Fausto Giunchiglia

The Logic of Machine Intelligence

From practice to theory to practice

October 13, 2023

Springer Nature

Contents

Part I World and representation

Part II World models and logics

Part III The logics of recognition

Part I
World and representation

Chapter 1
The world and the mind

We often confuse the world with our mental representation of the world itself. Is this a correct assumption?

1.1 Frames of mind

Following social theory, a *frame* is a schema of interpretation, a collection of anecdotes and stereotypes, that individuals rely on to understand and respond to events [5]. People build a series of mental representations of the world through biological and cultural influences. They then use these filters to make sense of the world. The choices people make are influenced by frames. Participation in a language community necessarily influences an individual's perception of the meanings attributed to words or phrases.

Example 1.1 (The car accident) In [8] *two experiments are reported in which subjects viewed films of automobile accidents and then answered questions about events occurring in the films. The question, "About how fast were the cars going when they smashed into each other?" elicited higher estimates of speed than questions which used the verbs collided, bumped, contucted, or hit in place of smashed. On a retest one week later, those subjects who received the verb smashed were more likely to say "yes" to the question, "Did you see any broken glass?", even though broken glass was not present in the film. These results are consistent with the view that the questions asked subsequent to an event can cause a reconstruction in one's memory of that event*" (Quote from the abstract of [8]).

Example 1.2 (The Asian disease problem) Tversky and Kahneman [6] demonstrated systematicity when the same problem is presented in different ways, for example in the *Asian disease problem*. Participants were asked to "imagine that the U.S. is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume the exact scientific estimate of the consequences of the programs are as follows." The first group of participants was presented with the following choice. In a group of 600 people,

- Program A: "200 people will be saved";
- Program B: "there is a 1/3 probability that 600 people will be saved, and a 2/3 probability that no people will be saved"

72% of the participants preferred program A, 28%, opted for program B. The second group of participants was presented with a different choice. In a group of 600 people,

- Program C: "400 people will die";
- Program D: "there is a 1/3 probability that nobody will die, and a 2/3 probability that 600 people will die"

In this decision frame, 78% preferred program D, with the remaining 2% opting for program C. Programs A and C are identical, as are programs B and D. The change in the decision frame between the two groups of participants produced a preference reversal: when the programs were presented in terms of lives saved, the participants preferred the secure program, A (= C). When the programs were presented in terms of expected deaths, participants chose the gamble D (= B).[4].

1.2 Optical Illusions

An optical illusion, or visual illusion, occurs when the visual system creates a perception that seems different from the surrounding reality. The main categories of illusions are physical, physiological and cognitive, each with types such as ambiguities, distortions, paradoxes and fictions. Examples include the apparent curvature of a stick in water (physical distortion), the effect of adapting to movement (physiological paradox), and the residual impression of an image (physiological fiction). Pathological visual illusions result from pathological changes in physiological mechanisms and can lead to visual hallucinations. These illusions can be used in the monitoring and rehabilitation of psychological disorders such as phantom limb syndrome and schizophrenia.

Example 1.3 (Herman Grid) A demonstration of how our perception can deceive us is *Herman Grid* [1], which is an optical illusion in which a grid of white dots on a black background appears to create dark spots at the points of intersection. Although we know that there are no spots actually present, our perception tricks us into believing that they are present. This example demonstrates how our visual perception (and so our senses) can deviate from objective reality.

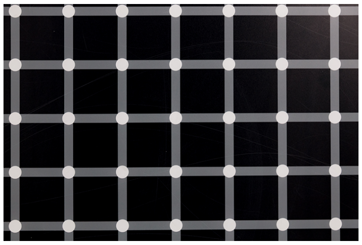


Fig. 1.1 Herman Grid

Example 1.4 (Kanizsa triangle) The illusion consists in the fact that looking at the image we hallucinate to see two triangles and 3 circles, but actually none of them is there.

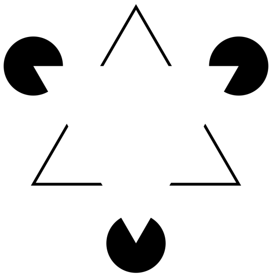


Fig. 1.2 Kanizsa Triangle

Example 1.5 (Pareidolia) Pareidolia is the tendency for perception to impose a meaningful interpretation on a nebulous stimulus, usually visual, so that one sees an object, pattern, or meaning where there is none. For example, we tend to see faces everywhere, even in the surface of the Moon.

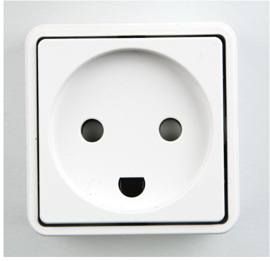


Fig. 1.3 Pareidolia

Example 1.6 (My Wife and My Mother-in-Law) This is a famous optical illusion in which viewers can see either a young woman looking away or an old woman in profile, depending on how they interpret the drawing's lines. The illusion plays on our ability to switch between different perspectives.

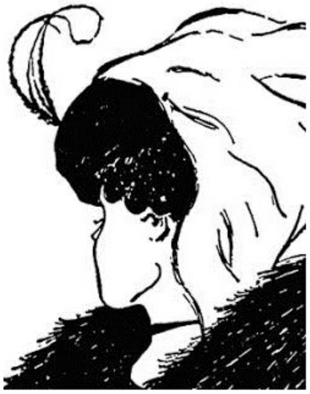


Fig. 1.4 My Wife and My Mother-in-Law

Example 1.7 (Impossible Trident) Also known as the "blivet", this illusion depicts a three-pronged trident that mysteriously transforms into two cylindrical shafts at the other end. This illusion plays with our perception of three-dimensional space.

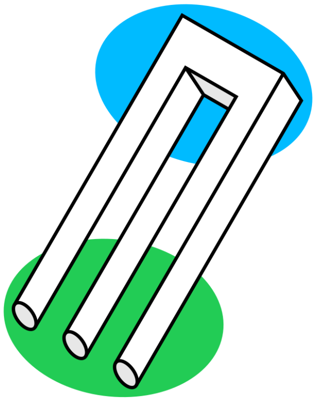


Fig. 1.5 Impossible Trident

Example 1.8 (Rubin's Vase) This is a classic example of figure-ground perception. Viewers can either see a vase in the center or two faces in profile facing each other.

The brain can switch between either interpretation but cannot see both at the same time.

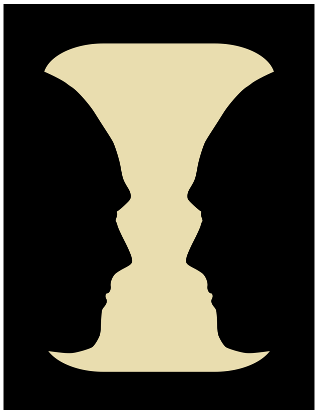


Fig. 1.6 Rubin's Vase

Example 1.9 (Penrose Triangle) This is an "impossible object" that cannot exist in three-dimensional space. It appears to be a solid object made of three straight beams of square cross-section, but its construction is impossible.

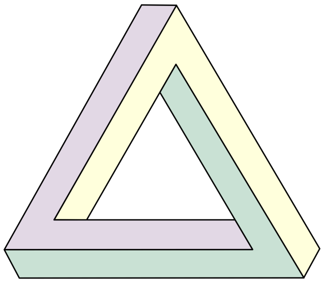


Fig. 1.7 Penrose Triangle

1.3 Mind Fallacies

A fallacy is reasoning that is logically invalid, or that undermines the logical validity of an argument. All forms of human communication can contain fallacies. The use of fallacies is common when the speaker's goal of achieving common agreement is more important to them than utilizing sound reasoning.

Fallacies can be classified depending on their structure (formal fallacies) or on their content (informal fallacies). A formal fallacy, also called a deductive fallacy or logical fallacy [2], represents a type of reasoning that loses validity due to a flaw in its logical structure. This flaw can be clearly represented within a standard logical system, such as propositional logic. In other words, it is a deductive argument that is invalid. Even though the premises of the argument might be true, the conclusion drawn from it is still false. Informal fallacies, the larger group, may then be subdivided into categories such as improper presumption, faulty generalization, and error in assigning causation and relevance, among others.

We provide below various examples of informal and formal fallacies.

Example 1.10 (Cognitive Bias) Cognitive biases [12] are an example of informal fallacies. They represent systematic patterns of deviation from the norm and rationality in the evaluation process. The Asian disease example, see above, is an instance of cognitive bias.

Example 1.11 (Misconceptions) Misconceptions are informal fallacies. A common misconception is a perspective or data that is often considered to be true but is actually false. Usually, such misunderstandings stem from entrenched traditions (such as gossipy tales), stereotypes, superstitions, fallacies, misinterpretations of science, or the spread of pseudoscience. Some of these misunderstandings are considered urban legends and often contribute to moral alarmism.

Example 1.12 (Cognitive Distortion) Cognitive distortions are an informal fallacy. They can be traced to "thinking fallacies," representing irrational or distorted ways through which we process information and perceive reality. Some of the main thinking fallacies involved include

- overgeneralization, which draws overly broad conclusions from a single negative event;
- mental filtering, which focuses attention only on the negative aspects of a situation;
- over-labeling, which assigns negative labels to oneself or others based on mistakes or failures;
- dichotomous thinking, which considers only extremes without acknowledging nuance;
- emotional reasoning makes one believe that one's feelings reflect objective reality;
- personalization leads one to interpret events as being directly related to oneself;
- Negative prediction involves predicting the worst without concrete evidence;
- Catastrophism makes one imagine the worst as the only possibility, ignoring alternatives, while sample selection draws general conclusions from a limited set of data or experiences.

Example 1.13 (Paradoxes) Paradoxes are examples of formal fallacies. Paradoxes are situations or statements that seem contradictory or contraintuitive, often challenging our normal thinking and expectations. They are intellectual puzzles that can cause confusion and amazement as they violate our common understanding of logic or the laws of reality.

Some paradoxes emerge from fallacious reasoning, where it appears that rules of thought are correctly applied, but the end result is nonetheless contradictory or nonsensical. These paradoxes teach us the importance of carefully examining the premises and inferences behind an argument.

Conversely, there are paradoxes that emerge from complex situations or situations that fall into categories of mathematical or philosophical problems. These paradoxes can challenge our intuition and reveal the limitations of our knowledge. In some cases, these paradoxes can highlight deep issues in the very structure of our rational thinking.

A particular type of paradoxes, called antinomies, is characterized by the presence of self-contradictions in situations where we would expect consistency. These paradoxes can be used to highlight the inherent challenges in dealing with concepts such as truth, description or infinity.

Example 1.14 (The Map - Territory confusion) The map-territory relationship [7] is a fundamental concept for understanding the fallacies of the human mind. Essentially, it points out that the mental representations we create, such as concept maps, models and interpretations, are not identical to the reality they seek to represent. This concept detects several distortions in the perception and interpretation of human reality. For example, people often generalize and make incorrect conclusions based on limited experiences, confirming their own biases and ignoring conflicting information. Cultural beliefs influence mental maps, leading to distorted perceptions. Cognitive distortions and overconfidence in representations can lead away from objective reality. In summary, understanding the map-territory relationship prompts us to be aware of discrepancies between our mental representations and actual reality, helping us to avoid wrong thinking traps and maintain a critical perspective.

1.4 So What?

In this chapter, we explored the inherent flaws in human thinking, such as fallacies, biases and misconceptions, which can affect our understanding of reality and decision making. However, we can adopt several strategies to overcome these challenges and promote more accurate, rational and logic-based thinking.

Logic is a crucial tool for avoiding fallacious reasoning. Formalizing thinking through logic provides us with a structured framework for evaluating arguments and drawing conclusions. The systematic approach of logic helps us recognize and foil fallacious reasoning. Learning to identify the premises, inferences and conclusions in an argument enables us to detect logical errors or inconsistencies. This is key in Computer Science and even more in Artificial Intelligence.

Chapter 2
Representations

The various types of fallacies described in Section 1.3 raise the issue of whether it is possible to deal with them. But why? To deal in which sense? Modulo some extreme cases, humans and humanity have been able to develop well and grow in time despite the pervasiveness of fallacies in human interactions with the world and with others. Two are the main reasons underlying this work. The first is that, because of the Web and social platforms, now humans are able to interact with people that are hardly known and with very different cultures and, furthermore, they get in contact with parts of the world that they never visited physically. The probability of misalignments and misunderstandings among people has grown immensely. The second is that, in this era where we want to build CS and AI systems which are more and more complex, more and more intelligent, and which pervasively interact with people in their everyday lives, we need to have systems which are robust, trustable, and whose behaviour we fully understand and also control.

The first step is to find a way to build representations of the world which are not ambiguous and which can be used as the basis for solving the interpretations problems highlighted in Section 1.3. This is the gaal of this section.

2.1 The Semantic Gap

Living organisms perceive reality, what we call the world, through the lenses of their perceptions organs. This process is not neutral. Different species and even different humans perceive the world differently. We talk of Semantic Gap relating to the impossibility for humans and machines to perceive the world as it really is, or even in the same way. The Semantic Gap is the source of the pervasive misalignment of the mental models of the world that humans, and also machines, build.

Intuition 2.1 (World) The **world** is what we perceive through the five senses and assume it exists. It is the spatio-temporal dimension in which humans live and interact with other humans and everything else around them.

Fig. 2.1 The 5 Senses: sight, hearing, touch, taste and smell.

Intuition 2.2 (Memory) When we perceive the world we create in our mind a **memory** of what we have perceived, the memory being itself a part of the world.

Intuition 2.3 (Mental Representations) **Mental representations** are a part of a person's memory. Mental representations are such that there is a correspondence between their contents and what is the case in the world they describe.

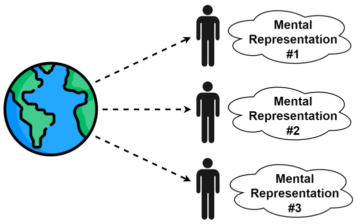


Fig. 2.2 Mental Representations.

Observation 2.1 (Mental representations) All humans have their own mental representations of the world. They are a fundamental mechanism enabling human knowledge, reasoning, action and communication.

Intuition 2.4 (Semantic gap) The **semantic gap** is the difference between the world and a human's mental representation of the world itself, what (s)he has perceived.

Observation 2.2 (Semantic gap) Most of the details of how perception and memory operate and how the different processes compose to generate memories is largely unknown. However we know that our memories are an encoding of what we perceive and that this encoding is partial and not faithfully representing what caused it.

2.2 Mental Representations

We have two types of mental representations.

Intuition 2.5 (Analogical mental representations) **Analogical mental representations** are mental representations that **depict** the world as we perceive it through the five senses.

Example 2.1 (Analogical mental Representations) We see an apple, we smell its fragrance, we taste it when eating.

Observation 2.3 (Analogical mental representations) Analogical mental representations enable us to acquire information about the world, directly from the world. They are used to act in the world, to learn from what has been previously perceived and to build an understanding of the world itself.

We describe analogical mental representations using languages. We use languages to build mental linguistic representations about the world, as represented in mental analogical representations.

Intuition 2.6 (Language) A **language** is any notation, generated by humans, agreed upon by humans, which allows to describe analogical representations, to reason about them, and to communicate about them to other humans.

Intuition 2.7 (Linguistic mental representations) Linguistic mental representations are mental representations that **describe** mental analogical representations using language.

Example 2.2 (Linguistic mental representations, language) The most important example of languages used in linguistic mental representations are the natural languages, e.g., Italian, and English, as memorized in our mind. Examples of linguistic mental representations are a poem and, in general, any piece of text describing the world that we remember.

Observation 2.4 (Linguistic mental representations) Linguistic mental representations are used to describe what is happening in analogical mental representations. They allow to communicate to other humans about our mental representations (and, thus, indirectly about the world), to learn from what has been previously described or perceived, and to reason in order to to derive unknown facts from what we already know.

Intuition 2.8 (Represent, depict, describe) To **represent** the world means anyone of two things: to depict it or to describe it.

Observation 2.5 (Analogical and linguistic mental representations) Analogical representations depict the world. Linguistic describe (the analogical representations of the world). They both represent the world.

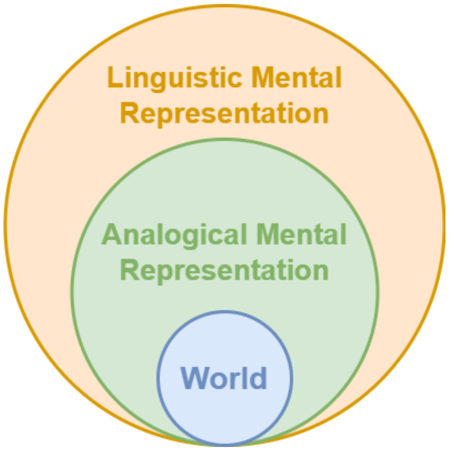

Linguistic Mental Representation
Analogical Mental Representation
World

Fig. 2.3 Diagram of Mental Representations

Observation 2.6 (Partiality of mental representations) Because of the semantic gap, mental representations never describe the world completely. This has consequences. First, there are indefinitely many analogical mental representations that describe the same real world situation. Similarly, there is an indefinite number of linguistic mental representations for the same analogical representation.

Observation 2.7 (Number of mental representations) Because of partiality there are indefinitely many analogical mental representations that describe the same real world situation. Furthermore, there is an indefinite number of linguistic mental representations for the same analogical representation.

Observation 2.8 (Diversity of mental representations) Because of partiality, any two mental representations are necessarily different, depending on the spacetime coordinates under which they are generated, and the purpose of the person who generates them

Example 2.3 (Diversity of mental representations) Two people describing the same trip would do it so differently. For example, one person might have a partial and rough mental representation of the city, based mainly on a few famous tourist sites and the positive experiences he had during the trip. While the other person might have a different mental representation focused on other aspects of the city. She might remember the difficulties she encountered in finding the right way, some less pleasant experiences with locals or bad weather during the trip. Her mental representation might be more influenced by these less positive aspects.

Intuition 2.9 (Consistency and inconsistency of mental representations) We say that any two mental representations are **inconsistent** when it is impossible for those two mental presentations to represent the (same part of the) world, as he know it. **Consistency** means absence of inconsistency. Two consistent mental representations are still diverse but they are compatible in the sense that there is a (analogical representation of the) world which is described by both.

Example 2.4 (Inconsistency of mental representations) It is impossible to have two different objects (e.g., two cats) exactly in the same place in the same moment or the same object (a cat) in two different places in the same moment. Similarly, given an object, certain properties (e.g., being of color blue) prevent other properties from holding (e.g., being red), again in the same moment.

Observation 2.9 (Subjectivity of mental representations) Given the world they perceive, humans build one or more among the many possible mental analogical and linguistic representations of what they have perceived. Each individual has a unique and personal perspective on the world, influenced by different experiences, knowledge and viewpoints.

Observation 2.10 (Subjectivity vs. objectivity of mental representations) Humans may confuse the real world with their mental representations. A consequence is the assumption that (their mental representation of) the world is the same for everybody. Would this be the case would all be living in the same (mental representation of the) world. Because of subjectivity, this assumption turns out to be wrong.

Observation 2.11 (Subjectivity, inconsistency and objectivity) Two subjective mental representations may be (mutually) inconsistent. The presence of inconsistency provides evidence of the subjectivity of the mental representations involved.

2.3 Representations

The subjectivity and heterogeneity of mental representations raises some important questions. Is it possible to guarantee that the mental representations of different people are the same? Or, at least, that they are not mutually inconsistent and also similar enough in some key features, in particular those which are relevant to the problem to be solved? How do we enforce or at least facilitate the construction of similar mental representations

Intuition 2.10 (Representations) A **representation** is a part of the world, developed by the mind of a human, that represents that human's mental representation, and is made accessible, via one of the five senses, to other humans.

As for mental representations, we have two types of representations.

Intuition 2.11 (Analogical Representations) Analogical representations depict analogical mental representations.

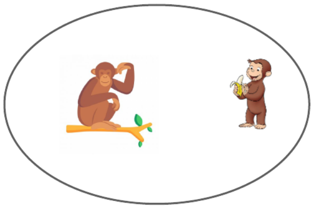
Intuition 2.12 (Linguistic Representations) Linguistic representations describe linguistic mental representations.

Example 2.5 (Representations) The following are examples of representations:

1. Any written natural language text is a linguistic representation, which can be generated on multiple media, for instance, paper, media, projection on a screen;
2. Any spoken natural language stream is a linguistic representation, which can be registered on transcribed on paper;
3. All forms of art, e.g., drawings, statues, paintings, music, monuments, are analogical representations.

