negative knowledge, implicitly encoded in the LoDE theory, to be made explicit in the LoP theory. This usually turns out to be a set of negations, or disjointness or implication axioms.
- **(Phase 3c).** Select the partial knowledge, implicitly encoded in the LoDE theory, to be made explicit in the LoP theory. This usually turns out to be a set of disjunction axioms.
- **(Phase 3d).** Add a set of of LoI axioms which encode the information provided linguistically which refines what is not explicitly stated in LoDE

Observation (Define the LoI theory). Usually, not all the implicit negative and partial knowledge of a LoD theory is made explicit in a LoI theory, in particular when it takes, implicitly or explictly, the form of disjunctions or universal / existential quantified formulas. The reason being that the complexity of reasoning grows exponentially with the number of disjunctions.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- Tell
- **Ask – Reasoning problems**
- Key notions

Reasoning as entailment wrt an assigment

Definition (Model checking). An interpretation I is a **model** of ϕ if there exists an assignment a such that

$$I \models \phi[a].$$

Definition (Satisfiability). A formula ϕ is **satisfiable** if there is an interpretation I and an assignment a such that

$$I \models \phi[a].$$

Definition (Validity). A formula ϕ is **valid** if, for every interpretation function I and every assignment a, we have

$$I \models \phi[a].$$

Definition (Logical Consequence). A formula ϕ is a **logical consequence** of a set of formulas T, in symbols T $\models \phi$, if for every interpretation I and every assignment a

$$I \models T[a] \quad \text{implies} \quad I \models \phi[a].$$

where $I \models T[a]$ means that I entails all the formulas in T under a.

Observation (Reasoning as entailment wrt an assignment). Remember that entailment plays a role only with formulas with free variables.

Reasoning as entailment (same as LoP)

Reasoning Problem (Model checking). Given a theory T and a model M, check whether M |= T.

Reasoning Problem (Satisfiability). Given a theory T, check whether there exists a model M such that M |= T .

Reasoning Problem (Validity). Given a theory T, check whether for all models M, M |= T .

Reasoning Problem (Unsatisfiability). Given a theory T, check whether there is no model M such that M |= T .

Reasoning Problem (Logical consequence). Given T_1 and T_2, check whether $T_1 \models T_2$;

Reasoning Problem (Logical equivalence). Given T_1 and T_2, check whether $T_1 \models T_2$ and $T_2 \models T_1$.

Reasoning problems – example

Example (Validity and unsatisfiability) Below some examples of validity.

- $\models \exists y. \forall x. p(x,y) \supset \forall x. \exists y. p(x,y)$
- $\models \exists x. \exists y. p(x,y) \supset \neg\forall x.\forall y. \neg p(x,y)$
- $\models \forall x. \forall y. p(x,y) \supset \exists x. \exists y. p(x,y)$
- $\models \forall x. (Sum(x, y) = 12) \supset \forall x. y.(Sum(x, y) = 12)$

Example (Logical consequence). Below some examples of logical consequence

- $\neg \exists x. A(x) \models \forall x. \neg A(x)$
- $\exists x. A, \exists x. A \supset \forall y.B \models \forall y.B$
- $\forall x.(Sum(x, y) = 12) \models \forall x.\exists y.(Sum(x, y) = 12)$

Example (Logical equivalence) Below some examples of logical equivalence.

- $\models \neg\exists x. A(x) \equiv \forall x.\neg A(x)$
- $\models \neg\forall x. A(x) \equiv \exists x.\neg A(x)$
- $\models \exists x. \exists y. A(x, y) \equiv \exists y.\exists x.A(x, y)$
- $\models \forall y.\forall x. A(x, y) \equiv \forall x.\forall y. A(x, y)$

Addendum* - Reasoning Problems

Observation (LoP Reasoning and LoI Reasoning). The reasoning problems are the same. As from the definition of entailment, the propositional connectives and, therefore, the consequent propositional reasoning is the same. The differences come with the quantifiers. The extent to which LoI reasoning can be reduced to LoP reasoning depends on whether the domain of interpretation is finite or infinite.

Observation (LoI reasoning – finite domains). In this case LoI reasoning reduces to LoP reasoning and therefore to the use of truth tables and, more practically, to the use of the DPLL decision procedure. There is however a first preliminary step where the universal and existential statements are expanded into LoP propositions. The drawback (due to the increased expressibility of LoI with respect to LoP) is that the number of propositions, and therefore the search space has an exponential explosion.

Observation (LoI reasoning – LoDE EGs). The use of LoI for reasoning about KGs is a special case of reasoning about finite domains, the reasoning being the limited expressibility of LoDE. In practical terms, as also later highlighted, the process is just an extension of the use of LoP for reasoning about LoDE EGs.