Example 2.6 (Linguistic and analogical mental representations)

- There is a tree
- There is a banana
- The monkey is eating a banana
- The monkey is sitting on a tree
- The monkey is scratching his head



Observation 2.12 (Partiality, number, diversity, (in)consistency, subjectivity and objectivity) Observations 2.6 on the partiality, 2.7 on the number, 2.8 on the diversity and 2.9 on the inconsistency of mental representations apply also to representations. Not being in the mind of people, representations cannot be said to be subjective or objective. The question is about the mental representations they generate.

Observation 2.13 (From mental representations to representations to mental representations) The process and consequences of generating representations is well represented in the analogical representation in Figure 2.4. That is: the representation is generated by a single person starting from his/her mental representation and in turn it generates new mental representations in the minds of the people looking at it.

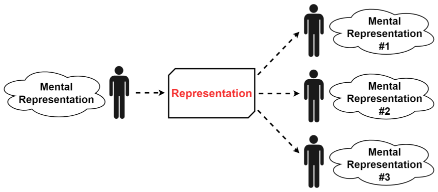


Fig. 2.4 From Mental Representations to Representations and back.

Observation 2.14 (From mental representations to representations) Representations, by their own nature and purpose, are such that there is a correspondence between their contents and those of the mental representations they describe. This is why people generate them.

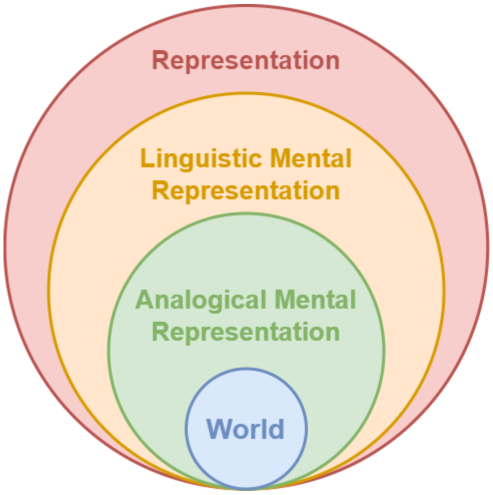

Representation
Linguistic Mental Representation
Analogical Mental Representation
World

Fig. 2.5 Diagram of Representations

Observation 2.15 (From representations to mental representations) There is no guarantee that a representation generates similar subjective mental representations. Think for instance of the many different interpretations, impressions, feelings that a piece of art generates.

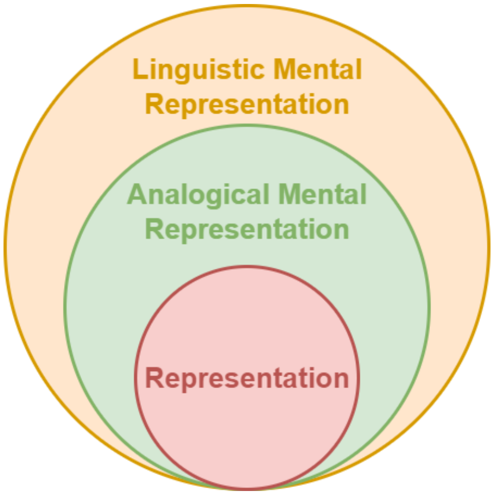

Linguistic Mental Representation
Analogical Mental Representation
Representation

Fig. 2.6 Diagram of Representations

2.4 Exercises

Exercise 2.1 (Linguistic and analogical mental representations) Create a linguistic representation for the analogical mental representation in Figure 2.1.

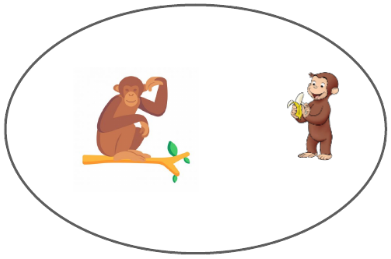


Fig. 2.7 Monkey and banana.

Exercise 2.2 (Linguistic and analogical mental representations) Create an analogical representation for this linguistic mental representation. The phrases are written in Tswana, a language spoken in southern Africa.

- Mongwe le mongwe o tshela mo
- lafatsheng la gagawe go ya ka kitso
- ya gagwe le go ya ka seo a se lemogang.
- Maitemogelo a tlhola seo, puo e letlelela
- go arologana tlhaloso
- le batho ba bangwe, ka moo

Chapter 3
Models and assertional theories

Observation 2.15 may suggest that there is no solution to the problem of subjectivity of mental representations. However this is not the case. The key observation is that representations are built by humans with the specific purpose of making mental representations of the same representation converge as much as possible, minimizing in particular the probability of inconsistencies. The question to be answered is how to build such representations.

3.1 Models

The starting point is analogical mental representations, as our representations start from here. Consider the following example.

Example 3.1 (What is in an analogical representation) Consider the analogical representation depicted in the image in Figure 3.1. We can see three people, that we can assume have names Paolo, Stefania and Sofia, that they are friends, various dogs, the fact that they are one at the right of the other, and of course much more.

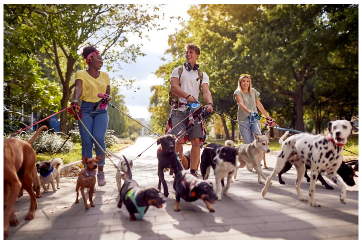


Fig. 3.1 An analogical representation of an everyday situation.

Observation 3.1 (Analogical representations as sets of facts) Any analogical representation, for instance that in Figure 3.1, always depicts various objects, e.g., Sofia, which belong to certain classes, e.g., "Sofia is a person", with certain properties articulated at various levels of complexity, e.g., "Sofia has blond hair", which are doing things, e.g., "Sofia is walking", and are engaged in certain relations with other objects, e.g. "Sofia is a friend Paolo and she is now interacting with her dogs". Despite their heterogeneity, all the statements above share the fact that they describe a certain state of affairs in the world. We call these statements facts. Any analogical representation can be thought of as a set of facts. We call analogical representations described as sets of facts, models

Intuition 3.1 (Fact) A **fact** $\mathtt{f}$ is something happening at certain spacetime coordinates.

Definition 3.1 (Model) A **model** $\mathtt{M}$ is a set of facts $\mathtt{F} = \{\mathtt{f}\}$

$$\mathtt{M} = \{\mathtt{f}\} \tag{3.1}$$

Observation 3.2 (Facts and models) Facts are the atomic, not further decomposable, elements of a model. Note that, contrary to models, facts are taken as a primitive notion and therefore cannot be formally defined

Example 3.2 (The facts of a model represented in Figure 3.1) A model, one among many others, of the situation represented in Figure 3.1 could for instance contain the following facts :

```
Sofia is a person    Paolo is a man
Rocky is a dog       Sofia is near Paolo
Sofia has blond hair Sofia is a friend of Paolo
Rocky is an animal   Rocky is the dog of Sofia
...
```

Observation 3.3 (The subjectivity of facts) Facts are what is observed and is also described, e.g., to third parties. The problem is that, just because of what discussed in Section 2.2 and, specifically what facts are is subjective and hidden in the minds of people who perceive them. How many more and/or different facts from those listed in Example 3.2 could you think of? Indefinitely many! Notice that any fact can be decomposed in any set of simpler facts if this is the current focus of the observer. So, for instance, instead of focusing on Sofia I could focus on her hair, or legs or . . .

Observation 3.4 (Mutually (in)consistent facts in a model) The model of Example 3.2 could be extended to assert the fact that Sofia is a woman or that Paolo is a person. We would however have problems extending it by adding the fact that Paolo is a woman, or that Sofia is a dog, as we would have two mutually inconsistent facts, something that we know cannot happen in the world as we perceive it. See also Observation 2.9. A model cannot contain facts which, at least intuitively. are mutually inconsistent. Beyond this simple example, the issue is how to formalize this intuition and then how to be able to detect it by reasoning about models.

Observation 3.5 (Facts and assertions) A fact, to be a fact, must be linguistically described as such. It is not by chance that in Example 3.2 we pointed to facts via a set of natural language descriptions. We call such descriptions, assertions. The simplest way to think of an assertion is as a declarative natural language sentence articulated in terms of a *subject* being in some more or less complex *relation* with an *object* (as in, e.g., "Stefania is walking with the dogs towards the city center"), or of a *subject* holding a certain more or less complex property (as in, e.g., "Stefania has blond long hair").

3.2 Assertional theories

Observation 3.6 (Assertions and assertional theories) Assertions are indivisible, we say atomic, descriptions of fact. Assertional theories are descriptions of models

Intuition 3.2 (Assertion) An **assertion** a is an atomic linguistic representation of some fact $\mathtt{f}$.

Definition 3.2 (Assertional theory) An **assertional theory** $\mathcal{T}_\mathcal{A}$ is a set of assertions $\mathcal{T}_\mathcal{A} = \{a\}$

$$\mathcal{T}_\mathcal{A} = \{a\} \tag{3.2}$$

We need to state that an assertion is the description of a specific fact and, more in general, that an assertional theory describes a model.

Example 3.3 (An assertional theory of the model represented in Figure 3.1) An assertional theory, one among many others, describing the facts from Example 3.2 in natural language could be, for instance:

Sofia è una persona	Paolo è un uomo	Rocky è un cane
Sofia è vicina a Paolo	Rocky è il cane di Sofia	Sofia è un'amica di Paolo
Rocky è un animale	Sofia ha i capelli biondi . . .	

As can be seen, the assertional theory is in Italian, since being in natural language it can also be expressed in this way, and it could equally be expressed in English.

3.3 Interpretation functions

Definition 3.3 (Interpretation function) Let $\mathcal{I}_A$ be an **interpretation function** of an assertional theory, defined as

$$\mathcal{I}_A : \mathcal{T}_A \rightarrow \mathtt{M}. \tag{3.3}$$

We say that a fact $\mathtt{f} \in \mathtt{M}$ is the **interpretation** of $a \in \mathcal{I}_A$, and write

$$\mathtt{f} = \mathcal{I}_A(a) = a^{\mathcal{I}_A} \tag{3.4}$$

to mean that a is a linguistic description of $\mathtt{f}$. We say that $\mathtt{f}$ is the **interpretation of** a, or, equivalently, that a **denotes** $\mathtt{f}$.

Observation 3.7 (Interpretation function, polysemy) $\mathcal{I}_A$ is assumed to be a function, that is, for any fact there is only one assertion describing it. In fact, we must guarantee that, if two facts f_1 and f_2 are different then they cannot both be the result of the interpretation of the same assertion a, i.e., it cannot be that if $\mathcal{I}_A(a) = f_1$ then also $\mathcal{I}_A(a) = f_2$. This phenomenon, called *polysemy* is pervasive in natural languages and it is one of the main sources of misunderstandings and, therefore, of the construction of diverging mental representations of the same representation. The polysemy of assertions arises directly from the polysemy of words. As examples: the proper name *Java* has three meanings, that is, it is a programming language, a type of coffee beans, and an island. The word *car* ha various meanings. For instance it may mean automobile or a car part of of a train. General words, such as *to do* have more than ten meanings. Polysemy is common to most words, in particular with those words which are most commonly used (people tend to give words their own specific meaning) and it is one of the major complications (not the only one) which arise when building natural language understanding systems.

Observation 3.8 (The non ambiguity of interpretation functions) As from Section 1.3 linguistic descriptions are ambiguous. As from Observation 3.7, one of the main reasons is the polysemy of words. However this ambiguity is in the mind of the listener/reader. The speaker/writer can be assumed to always have in mind the unique analogical representation (s)he is describing. The notion of interpretation function enforces this assumption forcing the speaker/writer to be explicit about the intended meaning.

Observation 3.9 (Interpretation function, synonymity) Two assertions are synonyms when they have the same meaning, that is, the interpretation of two different assertions a_1 and a_2, may denote the same fact f, i.e., $\mathcal{I}_A(a_1) = \mathcal{I}_A(a_2) = f$. Synonymous words are again pervasive in natural languages, in particular with the most common entities. People, and entities in general, have multiple names, e.g., name, surname, name plus surname, nicknames, which are synonymous. Multiple languages generate multiple names of the same entity (e.g., Great Britain, Gran Bretagna). There are also synonymous nouns, for instance *car* and *automobile*. Notice how the word *car* is both polysemous and synonymous. This is again quite common. In general synonymity is not a problem. However, in relational databases synonymity is not allowed, essentially for efficiency reasons. Databases are developed based on the *unique name assumption*, that is, in databases, different strings and assertions always mean different things.

Example 3.4 (An interpretation function providing an interpretation of the assertional theory describing Figure 3.1) . The natural interpretation function which interprets the sentences in Example 3.3 (left) to the facts in Example 3.2 (right) is

$$
\begin{aligned}
&\mathcal{I}_A(\text{Sofia è una persona}) &&= \texttt{Sofia is a person}\\
&\mathcal{I}_A(\text{Paolo è un uomo}) &&= \texttt{Paolo is a man}\\
&\mathcal{I}_A(\text{Rocky is a dog}) &&= \texttt{Rocky is a dog}\\
&\mathcal{I}_A(\text{Sofia is near Paolo}) &&= \texttt{Sofia is near Paolo}\\
&\mathcal{I}_A(\text{Rocky è il cane di Sofia}) &&= \texttt{Rocky is the dog of Sofia}\\
&\mathcal{I}_A(\text{Sofia è un'amica di Paolo}) &&= \texttt{Sofia is a friend of Paolo}\\
&\mathcal{I}_A(\text{Rocky è un animale}) &&= \texttt{Rocky is an animal}\\
&\mathcal{I}_A(\text{Sofia ha i capelli biondi} &&= \texttt{Sofia has blond hair}\\
&\ldots
\end{aligned}
$$

Observation 3.10 (Assertions and facts, subjectivity) The problem of the subjectivity of representations remains, this being an unavoidable fact of life. However the notions of fact, assertion and interpretation function give leverage. First, facts are assumed to be unequivocally described, via interpretation functions, by assertions where, in turn, are linguistic representations and, as such, can be shared. Second, assertions, though subjectively selected by humans, are assumed to be atomic, that is, to provide the minimal possible level of details at which a model can be described.

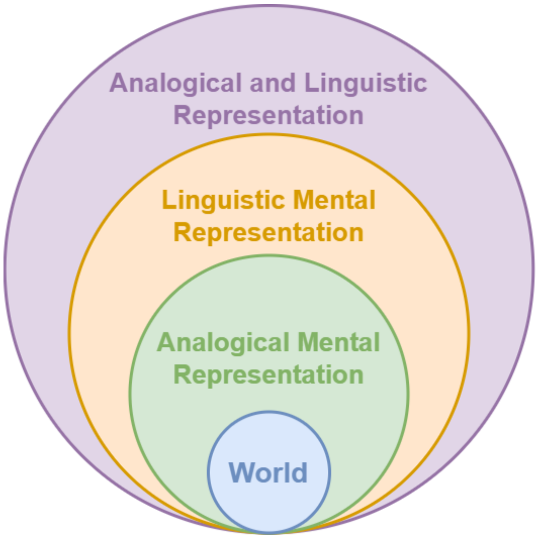

Analogical and Linguistic Representation
Linguistic Mental Representation
Analogical Mental Representation
World

Fig. 3.2 Diagram of Representations

Chapter 4
Formal models and assertional theories

In order to avoid fallacious reasoning we need to represent models and assertional theories in a unambiguous, that is formal, way. Four are the features of of interest to us:

- **Formality**: It should be a logical language, that is, with well defined syntax and semantics;
- **Universality**: it should be able to represent all types of facts;
- **Intuitiveness**: it should allow for assertions whose basic elements (entity names, concepts and properties) as well as their structure (that is how the basic elements are connected together to build assertions) should be, on one side, intuitive to people while, on the other side, have a direct map to the structure and organization of the reference domain;
- **Computational efficiency**: $\mathcal{L}_A$ should allow for a fast and efficient inference engine, exploiting the inherent efficiency of the data structures used to memorize the world model.

Observation 4.1 (Types of assertional languages) ER and UML models are intuitive but not universal, they represent only knowledge facts. DBs are computationally efficient but represent only data facts. Natural language is universal but its semantics are not formally defined and it is not computationally efficient. The latter weakness extends to well defined subsets of natural languages. As an instance of this case, logical languages are universal with well defined syntax and semantics but they are not intuitive to understand and also computationally not efficient (see also Section 6.5).

In Section 4.1, we introduce some basic definitions of set theory, useful in order to define $\mathtt{D}$, while, in Section 4.2, we introduce some basic definitions of graph theory useful in order to define $\mathcal{L}_A$.

4.1 Set theory

4.1.1 Basic definitions

We can define sets in two ways

- **Listing:** The set is described by listing all its elements (for instance, $A = \{a, e, i, o, u\}$).
- **Abstraction:** The set is described through a property of its elements (for instance, $A = \{x|x \text{ is a vowel of the Latin alphabet}\}$).

We have the following basic definitions.

Definition 4.1 (Empty Set) $\emptyset$ is the set containing no elements.

Definition 4.2 (Membership) $\boldsymbol{a} \in \boldsymbol{A}$, element $\boldsymbol{a}$ belongs to the set $\boldsymbol{A}$.

Definition 4.3 (Non-membership) $\boldsymbol{a} \notin \boldsymbol{A}$, element $\boldsymbol{a}$ doesn't belong to the set $\boldsymbol{A}$.

Definition 4.4 (Equality) $\boldsymbol{A} = \boldsymbol{B}$, if and only if $\boldsymbol{A}$ and $\boldsymbol{B}$ contain the same elements.

Definition 4.5 (Inequality) $\boldsymbol{A} \neq \boldsymbol{B}$, if and only if it is not true that $\boldsymbol{A} = \boldsymbol{B}$.

Definition 4.6 (Subset) $\boldsymbol{A} \subseteq \boldsymbol{B}$, if and only if all elements in $\boldsymbol{A}$ also belong to $\boldsymbol{B}$.

Definition 4.7 (Proper Subset) $\boldsymbol{A} \subset \boldsymbol{B}$, if and only if $\boldsymbol{A} \subseteq \boldsymbol{B}$ and $\boldsymbol{A} \neq \boldsymbol{B}$.

Definition 4.8 (Universal Set) The universal set is the set of all elements or members of all related sets and is denoted by the letter $\boldsymbol{\mathcal{U}}$.

We use **Venn diagrams** to represent sets. Venn diagrams consist of overlapping or intersecting circles representing sets and their relationships. Each circle represents a specific set, and the area where the circles overlap represents the elements shared between the corresponding sets. An element that does not belong to a set is represented as a dot outside the circle representing the set.

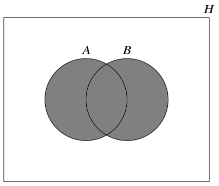


Fig. 4.1 Union set operation

Definition 4.9 (Union) Given two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, the union of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as the set containing the elements belonging to $\boldsymbol{A}$ or to $\boldsymbol{B}$ or to both, and is denoted with $\boldsymbol{A} \cup \boldsymbol{B}$.

Definition 4.10 (Intersection) Given two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, the intersection of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as the set containing the elements that belong both to $\boldsymbol{A}$ and $\boldsymbol{B}$, and is denoted with $\boldsymbol{A \cap B}$.

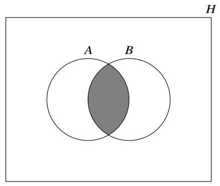


Fig. 4.2 Intersection set operation

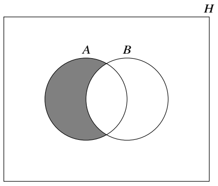


Fig. 4.3 Difference set operation

Definition 4.11 (Difference) Given two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, the difference of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as the set containing all the elements which are members of $\boldsymbol{A}$, but not members of $\boldsymbol{B}$, and is denoted with $\boldsymbol{A \setminus B}$.

Definition 4.12 (Complement) Given a universal set $\boldsymbol{U}$ and a set $\boldsymbol{A}$, where $\boldsymbol{A \subseteq U}$, the complement of $\boldsymbol{A}$ in $\boldsymbol{U}$ is defined as the set containing all the elements in $\boldsymbol{U}$ not belonging to $\boldsymbol{A}$, and is denoted with $\boldsymbol{A^c}$ or $\boldsymbol{U \setminus A}$.