Observation (LoI reasoning – infinite domains). While in principle the process is the same as with finite domains, in practice the problem requires a totally different approach. The problem is that with infinite domains it is impossible to perform the necessary preliminary expansion of the quantified formulas into ground formulas as this would generate infinitely long formulas. There is therefore a need to devise alternative methods, e.g., Tableau methods, which allow for the interleaving of partial quantifier expansion and propositional reasoning.

**Addendum slides are added to provide general background information. They are not part of the course material and, therefore, not a topic of the exam.*

Addendum* - LoI semi-decidability

Observation (Infinite domains, LoI semi-decidability). The infinity of the domain of the interpretation is the reason why LoI is in general semi-decidable. In fact, if a formula is entailed, then, sooner or later, you will prove it. If a formula is not entailed then you will never be able to prove it, that is, the process will potentially never terminate. The problem is that when, after investing a certain amount of resource you have not succeeded in proving a formula, you do not know whether this is because you chose the wrong search strategy or, simply, because the formula is not provable.

Observation (LoI semi-decidability, exploiting the law of the excluded middle). Given that the law of the excluded middle holds in LoI, then one could be tempted to run Loi reasoning, in parallel, on both the input formula and the negation of the input formula. This will improve the situation in case the input formula is unsatisfiable. In this case, the second reasoning process would eventually terminate while the first would not. However this is not a complete solution. Because of Goedel’s incompleteness theorem, there are formulas such that both the formula and the negation of the formulas are not entailed. In which case the process would not terminate for both formulas. An example of such formula is the LoI formula saying “This formula is not provable” which is not provable in any consistent theory. In fact, proving this formula would allow to prove its negation thus making the theory inconsistent.

Observation (Decidability, semidecidability). There are LoI theories which allow for infinite domains which are decidable, namely, such that the entailment of a formula can always be computed in a finite amount of time. It depends on the ability of a theory to self-reference itself. Thus, for instance, if one takes Peano Arithmetics (PA) the LoI axiomatization of reasoning about the natural numbers, then PS is decidable if one considers only *plus* and *successor* but undecidable if one adds *times*.

Addendum slides are added to provide general background information. They are not part of the course material and, therefore, not a topic of the exam.

Addendum* - LoI essential semi-decidability

Observation (Essential semi-decidability and incompleteness). PA, when modeling times is essentially semi-decidable, that is, if one adds the unprovable formula as an axiom, then it will be possible to generate a new "This formula is not provable" which will be unprovable in the new theory.

Observation (Essential incompleteness and reducibility of humans to machines). The essential incompleteness of PA is somehow (may be not?) relevant to AI as various people have argued that this is evidence that it is impossible to develop AI machines which are as intelligent as humans. The most relevant advocate of this position is the book "The emperor' new mind: concerning computers, minds, and the laws of physics" by the Nobel Prize, mathematical physicist, Roger Penrose. You can also find many other papers on related topics by the same author.

Observation (Inessential incompleteness). As discussed extensively in the course, most theories, including LoI theories, are partial, that is incomplete. Namely they do not describe all the truths in the intended model (which, in the case of PA, would be all the truths of arithmetics). As we know from previous lectures, this type of incompleteness, that we call inessential incompleteness, can be easily fixed by adding the missing truths as axioms. The trick is to deal if **inessential incompleteness** is that of reducing the number of models of the theory, possibly – if of interest – down to a single model, thus building a maximal theory.

Observation (Incompleteness in AI). In AI the issue of essential incompleteness plays little or no role in terms of reasoning, the main reason being that we deal for most of the time with finite domains. Inessential incompleteness plays instead a huge role as, as extensively discussed, the theories that humans build, even when formalizing AI/CS models, e.g., ER /EER models, DBs, KGs, are always incomplete. This a huge AI research area, which goes under the heading of **commonsense knowledge** and **commonsense reasoning**, where people try to find mechanisms apt at solving this problem. The world logics LoE, LoD, LoDE are a first step towards providing a logical formalization and solution to these problems

Observation (Incompleteness in CS). The issue of infinite domains and essential incompleteness is instead an important problems when reasoning about (the correctness and other properties) of programs. This is because all these programs work on infinite domains (e.g., the natural numbers, the reals, strings) and allow for programs which, because of loops and recursion, can loop for ever. The problem of termination is a key issue in formal methods.

**Addendum slides are added to provide general background information. They are not part of the course material and, therefore, not a topic of the exam.*

LoP – The Logic of Propositions

- Intuition
- Definition
- Domain
- Language
- Interpretation function
- Entailment
- The meaning of logical connectives
- Tell
- Ask – Reasoning problems
- **Reasoning problems – correlations**
- Key notions

Reasoning problems - Correlations (same as LoP)

Theorem. If a formula is valid, then it is also satisfiable, and it is also not unsatisfiable. That is:

Validity implies **Satisfiability** equivalent to **not Unsatisfiability**

Theorem. If a formula is unsatisfiable, then it is also not satisfiable, and also not valid. That is:

Unsatisfiability equivalent to **not Satisfiable** implies **not Valid**

Example: Valid, Satisfiable or Unsatisfiable?