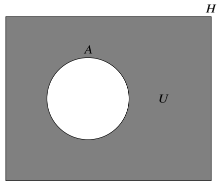


Fig. 4.4 Complement set operation

Theorem 4.1 (Properties of Operations)

- ***With same set***
  - $A \cap A = A$
  - $A \cup A = A$
- ***Commutative***
  - $A \cap B = B \cap A$
  - $A \cup B = B \cup A$
- ***Empty set***
  - $A \cap \emptyset = \emptyset$
  - $A \cup \emptyset = A$
- ***Associative***
  - $(A \cap B) \cap C = A \cap (B \cap C)$
  - $(A \cup B) \cup = A \cup (B \cup C)$
- ***Distributive***
  - $A \cap (B \cup C) = (A \cap B) \cup (A \cap C);$

  - $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

- ***De Morgan laws***
  - $\overline{A \cup B} = \overline{A} \cap \overline{B}$
  - $\overline{A \cap B} = \overline{A} \cup \overline{B}$

4.1.2 Relations

Definition 4.13 (Cartesian product) Given two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, the Cartesian product of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as the set of ordered couples $\boldsymbol{(a, b)}$ where $\boldsymbol{a \in A}$ and $\boldsymbol{b \in B}$, formally:

$$A \times B = \{(a, b) : a \in A \text{ and } b \in B\}$$

Example 4.1 (Cartesian product) Given $A = \{1, 2, 3\}$ and $B = \{a, b\}$, then

$$A \times B = \{(1, a), (1, b), (2, a), (2, b), (3, a), (3, b)\}$$
$$and$$
$$B \times A = \{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$$

Definition 4.14 (Relation) A relation R from the set A to the set B is a subset of the Cartesian product of A and B: $R \subseteq A \times B$.
If $(x, y) \in R$, then we will write xRy and we say 'x is R-related to y'.

Proposition 4.1 *A binary relation on a set A is a subset $R \subseteq A \times A$.*

Given a relation R from A to B:

- the **domain** of R is the set $Dom(R) = \{a \in A|$there exists a $b \in B, aRb\}$
- the **co-domain** of R is the set $Cod(R) = \{b \in B|$there exists an $a \in A, aRb\}$

Example 4.2 Given $A = \{1, 2, 3, 4\}$, $B = \{a, b, d, e, r, t\}$ and aRb iff in the Italian name of a there is the letter b, then $R = \{(2, d), (2, e), (3, e), (3, r), (3, t), (4, a), (4, r), (4, t)\}$

Example 4.3 Given $A = \{3, 5, 7\}$, $B = \{2, 4, 6, 8, 10, 12\}$ and aRb iff a iff a is a divisor of b, then $R = \{(3, 6), (3, 12), (5, 10)\}$

Definition 4.15 (Inverse relation) Let R be a relation from A to B. The inverse relation of R is the relation $R^{-1} \subseteq B \times A$ where

$$R^{-1} = \{(b, a)|(a, b) \in R\}$$

Definition 4.16 (Relation Properties) Let R be a binary relation A.R is:

- **reflexive** iff aRa for all $a \in A$
- **symmetric** iff aRb implies bRa for all $a, b \in A$
- **transitive** iff aRb and bRc imply aRc for all $a, b, c \in A$

- **anti-symmetric** iff aRb and bRa imply $a = b$ for all $a, b \in A$

Definition 4.17 (Equivalence relation) Let R be a binary relation on a set A. R is an equivalence relation iff it satisfies all the following properties:

- reflexive
- symmetric
- transitive

Remark 4.1 An equivalence relation is usually denoted with ~ or ≡

Definition 4.18 (Set partition) Let A be a set, a partition of A is a family F of non-empty subsets of A so that:

- the subsets are pairwise disjoint
- the union of all subsets is the set A

Remark 4.2 Each element of A belongs to exactly one subset in F

Definition 4.19 (Equivalence class) Let A be a set and ≡ an equivalence relation on A, given an x ∈ A we define equivalence class X the set of elements x' ∈ A s.t. x' ≡ x, formally:

$$X = \{x' | x' \equiv x\}$$

Remark 4.3 Any element x is sufficient to obtain the equivalence class X, which is denoted also with [x].

x ≡ x' implies [x]=[x']=X

Definition 4.20 (Quotient set) We define quotient set of A with respect to an equivalence relation ≡ as the set of equivalence classes defined by ≡ on A, and denote it with A / ≡.

Theorem 4.2 *Given an equivalence relation ≡ on A, the equivalence classes defined by ≡ on A are a partition of A. Similarly, given a partition on A, the relation R defined as xRx' iff x and x' belong to the same subset, is an equivalence relation on A.*

Example 4.4 (Parallelism relation) Two straight lines in a plane are parallel if they do not have any point in common or if they coincide.
The parallelism relation || is an equivalence relation since it is:

- reflexive: $r||r$
- symmetric: $r||s$ implies $s||r$
- transitive $r||s$ and $s||t$ imply $r||t$

We can thus obtain a partition in equivalence classes: intuitively, each class represent a direction in the plane.

Order relation:

Definition 4.21 (Order) Let A be a set and R be a binary relation on A.
R is an order (**partial**), usually denoted with $\leq$, if it satisfies the following properties:

- reflexive $a \leq a$
- anti-symmetric $a \leq b$ and $b \leq a$ imply $a = b$
- transitive $a \leq b$ and $b \leq c$ imply $a \leq c$

If the relation holds for all $a, b \in A$ then it is a **total order**.

A relation is a **strict order**, denoted with "<", if it satisfies the following properties:

- transitive $a < b$ and $b < c$ imply $a < c$
- for all $a, b \in A$ either $a < b$ or $b < a$ or $a = b$

4.1.3 Functions

Definition 4.22 (Functions) Given two sets A and B, a function f from A to B is a relation that associates to each element a in A exactly one element b in B. Denoted with:

$$f : A \longrightarrow B$$

The domain of f is the whole set A.
The image of each element a in A is the element b in B s.t. $b = f(a)$.
The co-domain of f (or image of f) is a subset of B defined as follows:

$$Im_f = \{b \in B | \text{ there exists an } a \in A \text{ s.t. } b = f(a)\}$$

Remark 4.4 It can be the case that the same element in B is the image of several elements in A.

Classes of functions:

Definition 4.23 (Surjective function) A function $f : A \longrightarrow B$ is surjective if each element in B is image of some elements in A :

$$\text{for each } b \in B \text{ there exists an } a \in A \text{ s.t. } f(a) = b$$

Definition 4.24 (Injective function) A function $f : A \longrightarrow B$ is injective if distinct elements in A have distinct images in B :

$$\text{for each } b \in Im_f \text{ there exists a unique } a \in A \text{ s.t. } f(a) = b$$

Definition 4.25 (Bijective function) A function $f : A \longrightarrow B$ is bijective if it is injective and surjective:

$$\text{for each } b \in B \text{ there exists a unique } a \in A \text{ s.t. } f(a) = b$$

Definition 4.26 (Inverse function) If $f : A \longrightarrow B$ is bijective we can define its inverse function:

$$f^{-1} : B \longrightarrow A$$

Remark 4.5 For each function f there is a inverse relation. This relation is a function iff f is bijective.

Example 4.5 (Inverse function) Example of two different inverse functions:

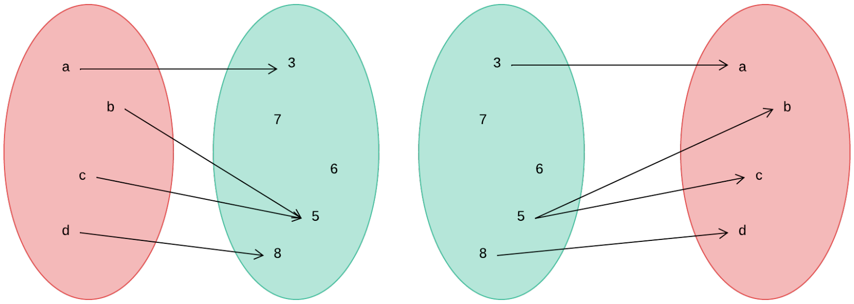


Fig. 4.5 Inverse of not bijective function

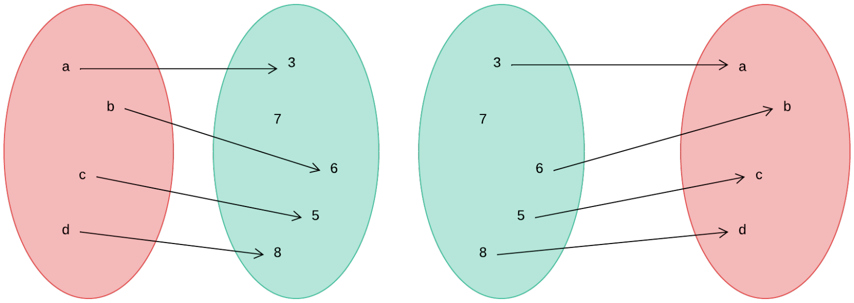


Fig. 4.6 Inverse of bijective function

Definition 4.27 (Composite function) Let $f : A \longrightarrow B$ and $g : B \longrightarrow C$ be functions. The composition of f and g is the function $g \circ f : A \longrightarrow C$ obtained by applying f and then g:

- $(g \circ f)(a) = g(f(a))$ for each $a \in A$
- $g \circ f = \{(a, g(f(a))|a \in A)\}$

4.2 Graph theory

4.2.1 Basic Notions

Definition 4.28 (Graph) A **graph** $\boldsymbol{G}$ is an ordered pair $\boldsymbol{G =< V, E >}$, where $\boldsymbol{V}$ is the set of **vertices** (or **nodes**) and $\boldsymbol{E}$ is the set of **edges** (or **links**). Edges are pairs of vertices.

Definition 4.29 (Order) The **order** of a graph is the number of vertices of the graph.

Definition 4.30 (Size) The **size** of a graph is the number of edges in the graph.

Definition 4.31 (Degree) The **degree** of a vertex is the number of edges incident on that vertex.

Definition 4.32 (Directed graph) A **directed graph** is a graph where edges are ordered pairs of distinct vertices $\boldsymbol{(x, y)}$. $\boldsymbol{x}$ and $\boldsymbol{y}$ are called the **end points**, where $\boldsymbol{x}$ is the **tail** and $\boldsymbol{y}$ is the **head**.

From now on we concentrate on directed graphs.

Definition 4.33 (Leaf, intermediate node) In a directed graph, a **leaf** is a node with no outgoing nodes. A node which is not a leaf is an **intermediate node**.

Definition 4.34 (Path) A **path**, also called a **linear graph**, is a graph where the vertices can be ordered in a sequence $\boldsymbol{v_1, v_2, \ldots, v_n}$, where the edges correspond to the pairs of consecutive vertices $\boldsymbol{\{v_i, v_{i+1}\}}$ for $\boldsymbol{i = 1, 2, \ldots, n-1}$.

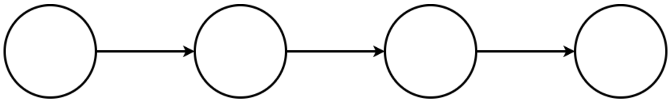


Fig. 4.7 Path Graph

Definition 4.35 (Cycle, cyclic graph) A **cycle**, also called a **circular graph**, is a path in which only the first and last vertices are equal. A **cyclic graph** is a graph which contains a cycle.

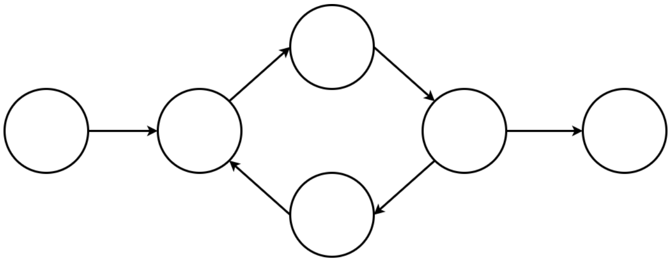


Fig. 4.8 Cyclic Graph

Definition 4.36 (Tree, rooted tree, root, leaf, intermediate nodes) A **tree** is an undirected graph in which any two vertices are connected by exactly one path. A **polytree**, or **or directed tree**, or **oriented tree**, is a directed acyclic graph whose underlying undirected graph is a tree. A **rooted tree** is a tree in which one vertex has been designated the root. A **root** is a node with no incoming nodes.

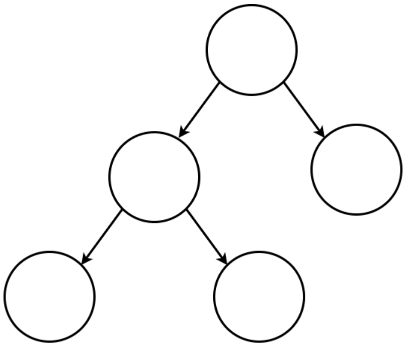


Fig. 4.9 Tree Graph

Definition 4.37 (Forest, polyforest, directed forest, oriented forest) A **forest** is an undirected graph. A **polyforest**, or **directed forest**, or **oriented forest**, is a directed acyclic graph whose underlying undirected graph is a forest.

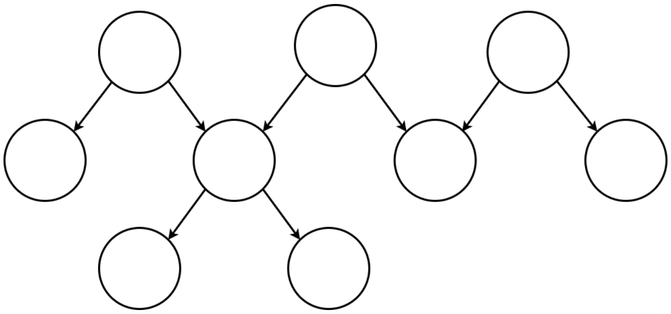


Fig. 4.10 Forest Graph

Definition 4.38 (Directed acyclic graph (DAG)) A **directed acyclic graph** (**DAG**) is a directed graph that does not contain any cycles.

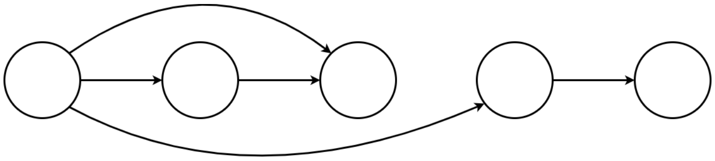


Fig. 4.11 Directed Acyclic Graph

4.2.2 Labeled Graphs

From now on we concentrate on labeled directed graphs.

Definition 4.39 (Labeled Graph) A labeled graph is a type of graph where each vertex and edge is assigned a label.

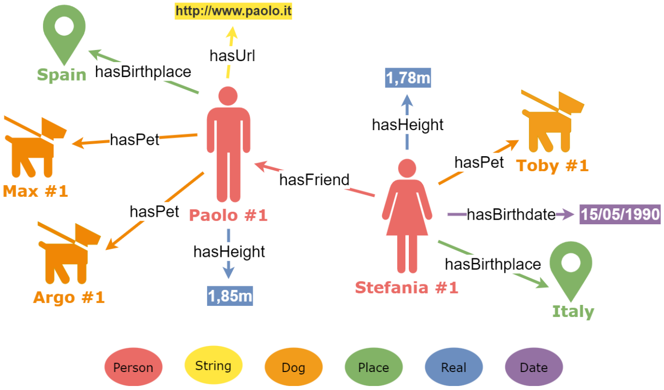


Fig. 4.12 Labeled Graph

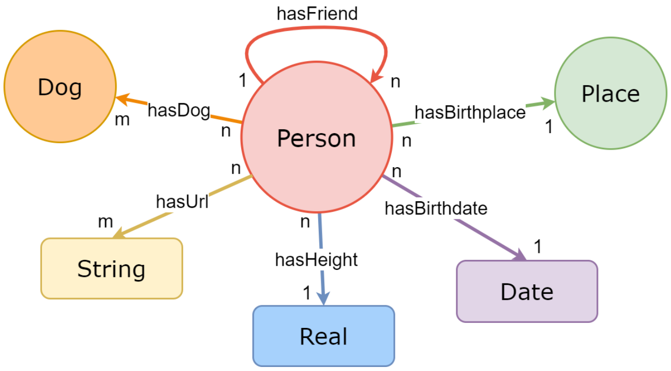


Fig. 4.13 Labeled Graph

4.3 Exercises

Exercise 4.1 (Linguistic and analogical mental representations) Create an analogical representation using set theory for this linguistic mental representation.

- In(tree, lab)
- In (monkey1, lab)
- In(monkey2, lab)
- Eating(monkey1, banana)
- SittingOn(monkey2, tree)*
- Scratching(monkey2, hisHead)*

Exercise 4.2 (Linguistic and analogical mental representations) Create an analogical representation using set theory for this linguistic mental representation. This time improve it using labels.

- In(tree, lab)
- In (monkey1, lab)
- In(monkey2, lab)
- Eating(monkey1, banana)
- SittingOn(monkey2, tree)*
- Scratching(monkey2, hisHead)*

Exercise 4.3 (Linguistic and analogical mental representations) Create an analogical representation using knowledge graphs for this linguistic mental representation.

- In(tree, lab)
- In (monkey1, lab)
- In(monkey2, lab)
- Eating(monkey1, banana)
- SittingOn(monkey2, tree)*
- Scratching(monkey2, hisHead)*

Exercise 4.4 (Linguistic and analogical mental representations) Create an analogical representation using knowledge graphs for this linguistic mental representation. This time improve it using labels.

- In(tree, lab)
- In (monkey1, lab)
- In(monkey2, lab)
- Eating(monkey1, banana)
- SittingOn(monkey2, tree)*
- Scratching(monkey2, hisHead)*

Part II
World models and logics

Chapter 5
World Model - extensional representation

Assertional theories and models are an important step ahead but there are still three main limitations:

- We are considering only the facts of the model in focus. What about the facts which can occur in the other possible models, describing possibly very different, situations?
- The language being used is very limited and consists only of the set of assertions which describe the facts of the model in focus. What about the assertions describing facts in all the the other models?
- As a consequence of the two facts above, the definition of Interpretation function is not general. A new and different interpretation function must defined for any new situation.

5.1 Domain

As from Equation (3.1) we have that a model $\mathtt{M}$ is defined as $\mathtt{M} = \{\mathtt{f}\}$, where $\{\mathtt{f}\}$ is a set of facts which are the case in certain situation.

Definition 5.1 (Domain (of interpretation)) A **Domain (of interpretation)** is a set of facts $\{\mathtt{f}\}$.

$$\mathtt{D} = \{\mathtt{f}\} \tag{5.1}$$

Definition 5.2 (Model) Given a domain $\mathtt{D}$, a **model** $\mathtt{M}$ is a subset of $\mathtt{D}$.

$$\mathtt{M} = \{\mathtt{f}\} \subseteq \mathtt{D} \tag{5.2}$$

Observation 5.1 (Domain, model) A domain is the set of all facts that we are willing to consider. A model is just the subset of fact that we define as depicting what is the case in the current situation. Definition 5.2 generalizes in the obvious way Definition 3.1

Example 5.1 (The facts of the domain represented in Figure 3.1) A domain collecting among others the facts represented in Figure 3.1 could for instance contain the following facts:

```
Sofia is a person          Sofia is a woman
Paolo is a man             Paolo is a person
Paolo is a dog             Rocky is a dog
Sofia is near Paolo        Rocky is the dog of Sofia
Rocky is the dog of Paolo ...
```

. . . and many more.

Observation 5.2 (Domain) While a model is the set of facts which are the case in a certain situation, a domain consists of the set of facts which are *potentially* the case for all possible situations. A Domain defines all and only that can be potentially perceived.

Observation 5.3 (Mutually inconsistent facts in a domain) As from Example 5.1 a domain, differently from a model, can contain facts which, at least intuitively are mutually inconsistent. Given a domain, there are many potential models, some of which are potentially mutually inconsistent. Domains must allow for the possible instantiation of distinct mutually inconsistent models, as it is normally the case in the world.

5.2 Assertional language

As from Equation (3.2) we have that an assertional Theory $\mathcal{T}_{\mathcal{A}}$ is defined as $\mathcal{T}_{\mathcal{A}} = \{a\}$, where $\{a\}$ is a set of assertions which describe the facts which are the case in certain situation.

Definition 5.3 (Assertional language) An **assertional language** $\mathcal{L}_A$ is a set of assertions $\{a\}$

$$\mathcal{L}_A = \{a\} \tag{5.3}$$

Definition 5.4 (Assertional theory) Given an assertional language $\mathcal{L}_A$, an **assertional theory** $\mathcal{T}_{\mathcal{A}}$ is a subset of $\mathcal{L}_A$.

$$\mathcal{T}_{\mathcal{A}} = \{a\} \subseteq \mathcal{L}_A \tag{5.4}$$

Observation 5.4 (Assertional language) While an assertional theory is the set of assertions which describes what is the case in a certain model, an assertional language consists of the set of assertions which describe all the facts that can *potentially* occur. Definition 5.4 extends in the obvious way Definition (3.2).

Example 5.2 (Assertional language) The language from Example 3.3 extended to name all the facts of the domain defined in Example 5.1 in the same way (i.e. quoted Italian translations of the English sentence describing a fact) is an assertional language.

Observation 5.5 (Completeness and correctness of an assertional language $\mathcal{L}_A$ with respect to a domain $\mathtt{D}$) An assertional language is not necessarily complete, that is, it does not necessarily contain assertions for all the facts in a domain (which, among other things, are in principle infinite). The key feature is that is should contain *all* the assertions deemed relevant. Vice versa an assertional language is requested to be correct, that is to contain *only* assertions which denote facts in the reference domain. This in order to avoid nonsensical assertions.

Example 5.3 (Assertional languages)

1. Languages which allow only for assertions in natural language of the form "`<subject> <verb> <object>`" describe facts about the world. The language used is a sequence of simple assertions without complex phrases.
2. Relational databases (DBs) describe facts about the world. The language used to describe the contents of a relational DB are tables;
3. Entity-relationship (ER) models describe general facts about the contents of databases. The are written using the ER diagram language, a specific labelled graph language;

5.3 Interpretation function

Definition 5.5 (Interpretation function) Let $\mathcal{L}_A$ be a language of assertions and $\mathtt{D}$ be a domain. Then an **Interpretation Function** of an assertional language $\mathcal{I}_A$ is defined as

$$\mathcal{I}_A : \mathcal{L}_A \to \mathtt{D} \quad (\mathcal{I}_A \subseteq \mathcal{L}_A \times \mathtt{D}) \tag{5.5}$$

We say that a fact $\mathtt{f} \in \mathtt{M}$ is the **interpretation** of $a \in \mathcal{I}_A$, and write

$$\mathtt{f} = \mathcal{I}_A(a) = a^{\mathcal{I}_A} \tag{5.6}$$

to mean that a is a linguistic description of $\mathtt{f}$. We say that $\mathtt{f}$ is the **interpretation of** a, or, equivalently, that a **denotes** $\mathtt{f}$.

Observation 5.6 (Interpretation function) Definition 5.5 generalizes as needed and substitutes Definition 3.3.

Observation 5.7 (Interpretation function, non-ambiguity and synonymity) As from Observation 3.7, 3.8, 3.9, interpretation functions, being functions, are not ambiguous thus not allowing for polysemous assertions and words while allowing for synonymity.

Observation 5.8 (Interpretation function, totality) Interpretation functions are total. This guarantees that any element of the language has an interpretation.

Observation 5.9 (Interpretation function, non-surjectivity) Interpretation functions are not necessarily surjective. In other words, if $\mathcal{I}_A : \mathcal{L}_A \rightarrow \mathtt{D}$, $\mathcal{L}_A$ may not be able to name all the facts in $\mathtt{D}$. This property is useful with infinite domains or when one is not interested in mentioning all the known facts. See also Observation 5.5.

Example 5.4 (Interpretation function) A from Example 5.2, take $\mathcal{L}_A$ to contain the assertions in Example 3.3, extended to describe all the facts in the domain $\mathtt{D}$ defined in Example 5.1. Take the domain $\mathtt{D}$ defined in Example 5.1. Then, an interpretation function $\mathcal{I}_A : \mathcal{L}_A \rightarrow \mathtt{D}$ can be constructed by extending in the obvious way the interpretation function defined in Example 3.4.

5.4 World model

The journey is complete. We have only to pull everything together.

Observation 5.10 (The roles of $\mathtt{D}$, $\mathcal{L}$, $\mathcal{I}_A$, $\mathtt{M}$, $\mathcal{T}_A$) The definitions provided in the previous sections can be summarized by the following figure.

$$\begin{tikzcd}
a \arrow[d, "\mathcal{I}_A"] & \mathcal{T}_A \arrow[l, "\in"'] \arrow[r, "\subseteq"] \arrow[d, dashed, "\mathcal{I}_A"] & \mathcal{L}_A \arrow[d, "\mathcal{I}_A"] \\
\mathtt{f} & \mathtt{M} \arrow[l, "\in"'] \arrow[r, "\subseteq"] & \mathtt{D}
\end{tikzcd} \tag{5.7}$$

In Equation (5.7), $\mathtt{D}$ defines the set of facts $\mathtt{f}$ of potential interest, $\mathtt{M}$ the set of facts we are focusing on, $\mathcal{L}_A$ the set of assertions a of potential interest, and, finally $\mathcal{T}_A$ is the theory describing $\mathtt{M}$. In other words, how single assertions a describe single facts $\mathtt{f}$ on one side, and how the overall sets of assertions $\mathcal{L}_A$ describe $\mathtt{D}$ on the other side, define the scope within which an assertional theory $\mathcal{T}_A$ can focus to describe specific models $\mathtt{M}$. Assertions, languages and theories are just the means for specifying the intended model stating the intended facts among this allowed by the reference domain.

Definition 5.6 (World Model)

$$\hat{\mathcal{W}} = \langle \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A \rangle \tag{5.8}$$

is a **world model**.

Observation 5.11 (World model) In Equation (5.7) the components of a world model, i.e., $\mathtt{D}$, $\mathcal{L}_A$, $\mathcal{I}_A$, define the general rules which are followed when building a representation. They are defined a priori, usually by experts in modeling and knowledge representation as general tools to be used by practitioners. They provide the general modeling infrastructure which allows to represent real world problems. They also provide a uniform framework under which any two representations can be compared and possibly even merged. Software practitioners usually study these models during some CS or AI classes and they use them *as is* when developing systems; think for instance of the large usage of ER models.

Observation 5.12 (From mental representations to world models) World models are spaces of possible representations, i.e., theories, designed to minimize the possibility of different mental representations of the same theory and corresponding model depicting the world.

World models provide the general framework within which assertional theories and models can be defined and compared.

Observation 5.13 (Theory and Model) Remember that, given a world model $\hat{\mathcal{W}} = \langle \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A \rangle$, we have that $\mathtt{M} = \{\mathtt{f}\} \subseteq \mathtt{D}$ (Equation (5.1)) and $\mathcal{T}_\mathcal{A} = \{a\} \subseteq \mathcal{L}_A$ (Equation (5.3))

Observation 5.14 (Defining a model via a theory) The most common way to model the world is by defining a set of assertions, what we call a theory. In other words, we construct a model $\mathtt{M}$ by selecting any subset $\mathcal{T}_\mathcal{A}$ of $\mathcal{L}_A$. This is the common approach when the task is that of representing from scratch a given part of the world which is of interest.

However, sometimes, one is given a predefined theory $\mathcal{T}_\mathcal{A}$ and a predefined model $\mathtt{M}$ and is asked how they relate. In which case we have the following.

Definition 5.7 (Correctness and completeness of an assertional theory $\mathcal{T}_\mathcal{A}$ with respect to a model $\mathtt{M}$) Let $\hat{\mathcal{W}} = \langle \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A \rangle$ be a world model. Let $\mathcal{T}_\mathcal{A} \subseteq \mathcal{L}_A$ and $\mathtt{M} \subseteq \mathtt{D}$ be an assertional theory and a model, respectively. Then we have two possible situations, as follows

- **Correctness**. Let $a \in \mathcal{L}_A$ be an assertion. If for all a, if $a \in \mathcal{T}_\mathcal{A}$ then $\mathcal{I}_A(a) = \mathtt{f} \in \mathtt{M}$ we say that $\mathcal{T}_\mathcal{A}$ is **correct** with respect to $\mathtt{M}$, or that $\mathtt{M}$ **is a model** for AL;
- **Completeness**. Let $\mathtt{f} \in \mathtt{M}$ be a fact. If, for all $\mathtt{f}$, if $\mathtt{f} \in \mathtt{M}$ then there is an assertion $a \in \mathcal{T}_\mathcal{A}$ such that $\mathcal{I}_A(a) = \mathtt{f}$ we say that $\mathcal{T}_\mathcal{A}$ is **complete** with respect to $\mathtt{M}$

The notions of **incorrectness** and **incompleteness** are defined in the obvious way

Observation 5.15 (Correctness and completeness of an assertional theory $\mathcal{T}_\mathcal{A}$ with respect to a model $\mathtt{M}$) An assertional theory may not be complete, namely there can be facts of the domain for which it does contain assertions. Incomplete descriptions of models are the default, because of ignorance or also because of missing interest. However a theory $\mathcal{T}_\mathcal{A}$ must contain *only* assertions about facts in $\mathtt{M}$ for $\mathtt{M}$ of $\mathcal{T}_\mathcal{A}$. This in order to avoid faulty information. Notice how this requirement is opposite to that on languages and domains as from Observation 5.5.

Example 5.5 (Correctness and completeness of an assertional theory $\mathcal{T}_\mathcal{A}$ with respect to a model $\mathtt{M}$*)* Consider Example 3.4. An assertional theory $\mathcal{T}_\mathcal{A}$ containing all and only the assertions domain of the interpretation function in Example 3.4 is correct and complete with respect to the model $\mathtt{M}$ containing all and only the facts which are in the domain of the same interpretation function. Any assertional theory which is a subset of $\mathcal{T}_\mathcal{A}$ is correct with respect to $\mathtt{M}$. $\mathtt{M}$ is not a model of any superset of $\mathcal{T}_\mathcal{A}$.

Chapter 6
World model - intensional representation

World models, allow us to exploit facts and assertions about them as the main components needed to build a representation of the world, to be later used for solving problems.

Observation 6.1 (World model, extensional representation) World models, as from Definition 5.6, are **extensional representations** of the world, namely they are defined as sets of assertions a and facts $\mathtt{f}$, plus an interpretation function $\mathcal{I}_A$ which allows to define which assertions denote which facts in one or more reference models.

But what is fact? How do we construct assertions about facts? The answer to this question requires defining an **intensional representation** of world models, namely the **representation mechanisms** which allow to construct assertions and facts starting from a finite set of primitive component elements. In the following, whenever needed in order to avoid confusion we adopt the following notation and terminology.

Notation 6.1 (Extensional and intensional representation of a set) Let $\mathcal{S}$ be a set. Then by $\mathcal{S}^e$ we mean the **extensional representation** of $\mathcal{S}$, i.e., as a set of elements (e.g., facts, assertions, but not only); by $\mathcal{S}^i$ we mean the **intensional representation** of $\mathcal{S}$, where the elements of $\mathcal{S}^e$ are defined intensionally, starting from a set of primitive components. The superscripts are dropped when no confusion arises.

6.1 Domain

Example 3.1, Observation 3.1 and also Observation 3.5 provide indications about how to construct an intensional representation of facts.

Intuition 6.1 (Domain, intensional representation) The intensional representation of a domain is composed of three components, as follows

- *entities*, associated with those elements of the representation which can be isolated and distinguished from the rest;
- *classes* (sets) of entities, characterized by the fact they have some common characteristics which is not shared by the entities of the other sets;
- *relations* among entities, which collect multiple entities sharing a common property.

Definition 6.1 (Domain, intensional representation) The **intensional representation $\mathtt{D}^i$ of a domain** $\mathtt{D}$ is defined as

$$\mathtt{D}^i =< \mathtt{E}, \{\mathtt{C}\}, \{\mathtt{R}\} > \tag{6.1}$$

with

$$\begin{aligned} &\mathtt{E} = \{\mathtt{e}\} \\ &\mathtt{C} \subseteq \mathtt{E} \\ &\mathtt{R} \subseteq \underbrace{\mathtt{E} \times \cdots \times \mathtt{E}}\ \mathtt{n}\text{ times} \end{aligned}$$

where $\mathtt{E} = \{\mathtt{e}\}$ is a set of **entities**, $\{\mathtt{C}\}$ is a set of **classes** of entities, $\{\mathtt{R}\}$ is a set of n-ary **relations** $\mathtt{R}^n$, for some n. $\mathtt{E}$ is called the **universe** of $\mathtt{D}^i$ or also the **universe of interpretation**.

Definition 6.2 (Fact, intensional representation) The **intensional representation of a fact** $\mathtt{f}$ has one of the following four forms

$$\begin{array}{ccc} \mathtt{e} & \in & \mathtt{C} \\ < \mathtt{e}_1, ..., \mathtt{e}_n > & \in & \mathtt{R} \\ \mathtt{C} & \subseteq & \mathtt{E} \\ \mathtt{R}^n & \subseteq & \mathtt{C}_1 \times \cdots \times \mathtt{C}_n \end{array} \tag{6.2}$$

with $\mathtt{e}, \mathtt{e}_i \in \mathtt{E}$ and $\mathtt{C}, \mathtt{C}_i \subseteq \mathtt{E}$.

Observation 6.2 (Fact, intensional representation) The intuition behind Definition 6.2 goes as follows. $\mathtt{e} \in \mathtt{C}$ means that a certain entity $\mathtt{e}$ belongs to a certain class $\mathtt{C}$, as with the statement that Sofia is a person. $< \mathtt{e}_1, ..., \mathtt{e}_n > \in \mathtt{R}$ means that n entities stand in certain relation, as with the statement that Paolo is in between Sofia and Stefania. $\mathtt{C} \subseteq \mathtt{E}$ means that $\mathtt{C}$, e.g. person, is a class. $\mathtt{R}^n \subseteq \mathtt{C}_1 \times \cdots \times \mathtt{C}_n$ means that a relation applies only to specific classes, as with the statement that persons are friends of animals.

Observation 6.3 (Complexity of facts) One could think of more sophisticated facts, for instance the fact $\mathtt{C}_1 \subseteq \mathtt{C}_2$. This is of course possible, at the price of complicating the assertional language, with an upper limit in the intuitiveness of the resulting world model. Here we are defining the simplest possible world model. Representation languages and logics can anyhow be used to increase the expressivity of the representation language.

Example 6.1 (Fact, intensional and extensional representation) Let us consider Example 5.1. The fact that Sofia is person is constructed by taking the entity Sofia and by asserting that it is a person. The fact that Sofia is near Paolo is constructed by asserting that the entities Sofia and Paolo stand in the relation of one being near the other.

Definition 6.3 (Domain, extensional representation) The **extensional representation $\mathtt{D}^e$ of a domain** $\mathtt{D}$, whose intensional representation $\mathtt{D}^i$ is as from Definition 6.1, is $\mathtt{D}^e = \{\mathtt{f}\}$ with $\mathtt{f}$ as from Definition 6.2.

Observation 6.4 (Entity) But what is an entity? The general idea is that an entity is something which can be perceived and can therefore be represented in an analogical representation of (a part of) the world. Is an entity a specific object? Or a person? Or an animal? Yes, but also anything happening in time, i.e., an event or a process. Entities are assumed to have names. Examples of (names of) entities: Federico, Pussy, Garfield, Trento, the last Football Worldcup, and so on.

Intuition 6.2 (Entity) An **entity** is anything which can be represented in a (analogical or linguistic) representation of the world and which has a name. An entity can be represented in the mental representation of the world any time it is perceived or its name is heard by a human.

Intuition 6.3 (Name) A **name** is a string, written in some language, that allows to refer to entities in representations and, more specifically, in world models.

Intuition 6.4 (Named entity) The vineyard in front of me is not an entity (for me). Even if I can distinctly see it, I don't have a name which I can use to refer to it, for instance when talking with others. Entities, to be entities, must be named entities.

Observation 6.5 (Domain, intensional representation) We intensionally model the world around us in terms of entities. Entities, in turn, are grouped in specific classes, depending on certain properties of theirs. Thus we have, for instance, the classes: human, person, woman, animal dog, car, pen, computer, and so on. Then entities can be put in relations among one another depending on their class. This for instance: humans are friends of humans, or animals, they eat food, a car is near a traffic light. And so on.

Observation 6.6 (Domain, partiality) A domain does not need to contain all five types of facts listed in Definition 6.2. The selection of which facts to include depends on what is being modeled. Roughly speaking domains can be split in two types. The first is data domains, so called because they describe what is perceived, which involve the first two types of facts in Definition 6.2. The second types is knowledge domains, so called because they focus on general statements, not directly perceivable, and describe general relations among classes and relations. Knowledge domains involve the last three types of statements. There are also what we call here mixed domains, namely domains which involve all types of facts. Mixed domains are very useful in communication, e.g., natural language communication among people, as they allow to express uniformly knowledge about entities, classes and relations.

Definition 6.4 (Domain, data, knowledge, mixed) A **data domain** contains only facts of the form $\mathtt{e} \in \mathtt{C}$ and $< \mathtt{e}_1, ..., \mathtt{e}_n >\in \mathtt{R}$. A **knowledge domain** contains only facts of the form $\mathtt{C}_1 \subseteq \mathtt{E}$, $\mathtt{R}^n \subseteq \mathtt{C}_1 \times \cdots \times \mathtt{C}_n$. A **mixed domain** contains all types of facts.

Example 6.2 (Data domain) The domain described in Example 5.1 is a data domain. Some of its facts can be intensionally represented as follows:

```
sofia ∈ Person                          <rocky, paolo> ∈ DogOf
sofia ∈ Woman                           paolo ∈ Dog
<paolo, rocky> ∈ HasDog                 rocky ∈ Dog
<sofia, paolo> ∈ Near                   <rocky, sofia> ∈ DogOf
paolo ∈ Man                             <sofia, paolo> ∈ FriendOf
<paolo, sofia, stefania> ∈ Between
```

Notationally, entities are written with first letters lower case while in Concepts and Properties it is upper case.

Example 6.3 (Knowledge domain) A knowledge domain describing the knowledge underlying the data domain in Example 5.1 is a data domain could for instance contain the following facts.

```
Person ⊆ Entity            HasDog ⊆ Person × Dog
Dog ⊆ Entity               DogOf ⊆ Dog × Person
Animal ⊆ Entity            FriendOf2 ⊆ person × person × person
Near ⊆ Entity × Entity     FriendOf1 ⊆ Person × Person
FatherOf ⊆ Person × Person ChildOf ⊆ Person × Person
```

where `Entity` stands for `E`.

Example 6.4 (Mixed domain) An example of mixed domain cane be built by unioning the domains in Example 6.2 and 6.3.

6.2 Assertional language

Definition 6.5 (Assertional language, intensional representation) The **intensional representation $\mathcal{L}^i_A$ of an assertional language** $\mathcal{L}_A$ is defined as

$$\mathcal{L}^i_A =< \mathcal{E}, \{\mathcal{C}\}, \{\mathcal{P}\} > \tag{6.3}$$

where $\mathcal{E} = \{e\}$ is a set of **(names of) entities**, $\{\mathcal{C}\}$ is a set of **concepts**, where a concept is a **name of a class**, $\{\mathcal{P}\}$ and a set of **properties**, where a property is a **name of a relation**.

Definition 6.6 (Assertional language, extensional representation) The **extensional representation $\mathcal{L}^e_A$ of an assertional language** $\mathcal{L}_A$, whose intensional representation $\mathcal{L}^i_A$ is as from Definition 6.5, is $\mathcal{L}^e_A = \{a\}$ with a having one of the following four forms

$$
\begin{aligned}
&\mathcal{C}(e)\\
&\mathcal{P}^n(e_1,\ldots,e_n)\\
&\mathcal{C}\\
&\mathcal{P}^n(\mathcal{C}_1,\ldots,\mathcal{C}_n)
\end{aligned}
\tag{6.4}
$$

where: $\mathcal{C}(e)$ should be read as the entity of name e belonging to the class of name $\mathcal{C}$, $\mathcal{P}(e_1,\ldots,e_n)$ as the entities of name $e_1,\ldots,e_n$ being involved in the relation of name $\mathcal{P}$.

Observation 6.7 (Abstract and concrete syntax) The assertions in Definition 6.6 must be read as *abstract syntax*, rather than *concrete syntax*. That is, they must be taken as placeholders for many different ways of describing the same fact in different assertional languages. Thus, for instance, the abstract syntax assertion *Person(Sofia)* is the abstract syntax representation of many concrete syntax representations such as, for instance, *Person(Sofia)*, *Sofia is a person*, *P(S)*, *(Sofia)Person* (postfix notation) and, most importantly, various types of graph notations (see below). What gives meaning to assertions is their intended meaning, as defined by the Interpretation function (see below). In concrete, the syntax used in the first two assertions is from the $\mathcal{LOE}$ logic, while that of the last three is from the $\mathcal{LOD}$ logic.

Terminology 6.1 (Alphabet) The intensional representation, in Definition 6.6, $\mathcal{L}^i_A$, of a language, in this case $\mathcal{L}_A$, defined extensionally, in this case $\mathcal{L}^e_A$, is also called the **alphabet** (of that language).

Observation 6.8 (Alphabet) Assertions describe facts. The first step is to associate to each element of the domain a name in the alphabet of the assertional language. Then, assertions are built compositionally following how facts constructed. This creates a one-to-one mapping between facts and assertions which immediately suggest the fact described by an assertion.