Prove that

- Blue Fomulas are valid,
- Magenta Formulas are satisfiable but not valid
- Red Formulas are unsatisfiable.

Satisfiable:

Valid:

$(\forall x.p(x)) \supset (\exists y.p(y))$

$\forall x.(p(x) \supset (\exists y.p(y)))$

$\exists x.p \supset q \equiv (\forall x.p) \supset (\exists x.q)$

$\exists x.\forall y.p(x,y) \supset \forall y.\exists x.p(x,y)$

Not Valid:

$\forall x.\exists y.p(x, y)$

$\forall y.\exists x.p(x, y) \supset \exists x.\forall y.p(x, y)$

Unsatisfiable:

$\exists x.(p(x) \wedge \neg p(x))$

$\neg\forall x.(p(x) \supset p(x))$

$\neg(p(a) \supset \exists x.p(x))$

$\forall x.p(x) \supset \forall x.q(x) \wedge \forall x.p(x)) \supset \forall x.q(x)$

Exercise: Valid, Satisfiable or Unsatisfiable?

Exercise (Valid, satisfiable, unsatisfiable). Say whether these formulas are valid, satisfiable or unsatisfiable.

1. $\forall x.P(x)$
2. $\forall x.P(x) \supset \exists y.P(y)$
3. $\forall x.\forall y.(P(x) \supset P(y))$
4. $P(x) \supset \exists y.P(y)$
5. $P(x) \vee \neg P(y)$
6. $P(x) \wedge \neg P(y)$
7. $P(x) \supset \forall x.P(x)$
8. $\forall x.\exists y.Q(x, y) \supset \exists y.\forall x.Q(x, y)$
9. $x = x$
10. $\forall x.P(x) \equiv \forall y.P(y)$
11. $x = y \supset \forall x.P(x) \equiv \forall y.P(y)$
12. $x = y \supset (P(x) \equiv P(y))$
13. $P(x) \equiv P(y) \supset x = y$

Reasoning problems – Correlations (same as LoP)

Theorem. The validity, satisfiability and unsatisfiability of a formula and of its negation correlate as follows:

If A is	then ¬ A is
Valid	**Unsatisfiable**
Satisfiable	**Not Valid**
Not Valid	**Satisfiable**
Unsatisfiable	**Valid**

Entailment properties (NEW!) (same as LoP)

Deduction theorem (Logical consequence, validity):

$$\Gamma, \phi \vDash \psi \text{ if and only if } \Gamma \vDash \phi \supset \psi$$

Observation 1: The deduction theorem explains (left to right) the meaning of implication. Implication is how we express logical consequence in language.

Observation 2: It also says (right to left) that from absurdity (i.e, $P \wedge \neg P$),we can derive everything, any formula (and assertion) A.

Entailment properties (NEW!) (same as LoP)

Refutation principle (Logical consequence, unsatisfiability):

$$\Gamma \vDash \phi \text{ if and only if } \Gamma \cup \{\neg \phi\} \text{ is unsatisfiable}$$

Observation 1: The refutation principle explains the meaning of negation. It captures the fact that absurdity (i.e, $P \wedge \neg P$) cannot be satisfied by any model depicting facts in the real world.

Observation 2: Algorithmitically, it suggests how to reason backwards from goals.

Reasoning problems – Correlations (same as LoP)

Logical Consequence (LC). Two possibilities

- Use the deduction theorem to reduce LC to a VAL problem
- Use the refutation principle to reduce to an UNSAT problem

Logical Equivalence (LE) reduces to LC.

LoI – The Logic of Interaction

- Intuition
- Definition
- Domain
- Language
  - The language of atomic propositions
  - The language of propositions
- Interpretation function
  - Atomic closed formulas
  - Atomic open formulas
- Entailment
- Tell
- Ask – Reasoning problems
- Reasoning problems – correlations
- **Key notions**

Key notions

- Constants, variables, functions, terms
- Predicates , atomic propositions
- Existential quantifier, universal quantifier,
- Free and bound occurrences of variables, variable assignment
- Ground formulas, closed formulas, open formulas
- Interpretation of open and closed formulas
- Entailment, entailment wrt an assignment
- Model checking, satisfiability, validity, unsatisfiability, logical consequence, logical equivalence (also wrt an assignment)
- Decidability, (essential) semi-decidability
- Deduction theorem, refutation principle

UNIVERSITY OF TRENTO - Italy

LoI

The Logic on Interaction

(HP2T)