Observation 6.9 (Assertional language, partiality) Observation 6.6 extends in the obvious way to assertional languages.

Definition 6.7 (Assertional language, data, knowledge, mixed) An **assertional data language** contains only assertions of the form $\mathcal{C}(e)$ and $\mathcal{P}(e_1,\ldots,e_n)$. An **assertional knowledge language** contains only facts of the form $\mathcal{C}$ and $\mathcal{P}^n(\mathcal{C}_1,\ldots,\mathcal{C}_n)$. An **assertional mixed language** contains both types of assertions.

Example 6.5 (An assertional data language describing the data domain in Example 6.2) Some assertions are as follows:

Person(sofia)	Woman(sofia)	Man(paolo)
Person(paolo)	Dog(paolo)	Dog(rocky)
Near(sofia, paolo)	DogOf(rocky, sofia)	FriendOf(sofia, paolo)
DogOf(rocky, paolo)	HasDog(sofia, rocky)	HasFriend(paolo, sofia)

Example 6.6 (An assertional knowledge language describing the knowledge domain in Example 6.3) Some assertions are as follows:

Person
Dog
Near(Entity, Entity)
HasDog(Entity,Entity)
FriendOf2(Man,Man,man)
HasFriend(Man,Dog)
Woman
Animal
FriendOf1(Person, Person)

where Entity is the concept denoting the set `Entity` of all the entities in the domain.

Example 6.7 (An assertional mixed language describing the mixed domain in Examples 6.2, 6.3) Some assertions are as follows:

Person
FriendOf1(Person, Person)
Dog(rocky)
Near(rocky, paolo)
FriendOf1(sofia, paolo)
Near(Entity,Entity)
Person(paolo)
Dog
Near(paolo, sofia)

Knowledge assertions imply about data assertions. Thus, for instance, one could infer that Sofia is a person from the fact that she is a friend of another person. This takes reasoning, i.e., what logics are for.

Example 6.8 (An assertional knowledge language, using ER models, describing the knowledge domain in Example 6.3) Consider Example 6.6: let us concentrate on the following concepts and roles:

Person
OwnedBy(Animal, Person)
Dog
OwnedBy(Dog, Man)
Animal
Woman
HasDog(Man, Dog)
OwnedBy(Dog, Woman)
HasAnimal(Person, Animal)
Man
HasDog(Women, Dog)

Here is a sample ER Model describing our example:

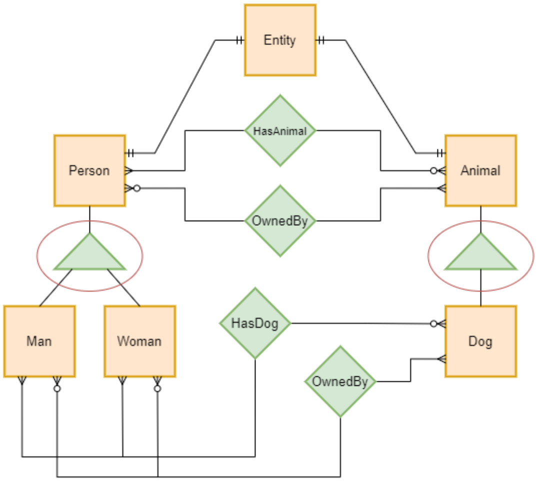
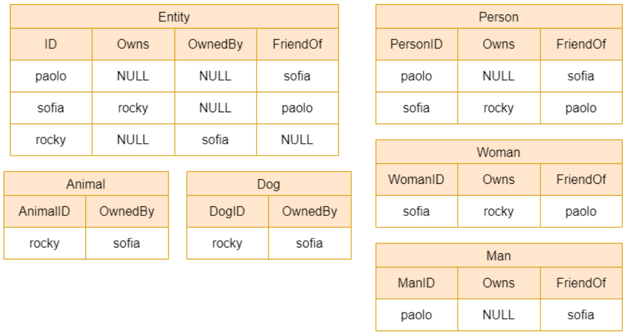


Fig. 6.1 ER Model.

Notation: squares represent concepts, rhombuses represent relationships, triangles represent an IS-A relationship (so for example Man is an instance of a Person). Notice that domains, as we have defined them, do not allow for the IS-A relationship facts (for the moment, see Observation 6.3). This is why this relation must not be considered for ER languages to be assertional languages. As for the arrows the one with two stangs represents a one-to-one relationship, while the fork arrows represent a many-to-many relationship, where those with a white dot identify the optional existence of the relationship.

Example 6.9 (An assertional data language, using relational DBs, describing the data domain in Example 6.2) Considering the assertions of the example 6.1 here is how our data are represented by a database:



Entity

ID	Owns	OwnedBy	FriendOf
paolo	NULL	NULL	sofia
sofia	rocky	NULL	paolo
rocky	NULL	sofia	NULL

Person

PersonID	Owns	FriendOf
paolo	NULL	sofia
sofia	rocky	paolo

Woman

WomanID	Owns	FriendOf
sofia	rocky	paolo

Animal

AnimalID	OwnedBy
rocky	sofia

Dog

DogID	OwnedBy
rocky	sofia

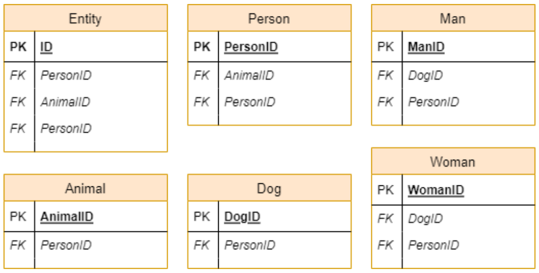
Man

ManID	Owns	FriendOf
paolo	NULL	sofia

Fig. 6.2 Database with Data.

We see how some boxes are instantiated to NULL, this being the case for lack of information.

Example 6.10 (An assertional knowledge language, using relations DB schemas, describing the knowledge domain in Example 6.2) Instead, in this example we see the relationships of the various keys in the database (let's continue to consider the assertions in example 6.1):



Entity	
PK	ID
FK	PersonID
FK	AnimalID
FK	PersonID

Person	
PK	PersonID
FK	AnimalID
FK	PersonID

Man	
PK	ManID
FK	DogID
FK	PersonID

Woman	
PK	WomanID
FK	DogID
FK	PersonID

Animal	
PK	AnimalID
FK	PersonID

Dog	
PK	DogID
FK	PersonID

Fig. 6.3 Database Relations.

Note that the foreign keys map to entity relationships, for instance, as from Exercise 6.9.

Observation 6.10 (Assertional languages, data, knowledge and mixed) The three examples clarify the scope and purpose of data, knowledge and mixed languages and domains. Data languages, describing facts in the real world formalize the contents of (relational) DBs or analogous representations (CSV , JSN, XML files). Knowledge languages specify the reference knowledge, e.g., relations, attributes, of data languages. Mixed languages are used to specify both.

6.3 Interpretation function

Definition 6.8 (Interpretation function, intensional interpretation) The **Intensional representation $\mathcal{I}^i_A$ of an interpretation function** $\mathcal{I}_A : \mathcal{L}_A \longrightarrow \mathtt{D}$ of an assertional language is defined as

$$\mathcal{I}^i_A =< \mathcal{I}_e, \mathcal{I}_C, \mathcal{I}_P > \tag{6.5}$$

with:

$$\begin{aligned} \mathcal{I}_e &: \mathcal{E} && \longrightarrow \mathtt{E} \\ \mathcal{I}_C &: \{\mathcal{C}\} && \longrightarrow \{\mathtt{E}\} \\ \mathcal{I}_P &: \{\mathcal{P}^n\} && \longrightarrow \{\mathtt{E}\} \times \cdots \times \{\mathtt{E}\} \end{aligned} \tag{6.6}$$

and such that:

$$\begin{aligned} \mathcal{I}_A(\mathcal{C}(e)) &= \mathcal{I}_C(\mathcal{C})(\mathcal{I}_e(e)) && = \mathtt{e} \in \mathtt{C} \\ \mathcal{I}_A(\mathcal{P}^n(e_1, \ldots, e_n)) &= \mathcal{I}_P(\mathcal{P}^n)(\mathcal{I}_e(e_1), \ldots, \mathcal{I}_e(e_n)) && = < \mathtt{e}_1, \ldots, \mathtt{e}_n > \in \mathtt{R}^n \\ \mathcal{I}_A(\mathcal{C}) &= \mathcal{I}_C(\mathcal{C}) && = \mathtt{C} \subseteq \mathtt{E} \\ \mathcal{I}_A(\mathcal{P}^n(C_1, \ldots, C_n)) &= \mathcal{I}_P(\mathcal{P}^n)(\mathcal{I}_C(C_1), \ldots, \mathcal{I}_C(C_n)) && = \mathtt{R}^n \subseteq \mathtt{C}_1 \times \cdots \times \mathtt{C}_n \end{aligned} \tag{6.7}$$

where: $\mathcal{I}_e$ is the **entity interpretation function**, $\mathcal{I}_C$ is the **concept interpretation function** and $\mathcal{I}_P$ is the **property interpretation function**

Notation 6.2 (Alternative notation for the interpretation function) Equation (6.7) is sometimes written as follows:

$$\begin{aligned} \mathcal{I}_A(\mathcal{C}(e)) &= \mathcal{C}^{\mathcal{I}}(e^{\mathcal{I}}) \\ \mathcal{I}_A(\mathcal{P}(e_i, e_j)) &= \mathcal{P}^{\mathcal{I}}(e_i^{\mathcal{I}}, e_j^{\mathcal{I}}) \\ \mathcal{I}_A(\mathcal{C}) &= \mathcal{C}^{\mathcal{I}} \\ \mathcal{I}_A(\mathcal{P}(\mathcal{C}_i, \mathcal{C}_j)) &= \mathcal{P}^{\mathcal{I}}(\mathcal{C}_i^{\mathcal{I}}, \mathcal{C}_j^{\mathcal{I}}) \end{aligned} \tag{6.8}$$

Observation 6.11 (Interpretation function) Equation (6.7) shows how $\mathcal{I}_A$ is applied recursively by applying it to the components of its input assertion till the interpretation of entities. In this process its components are applied as needed, that is, $\mathcal{I}_e$ to entities, $\mathcal{I}_C$ to concepts and $\mathcal{I}_P$ to properties. The one-to-one mapping between the language and the structure of the domain is exploited by the interpFretation function to build the meaning of an assertion compositionally starting from the meaning of its components.

Observation 6.12 (Interpretation function, intensional interpretation) When $\mathtt{D}^i$ and $\mathcal{L}^i_A$ are built following the rules highlighted in Observation 6.8 and $\mathcal{I}_A$ is as from Equation (6.7), then the representation of a the world is intuitive and self-documenting and mapped one-to-one to the analogical representation it is describing. This essentially allows the representation developer to focus only linguistic representations, only implicitly thinking about the corresponding analogical representation.

Definition 6.9 (Interpretation function, data, knowledge, mixed) A **data interpretation function** maps an assertional data language into a data domain. A **knowledge interpretation function** maps an assertional knowledge language into a knowledge domain. A **mixed interpretation function** maps a mixed language into a mixed domain.

Example 6.11 (A data interpretation function from the data language in Example 6.5 to the data domain in Example 6.2) An instance of application is

$$\begin{aligned}\mathcal{I}_A(\text{FriendOf1(sofia, paolo)} &= \mathcal{I}_P(\text{FriendOf1})(\mathcal{I}_e(\text{sofia}), \mathcal{I}_e\text{paolo})) = \\ &= <\mathtt{sofia,\ paolo}> \in \mathtt{FriendOf1}\end{aligned}$$

Example 6.12 (A knowledge interpretation function from the assertional language in Example 6.6 to the knowledge domain in Example 6.3) An instance of application is

$$\mathcal{I}_A(\text{FriendOf1}) = \mathcal{I}_P(\text{FriendOf1}) = \mathtt{FriednOf1} \subseteq \mathtt{Person} \times \mathtt{Person}$$

Example 6.13 (A knowledge interpretation from the ER model language in Example 6.1 to the knowledge domain in Example 6.3) An instance of application is

NEED PICTURE WITH MAPPING OF ONE CONCEPT AND ONE PREPERTY ELEMENT

Example 6.14 (A data interpretation function from the DB language in Example 6.9 to the data domain in Example 6.2) An instance of application is

NEED PICTURE WITH MAPPING OF ONE DATA ELEMENT

COMMENT ON HOW TO MAP NULL

Example 6.15 (A knowledge interpretation function from the DB schema language in Example 6.15 to the knowledge domain in Example 6.2) An instance of application is

NEED PICTURE WITH MAPPING OF ONE SCHEMA ELEMENT AND DATAE TYPES AND KEYS

COMMENT ON FOREIGN AND PRIMARY KEYS

6.4 World model

Definition 6.10 (World Model, intensional representation) Given a **World Model** $\hat{\mathcal{W}}$ is defined as

$$\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >,$$

its **intensional representation** $\hat{\mathcal{W}}^i$ is defined as

$$\hat{\mathcal{W}}^i =< \mathcal{L}_A^i, \mathtt{D}^i, \mathcal{I}_A^i > \tag{6.9}$$

with

$$\begin{aligned} \mathcal{L}_A^i &= < \mathcal{E}, \{\mathcal{C}\}, \{\mathcal{P}\} > \\ \mathtt{D}^i &= < \mathtt{E}, \{\mathtt{C}\}, \{\mathtt{R}\} > \\ \mathcal{I}_A^i &= < \mathcal{I}_e, \mathcal{I}_C, \mathcal{I}_P > \end{aligned}$$

where $\hat{\mathcal{W}}^i$ is (called) the **world model stencil**, $\mathtt{D}^i$ the **domain stencil**, $\mathcal{L}_A^i$ is the (assertional) **language stencil** and $\mathcal{I}_A^i$ is the **interpretation function stencil**.

Observation 6.13 (World model stencil) $\hat{\mathcal{W}}^i$ is all we need to define to a world model and to use it to implement theories and corresponding models. See Observations 5.13 and 5.14 and Definition 5.7.

Terminology 6.2 (Data, knowledge, mixed) We talk of **data, knowledge** and **mixed world models, theories** and **models** with the obvious meaning.

Observation 6.14 (World model) Definition 6.10 tells us that a world model is composed of a language $\mathcal{L}_A$ of assertions about a reference domain $\mathtt{D}$ where there the meaning of the assertions in $\mathcal{L}_A$ is unambiguous in the sense that the fact denoted by an assertion is univocally defined by the key role of the interpretation function $\mathcal{I}_A$. To this extent, it is worthwhile to summarize the key properties of interpretation functions, as introduced and discussed above.

- It does not allow for polysemy: an element in the language denotes one and only one element of the domain. Denotational ambiguity is not possible;
- It does allow for synonymity: one element of the domain my be denoted by more than on element of the language;
- It is total: any element of the language has a denotation in the reference domain. Meaningless assertions are not allowed;
- It is not (necessarily) surjective: there can be elements of the domain for which there is no elements in the language which denote them;

- It is compositional: the meaning of an assertion is computed by functional composition of its constituent elements. The meaning of assertions is intensionally encoded in their structure.

Observation 6.15 (Defining a world model) Building a world model $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$ requires going through the definition of its three components.The motivation is usually the need of modeling a certain problem domain. The modeller usually starts with an intuitive understanding of the domain, even if not worked out in full detail. The process usually follows the following steps:

1. Define $\mathcal{L}_A$, where the key idea is that $\mathcal{L}_A$ should capture the key aspects of the target domain;
2. Specify $\mathtt{D}$ making sure that it is a proper analogical representation of the target domain;
3. Define $\mathcal{I}_A$, in this process validating the fact that $\mathcal{L}_A$ and $\mathtt{D}$ are properly defined and mutually aligned.

Quite often the last two steps, at least initially, are performed only intuitively relying on the compositionality of the interpretation function. These steps are usually performed fully only when the world model has been in use for some time and its ambiguities, if any, have become explicit. Overall, the process of building world models is complex, requires a lot of expertise. It may take years to fully define a world model.

Example 6.16 (Defining a world model) All along this section, starting from Section 2, we have been constructing world models. Simple world models can be constructed by collecting language, domain and interpretation function from these examples and by suitable aligning them. Notice however that, also mentioned before, defining world models is a very complex task, requiring lots of expertise, filed validation and, ultimately, time. As already mentioned before, ER models, UML models, subsets of natural language enriched with suitable interpretation functions, are instances of world models.

Observation 6.16 (Using a world model - building and reasoning about a theory of the world) The problem is as described in Section 1.3 and Section 2.2. We have a world model and we need to use its machinery to build a set of assertions, i.e., a theory of the world, and we need to make sure what it represents, i.e., the reference model, and its properties. Once a theory is articulated with reference to a world model, then it is possible to study its properties to a general methodology and algorithms defined in the next subsection.

The key aspect in the definition or selection of a world model is the decision on the assertional language $\mathcal{L}_A$ and the level of ambiguity it allows.

Intuition 6.5 (Language, informal, semi-formal, Logical) World model languages are of three types, as follows:

- **Informal languages**, namely languages where the syntax of $\mathcal{L}_a$ is defined informally, for instance, in natural language and without using production rules.

- **Semi-formal languages** namely languages where the syntax of $\mathcal{L}_A$ is formally defined.
- **Formal languages**, also called **Logical languages**, namely languages where $\mathcal{L}_A$ as well as $\mathcal{I}_A$ are formally defined.

Example 6.17 (Language, informal, semi-formal, logical) With reference to the example above, we have the following

- Informal languages: all natural languages;
- Semi-formal languages: DBs' relational language, the ER and UML notation;
- Logical languages: the languages used in logics.

Observation 6.17 (Language, informal, semi-formal, logical) We have the following.

- Informal languages are easier to use as everybody knows them. In Computer Science they are typically used when writing early requirements, e.g., to be shown to customers. The main difficulty is that there is a high probability of misunderstanding. Now also used in the interactions with ChatBots.
- Semi-formal languages are typically used in Software Engineering when writing advanced requirements. They decrease the level of ambiguity and are very effective in the collaborative work among Software Engineers. They can also be used in automatic code generation. Problems may arise because of the presence of some level of semantic ambiguity
- Logical languages have two main uses: (i) The specification of highly critical SW and HW (e.g., safety or security critical systems) and (ii) the implementation of reasoning systems, typically Artificial Intelligence systems, capable of computing consequences from what is known.

6.5 Using a world model

Let us see how world models can be used in practice to represent and reason about the world. We focus on logical world models.

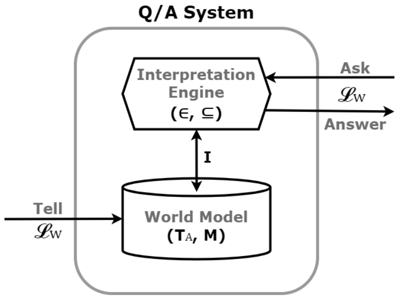


Fig. 6.4 Solving problems using a world model.

Intuition 6.6 (Solving problems using world models) The use of world models can be characterized as a solution to the need of answering questions (in natural language) or queries (in some formal language, e.g., SQL) about the world, thanks to the exploitation of some reasoning mechanism which allow to compute consequences from an existing body of knowledge about the world, i.e., a theory. This process is usually articulated in three main tasks, as follows (see also Figure 6.4):

- `Tell` *Data and Knowledge*, from a system *manager* to the *system*. This requires the specification of a language $\mathcal{L}_T$ which can be used to specify the facts of the world model being used.
- `Ask` *a Question*, from a *user* to the *system*, that is the specific questions / queries about which the user wants to know the answers. This requires the specification of a language $\mathcal{L}_Q$ used to write questions / queries.
- `Answer` *a Question*, from the *system* to the *user*, concerning the question which was previously asked, written in a language $\mathcal{L}_A$.

Some observations. Manager and user are usually different people. Usually the process of providing the system with additional information is continuous in time. Usually $\mathcal{L}_Q$ and $\mathcal{L}_A$ are the same language $\mathcal{L}$. Quite often the language $\mathcal{L}_T$ is different from both $\mathcal{L}_Q$ and $\mathcal{L}_A$. Three are the main motivations for this: efficiency of the reasoning, intuitiveness of the interaction with the user and, last but not least, the need for $\mathcal{L}_T$ to be most suited for the specification of the world model. For instance, as the following will make clear, graph-based models are very well suited for the specification of the world model, while, $\mathcal{L}_Q$ and $\mathcal{L}_A$ are often in (some fragments of) natural language. The translation across these languages is always automatic and implemented inside the Q/A system.

Terminology 6.3 (Language of the world model) In the following we assume that $\mathcal{L}_Q = \mathcal{L}_A = \mathcal{L}_T = \mathcal{L}_{\mathcal{W}}$, where $\mathcal{L}_{\mathcal{W}}$ is the representation **language of the world model**.

Definition 6.11 (Interpretation and entailment) Let $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$ be a world model. Let $\mathcal{T} \subseteq \mathcal{L}_A$ be a theory and $\mathtt{M} \in \mathtt{D}$ a model of $\hat{\mathcal{W}}$. Let $a \in \mathcal{T}$ be an assertion. Then we write

$$\begin{array}{ll} \mathtt{M} \models_{\mathcal{L}_A} a & \text{to mean } \mathcal{I}_A(a) \in M \\ \mathtt{M} \models_{\mathcal{L}_A} \mathcal{T} & \text{to mean } \mathcal{I}_A(a) \in M \text{ for all } a \in \mathcal{T} \end{array} \tag{6.10}$$

and say that $\mathtt{M}$ **entails** $\mathcal{T}$, or also that $\mathtt{M}$ **entails** a. The notation for the language $\mathcal{L}_A$ is dropped when not needed.

Intuition 6.7 (World models, reasoning problems) But which questions and which answers? All world models provide answers to four (basic) foundational questions that we state below as **reasoning problems**. Let us assume that we have a world model $\hat{\mathcal{W}}$ defined intensionally, that is, $\hat{\mathcal{W}}^i = < \mathcal{L}^i_A, \mathtt{D}^i, \mathcal{I}^i_A >$ and that we have a set $\{\mathtt{M}\}$ of models with $\mathtt{M} \subseteq \mathtt{D}$ and a set theories $\mathcal{T} \subseteq \mathcal{L}_A$. Then we have the following (notice that an assertion behaves like with one element):

Reasoning Problem 6.1 (Model checking) Given $\mathcal{T}$ and $\mathtt{M}$, check whether $\mathtt{M} \models \mathcal{T}$.

Observation 6.18 (Model checking and theory correctness) Model checking is the same as checking the correctness of a theory with respect to a model, as from Definition 5.7. Juts need to check whether the assertions in $\mathcal{T}$ occur in $\mathtt{M}$

Reasoning Problem 6.2 (Satisfiability) Given $\mathcal{T}$, check whether there exists $\mathtt{M}$ such that $\mathtt{M} \models \mathcal{T}$.

Observation 6.19 (Satisfiability) Any theory which does not represent negative information, as it is most often the case, is satisfiable. If negative information is allowed, it is sufficient to check whether the theory contains two facts which contradict each other.

Observation 6.20 (Query answering in DBs) Query answering in DBs is a sophisticated form of model checking / satisfiability. The contents of the DB are the reference world model, the query is the theory to be model checked, the answer is the set of instantiations which make the input theory correct.

Reasoning Problem 6.3 (Validity) Given $\mathcal{T}$, check whether for all $\mathtt{M}$, $\mathtt{M} \models \mathcal{T}$.

Observation 6.21 (Validity) Apply model checking to all models $\mathtt{M}$

Reasoning Problem 6.4 (Unsatisfiability) Given $\mathcal{T}$, check whether there is no $\mathtt{M}$ such that $\mathtt{M} \models \mathcal{T}$.

Observation 6.22 (Unsatisfiability) If negative information is allowed, heck for two contradictory assertions.

Intuition 6.8 (An architecture for solving problems using world models) The architecture supporting the use of world models, as specified in Intuition 6.6, is depicted in Figure 6.4. We can identified two main components, as follows

- A **world model (inference) engine** which encodes the available data and knowledge about the world and allows for minimal reasoning about them (see the reasoning problems in Intuition 6.7);

- An **Interpretation (inference) engine** implementing one or more of the reasoning problems defined in Intuition 6.7.

Notice that the system is selected at the beginning. The choice depends on the specifics of the problem be solved.

Observation 6.23 (World model inference engine) Reasoning in a world model amounts to checking to whether a query (a single assertion or a set of assertions as part of a theory) belongs to the model. More sophisticated queries, like the ones implemented in relational DBs can be implemented where one can leave some elements of the query unspecified. The inference engine will then find the proper s.

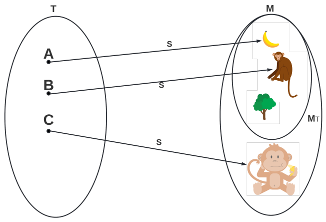
6.6 Exercises

Exercise 6.1 (ER Creation) Create an ER Model from this theory:

- There is a tree
- There is a banana
- The monkey is eating a banana
- The monkey is sitting on a tree*
- The monkey is scratching his head*

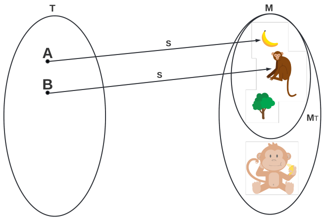
Exercise 6.2 (Complete and Correct?) Consider the sentences and the modeling of the theory. Say whether it is complete, correct, complete and correct, incomplete, or incorrect.

- A = "There is a banana"
- B = "There is a monkey"
- C = "There is a tree"
- D = "The monkey is eating a banana"



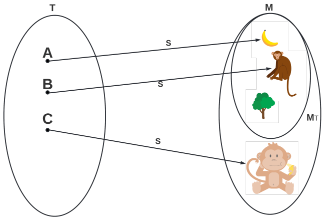
Exercise 6.3 (Complete and Correct?) Consider the sentences and the modeling of the theory. Say whether it is complete, correct, complete and correct, incomplete, or incorrect.

- A = "There is a banana"
- B = "There is a monkey"
- C = "There is a tree"
- D = "The monkey is eating a banana"



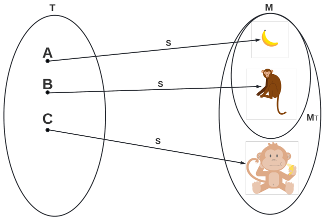
Exercise 6.4 (Complete and Correct?) Consider the sentences and the modeling of the theory. Say whether it is complete, correct, complete and correct, incomplete, or incorrect.

- A = "There is a banana"
- B = "There is a monkey"
- C = "The monkey is eating a banana"
- D = "There is a tree"



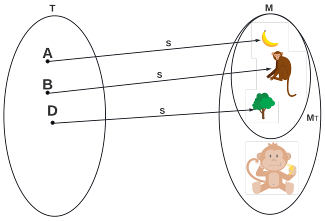
Exercise 6.5 (Complete and Correct?) Consider the sentences and the modeling of the theory. Say whether it is complete, correct, complete and correct, incomplete, or incorrect.

- A = "There is a banana"
- B = "There is a monkey"
- C = "There is a tree"
- D = "The monkey is eating a banana"



Exercise 6.6 (Complete and Correct?) Consider the sentences and the modeling of the theory. Say whether it is complete, correct, complete and correct, incomplete, or incorrect.

- A = "There is a banana"
- B = "There is a monkey"
- C = "There is a tree"
- D = "The monkey is eating a banana"



Chapter 7
Knowledge Graph - representing the world as a graph

We represent world models $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$ as **knowledge graph models** formally represented as $\hat{\mathcal{KG}} =< \mathcal{L}_{\mathcal{KG}}, \mathtt{D}, \mathcal{I}_{KG} >$, namely as special types of labeled graphs where each triple $<$ node, edge, node $>\in \mathcal{L}_{\mathcal{KG}}$ is an assertion $a \in \mathcal{L}_{\mathcal{KG}}$ describing a fact $\mathtt{f} \in \mathtt{D}$, with $\mathtt{f} = \mathcal{I}_{KG}(a)$. We call the theories in $\mathcal{L}_{\mathcal{KG}}$, **knowledge graphs**, and use the notation $\mathcal{KG} \subseteq \mathcal{L}_{\mathcal{KG}}$. Similarly to world models, we distinguish among $\hat{\mathcal{KG}}$'s representing data, knowledge or mixed information.

Definition 7.1 (Entity, etype and mixed $\hat{\mathcal{KG}}$) A $\hat{\mathcal{KG}}$ which is a data world model is called **entity graph model** ($\hat{\mathcal{E}}\mathcal{G}$). A $\hat{\mathcal{KG}}$ which is a knowledge world model is called **etype graph model** ($\mathcal{E}\hat{\mathcal{T}}\mathcal{G}$). A $\hat{\mathcal{KG}}$ which is a mixed world model is called **etype entity graph model** ($\mathcal{E}\hat{\mathcal{E}}\mathcal{G}$). This terminology extends in the obvious way to all the $\hat{\mathcal{KG}}$ components, as well as to all $\mathcal{KG}$'s and related models defined within a $\hat{\mathcal{KG}}$.

7.1 Domain

Let $\mathtt{KG} = \mathtt{D}^i =< \mathtt{E}, \{\mathtt{C}\}, \{\mathtt{R}\} >$ be the stencil of a KG domain. Let us define its components.

Definition 7.2 (Universe of Interpretation $\mathtt{E}$ of a $\mathtt{KG}$) The universe of interpretation $\mathtt{E}$ of a $\mathtt{KG}$ is defined as

$$\mathtt{E} = \mathtt{ET} \cup \mathtt{DT} \tag{7.1}$$

with $\mathtt{ET} \cap \mathtt{DT} = \emptyset$, where $\mathtt{ET} = \{\mathtt{E_T}\}$ with $\mathtt{E_T} = \{\mathtt{e}\}$, and $\mathtt{DT} = \{\mathtt{D_T}\}$ with $\mathtt{D_T} = \{\mathtt{v}\}$. $\mathtt{E_T}$ is an **entity type** (**etype**) and $\mathtt{D_T}$ is a **datatype** (**dtype**). The elements of etypes are called `entities`, those of dtypes are called `(data) values`.

Observation 7.1 (Etype, dtype) In a $\mathtt{KG}$, $\mathtt{E}$ is structured into a set of sub-universes, i.e., etypes and dtypes. In abstract, each such sub-universe is just like a class $\mathtt{C} \in \{\mathtt{C}\}$, namely a subset of $\mathtt{E}$. The fundamental difference is that etypes and dtypes are *types* which, as in programming languages, are defined when defining $\mathcal{L}_{\mathcal{KG}}$ and are therefore application independent. As such, these types come with certain properties and type operators builtin, most noticeably: a set of constructors which allow to build the elements of a type, a recognizer able to determine whether a certain element belongs to a certain type, and an equivalence relation which allows to decide whether two elements of that time are the same.

Example 7.1 (Etype) An example of etype is: `Location`, where intuitively a location is an etype which spatially contains other entities. Locations usually do not change their position with respect to their coordinate reference systems. Their space coordinates are therefore an important proxy for 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, the one which contains all elements in `ET` (its mot noticeable property is that it has a name, thus implementing the requirement that all entities must have a name); `Event`, whose most characterizing properties are its start and end times, `Person`, whose most characterizing properties are name, birth date, and parents; and many others.

Observation 7.2 (Dtype) Dtypes have the same properties as etypes plus two more: (i) the set of their members, i.e., their values, is predefined and (ii) the names of values are the same as the values themselves (that is data values denote themselves, thus for instance the number (properly called a numeral) `3.14` is the name of the number `3.14`).

Example 7.2 (Dtype) The following is a not exhaustive list of datatypes:

`Dtype, Float, Integer, Boolean, String, SpaceTime, Identifier`

where, `Float, Integer, Boolean, String` define, respectively the space of the real numbers, integers, boolean values, strings. `SpaceTime` is the set of values used to describe space and time. Thus, sub-dtypes of `SpaceTime` are `GeoCoordinate, Distance, XYCoordinate` but also `Date, Time, DateTime`, and so on. `Dtype` is the set of all the data values.

Observation 7.3 (Set of classes $\{\mathtt{C}\}$ **of a** `KG`**)** `KG` classes are world model classes "*as is*", see Definition 6.1.

Definition 7.3 (Object and data binary relations $\{\mathtt{R}\}$ **of a** `KG`**)** The set of relations $\{\mathtt{R}\} = \{\mathtt{OR}\} \cup \{\mathtt{DR}\}$ is a set of binary relations of a `KG` such that

$$\mathtt{R} \subseteq \mathtt{E}_{\mathtt{T}s} \times \{\mathtt{E}_{\mathtt{T}t} \cup \mathtt{D}_{\mathtt{T}t}\} \tag{7.2}$$

with $\mathtt{E}_{\mathtt{T}s}, \mathtt{E}_{\mathtt{T}t} \in \mathtt{ET}$ and $\mathtt{D}_{\mathtt{T}t} \in \mathtt{DT}$. If $\mathtt{R}$ is defined as:

$$\mathtt{R} \subseteq \mathtt{E}_{\mathtt{T}s} \times \mathtt{E}_{\mathtt{T}t} \tag{7.3}$$

then we say that $\mathtt{R}$ is a **binary object relation** $\mathtt{OR} \in \{\mathtt{OR}\}$. If $\mathtt{R}$ is defined as:

$$\mathtt{R} \subseteq \mathtt{E}_{\mathtt{T}} \times \mathtt{D}_{\mathtt{T}_t} \tag{7.4}$$

then we say that R is an **binary data relation** $\mathtt{DR} \in \{\mathtt{DR}\}$.

Observation 7.4 (Arity of a relation R) In a KG there are only binary relations, this allowing for the use of graph with single input and single output edges. The representation of a world model with more complex edges requires its reformulation to allow only for one-to-one edges.

Observation 7.5 (Arguments of a relation R) Differently from Definition 6.1, relations take as arguments etypes and dtypes. See Observation 7.1 for an explanation. We will later how to reintroduce relations taking as arguments used defined concepts.

Observation 7.6 (Object relation) Object relations are relations between entities thus depicting how entities interact. Thus, for instance, some examples are: `Near(Person, Tree)`, `HasFather(Person, Person)`, `TalksTo(Person,Dog)`. Notice that `Person, Tree, Person, Dog` are etypes, rather than concepts.

Observation 7.7 (Data relation) Data relations are depict properties of entities as such, independently of their interactions with other entities. Thus, for instance, some examples are: `Height(Person, Float)`, `HasName(Person, String)`, `HasId(Entity, Identifier)`. Notice that `Float, String, Identifier` are dtypes, rather than concepts, as it was the case in Section 5.

Observation 7.8 (Relation, cardinality) The cardinality of a relation can be: 1-to-1, 1-to-n, m-to-n (where one of m or n can also be 0, this meaning that there can be entities which do not belong to any relation).

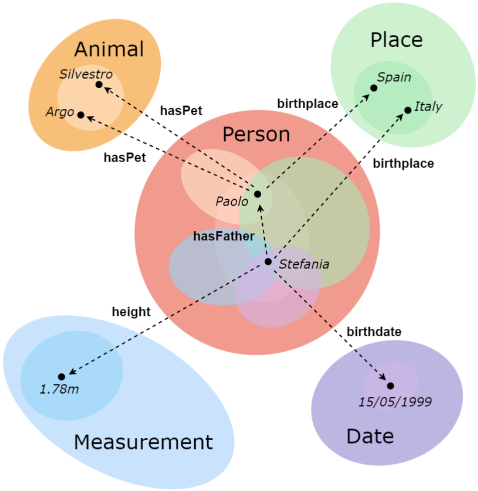


Fig. 7.1 A Venn Diagram of an EG.

Example 7.3 (Domain in an $\hat{\mathcal{EG}}$, Venn Diagram) We represent the domains of EGs as in Figure 7.1. Relations are represented as links between entities of the appropriate etype to entities or values of the appropriate etype or dtype, respectively.

Example 7.4 (Domain in $\mathcal{E}\hat{\mathcal{T}}\mathcal{G}$, Venn Diagram) We represent the domains of ETGs as in Figure 7.1. Relations are represented as links between etypes and etypes/ dtypes.

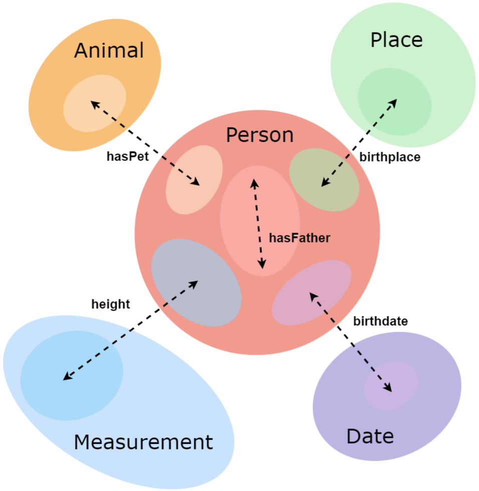


Fig. 7.2 A Venn Diagram of an ETG.

7.2 Assertional language

Let $\mathcal{L}^i_{\mathcal{KG}} = \mathcal{L}^i_A =< \mathcal{E}, \{\mathcal{C}\}, \{\mathcal{P}\} >$ be the stencil of the $\hat{\mathcal{KG}}$ language $\mathcal{L}^i_{\mathcal{KG}}$. The definition of $\mathcal{L}^i_{\mathcal{KG}}$ follows directly from the definition of $\mathtt{D}^i$ in Section 7.1.

Definition 7.4 (Concept) We have

$$\{\mathcal{C}\} = \mathcal{ET} \cup \mathcal{DT} \tag{7.5}$$

where $\mathcal{ET} = \{\mathcal{E}_\mathcal{T}\}$ and $\mathcal{DT} = \{\mathcal{D}_\mathcal{T}\}$, with $\mathcal{E}_\mathcal{T}$ and $\mathcal{D}_\mathcal{T}$ being, respectively, (names of) the **etypes** and **dtypes** in $\hat{\mathcal{KG}}$.

Definition 7.5 (Object and data property) We have

$$\{\mathcal{P}\} = \{\mathcal{OP}\} \cup \{\mathcal{DP}\} \tag{7.6}$$

where $\{\mathcal{OP}\}$ and $\{\mathcal{DP}\}$ are defined as follows:

$$\begin{aligned}\{\mathcal{OP}\} &\subseteq \{\mathcal{E}_\mathcal{T}\} \times \{\mathcal{E}_\mathcal{T}\} \\ \{\mathcal{DP}\} &\subseteq \{\mathcal{E}_\mathcal{T}\} \times \{\mathcal{D}_\mathcal{T}\}\end{aligned} \tag{7.7}$$

The elements of $\{\mathcal{OP}\}$ are called **object properties**, those of $\{\mathcal{DP}\}$ **data properties**.

Example 7.5 ($\mathcal{EG}$) The $\mathcal{EG}$ of this example represents the domain depicted in Example 7.3. The key observation is that the number of nodes corresponds to the number of entities and values, while the number of edges corresponds to the instantiated property values. Notice how n-to-m cardinality constraints allow for edges labeled with the same property to come out of the same etype node. Note also how datatype nodes are and can only be leaf nodes.

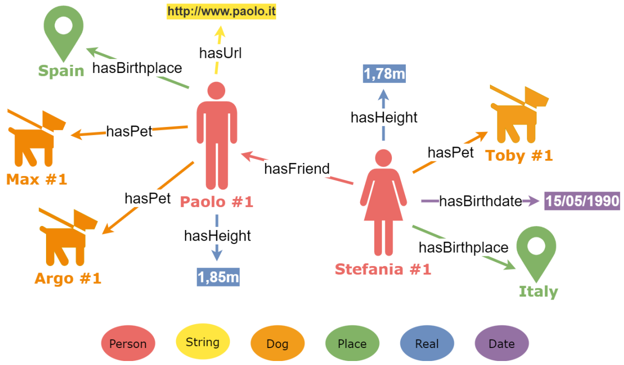


Fig. 7.3 Example of an EG representing information regarding humans

Example 7.6 ($\mathcal{ETG}$) The $\mathcal{ETG}$ of this example represents the domain depicted in Example 7.4. There is node per etype and data type. The $\mathcal{ETG}$ encodes some meta-information, e.g., the cardinality constraints, which can drive and control the instantiation of the $\mathcal{EG}$'s build starting from an $\mathcal{ETG}$. Similarly to $\mathcal{EG}$'s, dtype nodes are leaves.

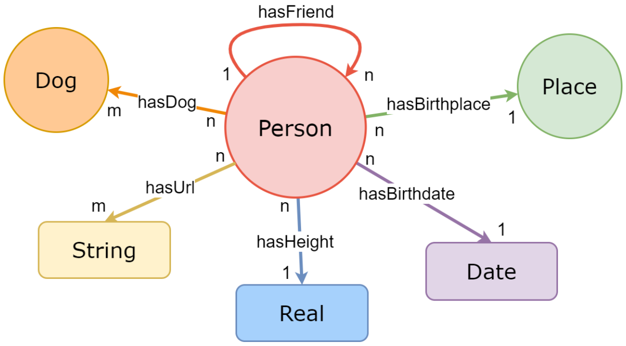


Fig. 7.4 Example of an ETG representing information regarding humans

Observation 7.9 **($\hat{\mathcal{EG}}$, $\mathcal{E}\hat{\mathcal{T}}\mathcal{G}$, $\mathcal{E}\hat{\mathcal{E}}\mathcal{G}$)** In $\hat{\mathcal{KG}}$'s assertions are triples $<$ node, edge, node $>\in \mathcal{L}_A$. In $\hat{\mathcal{EG}}$'s nodes represent either entities with their etype or values with their dtype. In $\mathcal{E}\hat{\mathcal{T}}\mathcal{G}$'s nodes represent etypes. $\mathcal{E}\hat{\mathcal{E}}\mathcal{G}$'s have both types of nodes. Edges are labeled by property names and represent relations. Edges from etypes/ entities to etypes/ entities represent object relations. Edges from etypes/ entities to dtypes/ values represent data relations.

7.3 Interpretation function

The interpretation function of a $\hat{\mathcal{KG}}$ is a direct mapping from the assertional language $\mathcal{L}^i_{\mathcal{KG}}$ to the target domain of interpretation $\mathtt{D}$. As it can be seen from the examples in Section 7.1 and Section 7.2, there is an almost direct mapping between the language and the domain of interpretation. In practice this means that, once the denotation of the single language elements of $\mathcal{L}_{\mathcal{KG}}$ is clarified, and it is made sure that the interpretation functions satisfies all the requirements (see Section 6.4), the intended meaning of a $\mathcal{KG}$ can be directly read off the $\mathcal{KG}$ itself.

7.4 Knowledge graph

Knowledge graph models of the world $\hat{\mathcal{KG}} =< \mathcal{L}^i_{\mathcal{KG}}, \mathtt{D}^i, \mathcal{I}^i_{KG} >$ are a universal, intuitive and self-explaining and computationally efficient representation of the world. Once a $\hat{\mathcal{KG}}$ is provided, it is sufficient to build your own favourite $\mathcal{KG}$ and all the operations described in Section 6.5 are available.

Observation 7.10 (Types of knowledge graphs) As also discussed in Observation 6.3, the world models and knowledge graphs introduced so far have minimal expressibility. In practice, knowledge graphs are often enriched with further constructors which allow for the description of more complex facts. This is a perfectly fine operation with the catch that one must be be careful in selecting the right trade-off between complexity, intuitiveness and computational complexity of the selected representation.

Observation 7.11 (From knowledge graphs to Logics) $\mathcal{KG}$'s allow for an easy embedding of in the most appropriate logic with the goal of enabling reasoning about them. This will be the topic of the following sections.

7.5 Exercises

Exercise 7.1 (Create a set diagram) Consider this map of the Milan metro system:

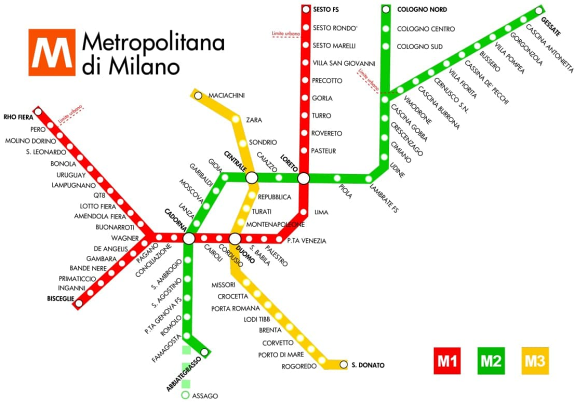


Fig. 7.5 Milan underground map

What needs to be done:

- Extract **relevant sets of objects**: Lines, Stations, Intersections, Terminals, Number of stations, Colors, ...
- Instatiate **elements of the domain**: redLine, Cadorna, RHO fiera, Milan, Yellow
- Extract **relevant relations**: hasColor, belongsTo, isPartOf, nearTo, ...

Chapter 8
Logic - extensional representation

World models allow us to specify the main components needed to build a representation of the world, both extensionally and intensionally, to be later used for solving problems. But this is not sufficient.

Observation 8.1 (Assertional languages, limitations) The description of domains and models using *only* assertions is very limited. One would like to have more flexible ways to describe them. One such example of richer linguistic description if the natural language description of Figure 3.1 reported in Example 8.1

Example 8.1 (A rich linguistic description) Consider the analogical representation in Figure 3.1. A good informal theory describing its contents can be articulated as follows.

"Paolo, Stefania and Sofia, three great friends, are in a lovely park, surrounded by the beauty of nature. They each have four adorable dogs with them, ready for a leisurely walk. Paolo, with his infectious smile, holds the reins of four friendly dogs: Argo, who seems to be the most energetic animal in the group; Luna, with dark fur and bright eyes; Max, a friendly crossbreed; and Penny, a small, affectionate dog. Stefania, with her passion for animals, enjoys her four faithful companions: Toby, an affectionate dog with a constantly moving tail; Ginger, a sweet brown-and-white dog with pointed ears; Rocky, rugged-looking but tender-hearted; and Luna, a cute and playful puppy. Sofia, with her quiet air, walks elegantly followed by her four adorable dogs: Balto, a thick-coated, pointy-eared dog; Rex, a protective and loyal soul; Bella, an elegant and affectionate pooch; and Charlie, a friendly breed mix with his tongue out, ready to make friends."

Being able to express linguistic descriptions like the one in Example 8.1 requires extending substantially the expressibility of the languages used to describe world models, the so-called representation languages. This forces us to shift from world models to logics.

8.1 Domain

The goal is to have a more expressive language, if compared to that of a world model, for describing a domain of interpretation, leaving the latter unchanged. See Section 5.1.

Observation 8.2 (Domain, model) As from Definition 5.2, we have $\mathtt{M} = \{\mathtt{f}\} \subseteq \mathtt{D}$, where $\{\mathtt{f}\}$ is a fact, $\mathtt{M}$ a model and $\mathtt{D}$ a domain of interpretation.

8.2 Representation language

Definition 8.1 (Representation language, atomic formulas, complex formulas, representation interpretation function) Let $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$ be a world model with $\mathcal{I}_A : \mathcal{L}_A \longrightarrow \mathtt{D}$. Let $\mathcal{L}_a$ be such that $\mathcal{L}_A \subseteq \mathcal{L}_a$ and such that there is a **representation interpretation function** $\mathcal{I} : \mathcal{L}_a \longrightarrow \mathtt{D}$, with $\mathcal{I}_A \subseteq \mathcal{I}$. Then, a **representation language** $\mathcal{L}$ is defined as

$$\mathcal{L} = \{w\} = \mathcal{L}_a \cup \mathcal{L}_c, \quad \text{with } \mathcal{L}_a \subset \mathcal{L}_c. \tag{8.1}$$

where: $\mathcal{I}_A$ is as in Definition 6.7, $w \in \mathcal{L}$ is a **(well-formed) formula**, $w \in \mathcal{L}_a$ is an **atomic (well-formed) formula** and $w \in \mathcal{L}_c$ is a **complex (well-formed) formula**. In turn $\mathcal{L}_a$ and $\mathcal{L}_c$ are the **language of atomic formulas** and **of complex formulas**, respectively.

Observation 8.3 (Language) Definition 8.1 of (representation language) formalizes the informal notion of language provided in Intuition 2.6.

Observation 8.4 (Representation interpretation function) An interpretation representation function $\mathcal{I}$ extends an interpretation function $\mathcal{I}_A$, that is $\mathcal{I}_A \subseteq \mathcal{I}$, in the sense that it behaves the same as $\mathcal{I}_A$ for all assertions $a \in \mathcal{L}_a$. Representations interpretation functions are defined in Definition 8.4.

Observation 8.5 (Representation language and world model) Given a world model $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$ the tuple $\hat{\mathcal{W}} =< \mathcal{L}, \mathtt{D}, \mathcal{I} >$ is not a world model for two reasons. The first is that, as we will detail below, $\mathcal{L}_a$ contains formulas (those in $\mathcal{L}_a - \mathcal{L}_A$) which cannot be interpreted by an interpretation function. The second is that $\mathcal{L}_c$ contains formulas (those in $\mathcal{L}_c - \mathcal{L}_a$) for which there is no interpretation. As discussed in Observation 8.1 and exemplified in Example 8.1, the overall goal is to provide more flexibility in the way world models are described. This is achieved two ways: via atomic formulas by extending the scope of interpretation functions, still maintaining the same domain; and via complex formulas which elaborate linguistically on the contents of the domain. Reading the text in Example 8.1 it is easy to find out that a lot of that text only loosely relates to the picture in Figure 3.1.

Definition 8.2 Given a representation language $\mathcal{L}$, a **theory** $\mathcal{T}$ is defined as

$$\mathcal{T} = \{w\} \subseteq \mathcal{L} \tag{8.2}$$

Observation 8.6 (Theory) Definition 8.2 generalizes in the obvious way Definition 5.4

Observation 8.7 (What representation languages represent) As also highlighted by Example 8.1, the existence of representation languages is motivated by the fact there is information that cannot be easily and intuitively in a world model. To provide some examples:

- *Negative information*. World models usually do not represent what is not the case in the part of the world being represented. And there is indeed a very good reason for his. Namely for anything being represented, there are indefinitely many things that this anything is not. Thus for instance, you see a woman with blond hair and various characteristics. Well, this woman, is not a man, not an animal, not with black hair, not with red hair, not at home, not ..;
- *Partial information*. You perceive a person from a certain distance and you cannot distinguish whether it is woman or a man;
- *Consequential information*. Since you see a woman, you know, as a consequence, that it is also a person
- *Equivalent information.* Since you see yellow car with a plate on the roof, you know it is a taxi
- ... and much more

Natural language descriptions like the one in Example 8.1 usually complement the description of the world model with lots of additional information which, hopefully, will facilitate a correct, non-diverging, mental model of the world model being describes. The take home message is that world models are direct linguistic representations of mental analogical representations. As such they can hardly represent information which is not explicitly encoded in an analogical representation but which is, rather, about how observers relate to analogical representations

Observation 8.8 (What representation languages do not represent) Representation languages describe world models. But they do not capture the *pragmatics* of how linguistic descriptions are used, nor the additional text used to enforce these pragmatics, for instance: the fact that one likes what (s) sees, whether (s)he is upset, whether (s)he is trying to convince someone else of the truthfulness of what (s)he is describing, and much more.

Terminology 8.1 ((Representation) language and interpretation function) For simplicity, whenever no confusion arises, in the following we talk of **language** and **interpretation function** meaning **representation language** and **representation interpretation function**, respectively.

Observation 8.9 (Atomic formulas) Assertions are atomic formulas, but some atomic formulas are not assertions. The key property that atomic formulas share with assertions is that they are interpreted by an interpretation function. In other words the meaning of atomic formulas, like that of assertions, can be computed directly from the domain.

Definition 8.3 (Atomic assertions, complex assertions) Given a language $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$, $\mathcal{L}_a$ is defined as

$$\mathcal{L}_a = \mathcal{L}_A \cup \mathcal{L}_{AC} \tag{8.3}$$

where: $\mathcal{L}_A$ is an assertional language, that we also call a **language of atomic assertions**, and $\mathcal{L}_{AC}$ is a **language of complex assertions**.

Observation 8.10 (Atomic formulas, knowledge and data operators) Atomic formulas are distinguished based on whether they use operators which generate

Example 8.2 (Atomic formulas, knowledge operators) We can build complex atomic formulas as follows. If $\mathcal{C}_1$ and $\mathcal{C}_2$ are concepts, then (following the notation of $\mathcal{LOD}$), $\mathcal{C}_1 \sqcap \mathcal{C}_2$, is also a concept description, where $\mathcal{C}_i$ can be an atomic assertion as well as a complex assertion. Examples of assertions in this language are (";" is used to separate different formulas):

"being a person" ; "having blond hair" ; "having a dog" ;
"having blond hair $\sqcap$ "being a person" ;
"being a person" $\sqcap$ "having blond hair" $\sqcap$ "being a person"
...

... and so on, with indefinitely long complex atomic formulas. As from $\mathcal{LOD}$ is that the interpretation of $\mathcal{C}_1 \sqcap \mathcal{C}_2$ is the intersection of the interpretations of $\mathcal{C}_1$ and $\mathcal{C}_2$ Thus for instance the atomic concept in the second line denotes a person with blond hair.

Example 8.3 (Atomic formulas, data operators) !!DA FARE!! FUNZIONI ESEMPIO CON "friendOf" e "nearTo"

Observation 8.11 (Complex formulas) Differently from atomic formulas, complex formulas are not interpreted by an interpretation function. The idea is that complex formulas allow to compose atomic formulas in longer complex articulations whose meaning can be somehow deduced from the meaning of the constituent atomic formulas via entailment, see below.

Example 8.4 (Complex formulas) We can build complex atomic formulas as follows. If A_1 and A_2 are formulas, then (following the notation of $\mathcal{LOP}$), A_1 $\underline{\texttt{xor}}$ A_2, is also a formula where A_i can be an atomic as well as a complex formula. Examples of assertions in this language are (";" is used to separate different formulas):

"Sofia is a person" ;
"Sofia is a person" $\underline{\texttt{xor}}$ "Sofia is a person";
"Sofia is a person" $\underline{\texttt{xor}}$ "Paolo is a man" ;
("Sofia is a person" $\underline{\texttt{xor}}$ "Paolo is a man") $\underline{\texttt{xor}}$ "Paolo is a dog" ;

... and so on, with indefinitely long complex formulas. The intuition is that A_1 $\underline{\texttt{xor}}$ A_2 contains one and only one fact between the facts denoted by A_1 and A_2.

Example 8.5 (Representation languages) The following are examples of representation languages:

1. All the natural languages, as used by people in their everyday life;
2. The language of arithmetics which describes how to perform plus and minus operations on natural numbers. The language of arithmetic is a simplified natural language which allows to mention, among others, numbers, variables, plus, minus, times, and also to compose phrases in more complex phrases;
3. Relational database (DB) languages do not extend to representation languages;
4. Entity-relationship (ER) languages do not extend to representation languages.

Observation 8.12 (Graph-based languages and representation languages) Graph-based languages (e.g., ER, or DB languages) in general are hard to extend to representation languages due to the inherent difficulty of keeping intuitiveness, which is the key feature of graph-based languages. Representation languages are most easily representation in languages which have a sequential structure like that of natural languages.

8.3 Interpretation Function

Definition 8.4 (Interpretation function) Given a language $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$, $\mathcal{L}_a$ with $\mathcal{L}_a = \mathcal{L}_A \cup \mathcal{L}_{AC}$. Let $\mathtt{D}$ be a domain. Then an **Interpretation Function** $\mathcal{I}$ for $\mathcal{L}_a$ is defined as

$$\mathcal{I} : \mathcal{L}_a \ \rightarrow \ \mathtt{D} \quad (\mathcal{I} \subseteq \mathcal{L}_a \times \mathtt{D}) \tag{8.4}$$

with

$$\mathcal{I} = \mathcal{I}_A \circ \mathcal{I}_{AC} \tag{8.5}$$

where

$$\begin{array}{l} \mathcal{I}_{AC} : \mathcal{L}_a \rightarrow \mathcal{L}_A \\ \mathcal{I}_A \quad : \mathcal{L}_A \rightarrow \mathtt{D} \end{array} \tag{8.6}$$

where: $\mathcal{I}_A$ is an **interpretation function for atomic assertions** and $\mathcal{I}_{AC}$ is an **interpretation function for complex assertions**. $\mathcal{I}_A$ is as defined in Definition 5.5 and 6.8. Furthermore, we say that a fact $\mathtt{f} \in \mathtt{M}$ is the **interpretation** of $w \in \mathcal{L}_a$, and write

$$\mathtt{f} = \mathcal{I}(a) = a^{\mathcal{I}} \tag{8.7}$$

to mean that w is a linguistic description of $\mathtt{f}$. We say that $\mathtt{f}$ is the **interpretation of** w, or, equivalently, that w **denotes** $\mathtt{f}$.

Observation 8.13 (Interpretation function) Definition 8.4 extends Definition 5.5 to apply to atomic formulas which are not assertions. All considerations made in Section 5.3 apply.

8.4 Entailment

The meaning of representation languages is computed by reducing the meaning of complex formulas to that of their constituent atomic formulas.

Definition 8.5 (Entailment relation) Let $\mathtt{M} \subseteq \mathtt{D}$ and $w \in \mathcal{L}$ be a formula. Then $\models$, to be read "**entails**", is an **entailment relation** defined as

$$\models\ \subseteq\ \mathtt{D} \times \mathcal{L} \tag{8.8}$$

We also write

$$\mathtt{M}\ \models_{\mathcal{L}}\ \mathcal{T} \qquad (\mathtt{M}\ \models_{\mathcal{L}}\ w) \tag{8.9}$$

with $\{\mathtt{M}\}$ being a set of models $\mathtt{M}$ and $\mathcal{T}$ a set of formulas w. $\mathtt{M} \models \mathcal{T}$ stands for $\mathtt{M} \models w$ for all $w \in \mathcal{T}$. The notation for the language $\mathcal{L}$ is dropped when not needed. We say that $\mathtt{M}$ **entails** w, or also that $\mathtt{M}$ **entails** $\mathcal{T}$.

Definition 8.6 (Entailment of an atomic formula) If w is an atomic formula then we have

$$\mathtt{M}\ \models\ w \quad \text{if and only if} \quad \mathcal{I}(w) \in \mathtt{M} \tag{8.10}$$

Observation 8.14 (Entailment of atomic formulas) Entailment of atomic formulas reduces to their interpretation.

Observation 8.15 (Entailment of complex formulas) Entailment of complex formulas operates in two steps, similarly to how interpretation functions operate on complex atomic formulas. In the first step, it reduces the entailment of a complex formula to that of its component atomic formulas. In the second step it applies the interpretation function o atomic formulas.

Observation 8.16 (Entailment relation) Interpretation is a function. Entailment is a relation. It is a many-to-many relation. There may be multiple atomic and/or complex formulas that denote a fact and, symmetrically, for the same formula there maybe multiple facts entailed by it (the latter property being the one which makes entailment a relation).

Observation 8.17 (Entailment relation and interpretation function) The key difference between interpretation being a function and entailment being a relation is that with entailment we allow a formula (theory) to have multiple meanings, i.e., to denote multiple (sets of) facts while interpretation does not allow for assertions to be polysemous. The key motivation for this is that certain logics, in particular those used to formalize decision making, e.g., $\mathcal{LOP}$, allow to model *partial knowledge*, that is, the fact that a person does not have complete knowledge about the world, a situation which is intrinsic to human knowledge. In the case of partial knowledge, a formula not present in the model can be (not always) indifferently taken as holding or not holding See also Observation 8.18.

Observation 8.18 (Entailment relation and interpretation function, atomic and complex formula) One could think of distinguishing between atomic and complex facts and say that they are denoted by atomic and complex formulas, respectively. We do not take this step as there is not such thing as a complex fact. Facts are in the world. What is described by a complex formula is not in the world, it is in the mind, it is in how a person describes the world. Think, for instance, of a formula stating my partial knowledge about the current situation, for instance the fact that on the table there is a pen or a pencil (where I say this simply because I am not close enough and I cannot see clearly). In the world there is no such thing as a pen or a pencil. On the table there is either a pen or a pencil, given that I can clearly perceive that there is only one object!

Definition 8.7 (Interpretation and entailment) If an interpretation $\mathcal{I}$ is a model for a theory $\mathcal{T}$ (or a formula w), then we say $\mathcal{I}$ entails $\mathcal{T}$ (or w) and write

$$\mathcal{I} \models \mathcal{T} \quad (\mathcal{I} \models w) \tag{8.11}$$

Example 8.6 (Complex formulas) Consider complex formulas as defined in Example 8.4, that is formulas of the form A_1 $\underline{\texttt{xor}}$ A_2, where A_i is any formula. Let us assume that A_1 and A_2 are atomic formulas. Then A_1 $\underline{\texttt{xor}}$ A_2 will be denoted by a model $\texttt{M}$ containing the denotation of A_1 or by one containing the denotation of A_2. In formulas:

$$\begin{array}{ll}
\mathcal{I}(A_1) & \models A_1 \\
\mathcal{I}(A_1) & \models A_1 \; \underline{\texttt{xor}} \; A_2 \\
\mathcal{I}(A_1) & \not\models A_1 \; \underline{\texttt{xor}} \; A_2 \\
\{\mathcal{I}(A_1), \mathcal{I}(A_2)\} & \not\models A_1 \; \underline{\texttt{xor}} \; A_2
\end{array}$$

The above example shows how many models may denote the same formulas (first three equations) ad also how the same formula may denote multiple models (first and last equation).

Entailment is how we formalize (human) reasoning.

Definition 8.8 (Logical entailment) Let $\texttt{M} \subseteq \texttt{D}$ be a model and $\mathcal{T}_1, \mathcal{T}_2 \subseteq \mathcal{L}$ be two theories and $w \in \mathcal{L}$ a formula. Then we write

$$\mathcal{T}_1 \models_{\{\texttt{M}\}} \mathcal{T}_2 \quad (\mathcal{T}_1 \models_{\{\texttt{M}\}} w) \tag{8.12}$$

and say that $\mathcal{T}_1$ **(logically) entails** $\mathcal{T}_2$ (w) with respect to the **set of models** $\{\texttt{M}\}$ if

$$\text{for all } \; \texttt{M} \in \{\texttt{M}\}, \quad \text{if} \quad \texttt{M} \models \mathcal{T}_1 \quad \text{then} \quad \texttt{M} \models \mathcal{T}_2 \quad (\texttt{M} \models w)$$

Terminology 8.2 (Logical entailment v.2, v.3) Sometimes Logical entailment is defined as

$$\mathcal{T}_1 \models \mathcal{T}_2 \quad (\mathcal{T}_1 \models w) \tag{8.13}$$

to mean **all models** $\mathtt{M} \subseteq \mathtt{D}$, that is

$$\text{for all } \mathtt{M} \subseteq \mathtt{D}, \quad \text{if} \quad \mathtt{M} \models \mathcal{T}_1 \quad \text{then} \quad \mathtt{M} \models \mathcal{T}_2 \quad (\mathtt{M} \models w)$$

Sometimes logical entailment is defined as

$$\mathcal{T}_1 \models_{\mathcal{T}} \mathcal{T}_2 \quad (\mathcal{T}_1 \models_{\mathcal{T}} w) \tag{8.14}$$

to mean **all models that entail** $\mathcal{T}$, that is

$$\text{for all } \mathtt{M} \text{ such that } \mathtt{M} \models \mathcal{T}, \quad \text{if} \quad \mathtt{M} \models \mathcal{T}_1 \quad \text{then} \quad \mathtt{M} \models \mathcal{T}_2 \quad (\mathtt{M} \models w)$$

We will use all these definitions interchangeably, as they are different formalizations of the same notion

Observation 8.19 (Logical entailment) Logical entailment must be read as follows. Take one or more models $\mathtt{M}$. Then keep all the models that entail $\mathcal{T}_1$ and discard the others. If all such models entail also $\mathcal{T}_2$ then $\mathcal{T}_1$ entails $\mathcal{T}_2$. Notice the key role of $\mathcal{T}_1$ in forcing the discharge of the models where it does not hold. $\mathcal{T}_1$ forces the focus only on a precise set of models.

Observation 8.20 (Logical entailment and reasoning) Logical entailment is the formalization of the process of reasoning. We start with some assumptions, that we model as one or more models, and see whether, under these same assumptions also $\mathcal{T}_2$ holds. This is exactly how we intuitively reason. Notice that the vice versa does not hold. Namely, if $\mathtt{M} \not\models \mathcal{T}_1$ then we may have $\mathtt{M} \models \mathcal{T}_2$ or $\mathtt{M} \not\models \mathcal{T}_2$

Example 8.7 (Reasoning via entailment) One, for instance, would like to deduce from the fact that if it rains then people don't leave home and the fact that it rains, the fact that people don't leave home. Or that if my hair is blond then it is not black. Or that, being near you excludes being far from you. And so on. Of course the kind of conclusions that one can draw depend on the specifics of how the entailment relation is define. The definition of logics which focus on particular types of reasoning will be the topic of the following sections.

Terminology 8.3 (Entailment, axiom, theorem) Consider the following form of entailment: $\mathcal{T}_1 \models w$. Historically the finite set of formulas $w \in \mathcal{T}$ are called **axioms** while its infinite logical consequences are called **theorems**. Axioms are guaranteed to hold, to be some a priori information in our case, for instance, represented by a $\hat{\mathcal{KG}}$ theory.

Terminology 8.4 (Deductive reasoning) The type of reasoning implemented by logical entailment is called **deductive reasoning**.

Terminology 8.5 (Reasoning, forward and backward, goal) Forward reasoning is the reasoning process by which one tries to prove a theorem, so called the **goal**, by deriving logical consequences from the axioms. Dually, with **backward reasoning**

one starts for the goal and tries to reduce the holding of the goal to the holding of the axioms, which are known to hold.

Observation 8.21 (Reasoning, forward and backward) At the start of art, in AI all inference engines work backward, the reason being that in this way the inference engine can exploit the information encoded in the structure of the goal.

Independently of the specifics of an entailment relation, and similarly to interpretation functions, entailment relations are requested to satisfy a certain set of principles which constrain how we want reasoning to behave. Let $\Gamma = \{w\}$, $\Sigma = \{w\}$ be sets of formulas and w, w_1, w_2 be formulas.

Intuition 8.1 (Reflexivity)

$$w \models w \tag{8.15}$$

Observation 8.22 (Reflexivity) Every fact entails itself. Knowledge asserts itself as being knowledge. This is the essence of what knowledge is about.

Intuition 8.2 (Cut)

$$\text{If} \quad \Gamma \models w_1 \quad \text{and} \quad \Sigma \cup \{w_1\} \models w_2 \quad \text{then} \quad \Gamma \cup \Sigma \models w_2 \tag{8.16}$$

Observation 8.23 (Cut) There are two ways to interpret cut. The first and most common is that reasoning can be made efficient by dropping intermediate irrelevant results. The second is transitivity, namely the fact that reasoning con be composed by chaining independent reasoning sessions, something that people do all the time during their everyday life.

Intuition 8.3 (Compactness)

$$\text{If} \quad \Gamma \models w \quad \text{then there is a finite subset} \quad \Gamma_0 \subseteq \Gamma \quad \text{such that} \quad \Gamma_0 \models w \tag{8.17}$$

Observation 8.24 (Compactness) Consider infinity as the possibility of describing another fact in the process of reasoning. Thus, for instance, natural numbers are infinite and, no matter how many numbers have already been used so far, it is always possible to provide a new one. Compactness says that logical consequence must be computed using a finite set of assumptions. Logical consequence for an hypothetically infinite set of formulas is not a behaviour that is considered of interest.

Intuition 8.4 (Monotonicity)

$$\text{If} \quad \Gamma \models w \quad \text{then} \quad \Gamma \cup \Sigma \models w \tag{8.18}$$

Observation 8.25 (Monotonicity) Monotonicity implements a fundamental and intuitive property of knowledge, for instance of scientific knowledge. If knowledge increases then what can be derived from it via reasoning can only increase. At most it can stay the same if the new piece of knowledge was implied by what is already known.

Intuition 8.5 (NonMonotonicity)

$$\Gamma \models w \quad \text{and} \quad \Gamma \cup \Sigma \not\models w \tag{8.19}$$

Observation 8.26 (NonMonotonicity) Despite its intuitiveness, monotonicity is a property which most often does not hold. This is extensively the case with commonsense reasoning, a topic extensively studied in AI. How many times getting to know something new has forced us to change our mind? The historical AI example of commonsense nonmonotonic reasoning is that the belief that all birds fly can be defeated by the fact that penguins are birds and they do not fly. Scientific knowledge is full of examples where prior knowledge was defeated by later evidence, and lots of theories in the philosophy of science have been built about it. The standard example of the nonmonoticity of scientific knowledge is the discovery that it is the earth rotating around the sun, and not vice versa. From a practical point of view all the logics which formalize mathematical reasoning and used in formal methods, as applied to, e.g., programming languages, are monotonic, while most logics defined in AI are nonmonotonic. Negation by failure, as implemented in relational DBs is nonmonotonic.

8.5 Logic

The journey is complete. We have only to pull everything together.

Observation 8.27 (The roles of $\mathtt{D}$, $\mathcal{L}$, $\mathcal{I}$, $\models$, $\mathtt{M}$, $\mathcal{T}$) The definitions provided in the previous sections can be summarized in the following figure.

$$\begin{tikzcd}
a \arrow[d, "\mathcal{I}"] & \mathcal{T} \arrow[l, "\in"'] \arrow[r, "\subseteq"] \arrow[d, dashed, "\models"] & \mathcal{L}_a \arrow[d, "\mathcal{I}"] \\
\mathtt{f} & \mathtt{M} \arrow[l, "\in"'] \arrow[r, "\subseteq"] & \mathtt{D}
\end{tikzcd} \tag{8.20}$$

with $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$.

Compare also with Figure 5.7 for world models. In Equation (8.20), $\mathtt{D}$ defines the set of facts $\mathtt{f}$ of potential interest, $\mathcal{L}_a$ the set of assertions a of potential interest, $\mathcal{L}$ the formulas w that we can use to describe the facts in $\mathtt{M}$, $\models$ the conditions under which any $w \in \mathcal{L}$ is entailed by $\mathtt{M}$, $\mathtt{M}$ the set of facts we are focusing on and, finally $\mathcal{T}$ is the theory describing $\mathtt{M}$. The key difference from the corresponding world model equation, is that here the relation between model has been extended from the interpretation function to the entailment relation.

Definition 8.9 (Logic) Take any representation as from Equation (8.20). Then

$$\hat{\mathcal{L}} =< \mathcal{L}, \mathtt{D}, \mathcal{I}, \models> \tag{8.21}$$

is a **logic**.

Observation 8.28 (Logic) In Equation (8.20) the components of a logic, i.e., $\mathtt{D}$, $\mathcal{L}$, $\mathcal{I}$, $\models$, define the general rules which are followed when defining mechanisms for representing and reasoning about world models, i.e., representations. They are defined a priori, extending world models, usually by experts in knowledge representation and logic, as a general tools to be used by practitioners. They provide the general modeling and reasoning infrastructure which allows to represent and reason about representations of real world problems. They also provide the mechanisms by which any two representations can be compared and possibly even merged. Software practitioners usually study these models during some CS or AI classes and they use them *as is* when developing systems; think for instance of the large usage of ER and UML models, and also of logics.

Observation 8.29 (Logic and World model) If $\hat{\mathcal{L}} =< \mathcal{L}, \mathtt{D}, \mathcal{I}, \models>$ and $\mathcal{L} = \mathcal{L}_A$ then $\hat{\mathcal{L}} = \hat{\mathcal{W}}$ with $\hat{\mathcal{W}} =< \mathcal{L}_A, \mathtt{D}, \mathcal{I}_A >$. In fact the entailment relation collapses into the interpretation function, see Equation (8.11).

Observation 8.30 (From mental representations to world models to logics) World models provide general mechanisms for generating non-ambiguous representations, i.e., theories of mental representations. Logics provide the mechanisms for reasoning about theories and the models they describe.

Observation 8.31 (Defining a model via a theory) The most common way to model the world is by defining a set of assertions, what we call a theory. In other words, we construct a model $\mathtt{M}$ by selecting any subset $\mathcal{T}$ of $\mathcal{L}$. This is the common approach when the task is that of representing from scratch a given part of the world which is of interest.

However, sometimes, one is given a predefined theory $\mathcal{T}$ and a predefined model $\mathtt{M}$ and is asked how they relate. In which case we have the following.

Definition 8.10 (Correctness and completeness of a theory $\mathcal{T}$ with respect to a model $\mathtt{M}$) Let $\hat{\mathcal{L}} =< \mathcal{L}, \mathtt{D}, \mathcal{I}, \models>$ be a logic. Let $\mathcal{T} \subseteq \mathcal{L}$ and $\mathtt{M} \subseteq \mathtt{D}$ be a theory and a model, respectively. Then we have two possible situations, as follows

- **Correctness**: $\mathtt{M} \models \mathcal{T}$, in which case we say that $\mathcal{T}$ is **correct** with respect to $\mathtt{M}$, or that $\mathtt{M}$ **is a model** for T;
- **Completeness**: If, for all facts $\mathtt{f} \in \mathtt{M}$ there is a formula $w \in \mathcal{L}_a$ such that $\mathcal{I}(w) = \mathtt{f}$, in which case we say that $\mathcal{T}$ is **complete** with respect to $\mathtt{M}$

The notions of **incorrectness** and **incompleteness** are defined in the obvious way

Chapter 9
Logic - intensional representation

We need to provide the specifics of how to compute entailment, starting from an account of how $\mathcal{L}$ and $\mathcal{I}$ are constructed

9.1 Domain

The goal is to have a more expressive language, if compared to that of a world model, for describing a domain of interpretation, leaving the latter unchanged. See Section 6.1.

9.2 Representation language

Definition 9.1 (Language, intensional representation) Let $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$ be a language. Then its **intensional representation** is

$$\mathcal{L}^i =< \mathcal{L}^i_a, \mathcal{L}^i_c > \tag{9.1}$$

where $\mathcal{L}^i_a$ is the **language of atomic formulas, intensionally defined** and $\mathcal{L}^i_c$ is the **language of complex formulas, intensionally defined**.

$\mathcal{L}^i_a$ and $\mathcal{L}^i_c$ are also pairs.

Definition 9.2 (Language of atomic formulas, intensional representation) Let $\mathcal{L}^i =< \mathcal{L}^i_a, \mathcal{L}^i_c >$ be a language intensionally defined. Then the **intensional representation** of $\mathcal{L}^i_a$ is

$$\mathcal{L}^i_a =< \mathcal{A}_a, \{\mathcal{FR}\}_a > \tag{9.2}$$

where: $\mathcal{A}_a$ is the **alphabet** of $\mathcal{L}_a$ and $\{\mathcal{FR}\}_a$ is the set of **formation rules for $\mathcal{L}^e_a$** with

$$\mathcal{A}_a =< \mathcal{E}, \{\mathcal{C}\}, \{\mathcal{R}\} >$$

$$\mathcal{L}^e_a = \{w : w \in \mathrm{C}(\{\mathcal{FR}\}_a, \mathcal{A}_a)\}$$

where: $\mathcal{E}$ is a set of **(names of) entities**, $\{\mathcal{C}\}$ is a set of **concepts**, where a concept is a **name of a class**, $\{\mathcal{P}\}$ and a set of **properties**, where a property is a **name of a relation** and, finally, $\mathrm{C}(\{\mathcal{FR}\}_a, \mathcal{A}_a)$ is the transitive closure of $\{\mathcal{FR}\}_a$ on $\mathcal{A}_a$.

Definition 9.3 (Formation rule) We restrict ourselves to languages with context-free grammars. Accordingly, we take $\{\mathcal{FR}\}_a = \{\mathcal{R}_a\}$, where each formation rule $\mathcal{R}_a$ has form

```
<expression> ::= -expression-
```

where, following `BNF` notation[1]:

- `<expression>` is a *nonterminal* expression. Nonterminals are enclosed within <>;
- Symbols that do not appear on the left side of a rule are called *terminals*;
- `-expression-` consists of one or more sequences of either terminal or nonterminal symbols;
- `::=` allows for `<expression>` to be replaced with a sequence occurring in `-expression-`;
- Sequences in `-expression-` are separated by the bar "|", indicating choice in the substitution.

Observation 9.1 (Formation rule) A generic formation rule $\mathcal{R}_a$ can be visualized as

$$\texttt{<NT> ::= <NT}_1\texttt{> | ... | <NT}_l\texttt{> | T}_1\texttt{ | ... | T}_m$$

with the possibility of no occurrences of terminal or nonterminal sequences.

Observation 9.2 (Transitive closure) $\mathrm{C}(\{\mathcal{FR}\}_c, \mathcal{L}_a)$ is the minimal set of formulas which can be obtained by recursively applying the rules of $\{\mathcal{FR}\}_c$ to their own results, starting from $\mathcal{L}_a$. Atomic formulas are black boxes for $\{\mathcal{FR}\}_c$ in the sense that the rules in $\{\mathcal{FR}\}_c$ can compose them into complex formulas but cannot change their internal structure.

Observation 9.3 (Infinity of the transitive closure) Formation rules, as from Definition 9.3, are recursive, see also Observation 9.1 in the sense that there are might be noterminals which appear both on the left and on the right of a production rules. This opens up the possibility of indefinitely long elements. That is the set $\mathrm{C}(\{\mathcal{FR}\}_c, \mathcal{L}_a)$ in the case of recursive production rules, as it is most often the case, is infinite.

Observation 9.4 (Alphabet) The alphabet $\mathcal{A}_a$ is infinite.

Example 9.1 (Language of atomic formulas, knowledge generation rules) Consider the language defined in Example 8.2 which allows for atomic complex formulas of shape $\mathcal{C}_1 \sqcap \mathcal{C}_2$ where $\mathcal{C}_i$ is a concept. The `BNF` generating this language consists of the following two formation rules:

[1] See `https://it.wikipedia.org/wiki/Backus-Naur_Form` for details.

```
<awff> ::=      <concept>
<awff> ::= <awff> ⊓ <awff>
```

where `<concept>` is non-terminal symbol which stands for any element $\mathcal{C}$ of the alphabet. ⊓ is a terminal symbol which, as such, cannot be further decomposed.

Example 9.2 (Language of atomic formulas, data generation rules) !!!!!!!TOBEDONE!!!!!!!!!!!!!!!!

Observation 9.5 (Application of formation rules, generation, recognition) Formation rules can be applied for two reasons. The first, as it is the case in Example 9.1 and Example 9.2, is in the process of **generation** of the closure; the second is in the process of **recognition** of whether e certain input formula belongs to the closure. The two sets of formation rules, while capture the same intuition, work in opposite directions, the first starting from terminals and developing more and more complex elements of the closure (generating, in the case of recursive production rules an infinite set), the second decomposing the input, complex, input down to the terminal elements. Failure in this process means that the input does not belong to the closure. This process will be used below in the definition of the interpretation function and the entailment relation.

Example 9.3 (Formation rules, generation and recognition) The following two formation rules for closure generation

```
<awff> ::=      <concept>
<awff> ::= <awff> ⊓ <awff>
```

can be rewritten as the following recognition rules for the same closure

$$\begin{array}{ll} \mathcal{W}(\texttt{<awff>}) & ::= \mathcal{W}(\texttt{<concept>}) \\ \mathcal{W}(\texttt{<awff1>} \sqcap \texttt{<awff2>}) & ::= \mathcal{W}(\texttt{<awff1>}) \;\; \text{and} \;\; \mathcal{W}(\texttt{<awff1>}) \end{array}$$

$\mathcal{W}$ checks the well-formedness of the input formula. $\mathcal{W}$ returns true is the recognition process terminates properly, false otherwise.

Definition 9.4 (Language of complex formulas, formation rules) Let $\mathcal{L}^i =< \mathcal{L}^i_a, \mathcal{L}^i_c >$ be a language intensionally defined. Then the **intensional representation** of $\mathcal{L}^i_c$ is

$$\mathcal{L}^i_c =< \mathcal{L}^e_a, \{\mathcal{FR}\}_c > \tag{9.3}$$

where $\{\mathcal{FR}\}_c$ is the set of **formation rules for** $\mathcal{L}^e_c$ with

$$\mathcal{L}^e_c = \{w : w \in \mathrm{C}(\{\mathcal{FR}\}_c, \mathcal{L}^e_a)\}$$

where: $\mathcal{L}^e_a$ is as from Definition 9.2 and, $\mathrm{C}(\{\mathcal{FR}\}_c, \mathcal{L}^e_a)$ is the transitive closure of $\{\mathcal{FR}\}_c$ on $\mathcal{L}_a$.

Example 9.4 (Language of complex formulas, intensional representation) Consider the language defined in Example 8.6 which allows for atomic complex formulas of shape A_1 $\underline{\texttt{xor}}$ A_2 where A_i is any formula. The BNF generating this language consists of the following two formation rules:

<wff> ::= <awff>
<wff> ::= <wff> $\underline{\texttt{xor}}$ <wff>

where <awff> is a non-terminal symbol which can be grounded into any atomic formula in $\mathcal{L}_a$.

9.3 Interpretation function

Let us proceed with the intensional definition of the interpretation function.

Definition 9.5 (Interpretation function, intensional representation) Let $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$ be a language with $\mathcal{L}_a = \mathcal{L}_A \cup \mathcal{L}_{AC}$. Let the interpretation function $\mathcal{I} : \mathcal{L}_a \rightarrow \texttt{D}$ be defined as $\mathcal{I} = \mathcal{I}_A \circ \mathcal{I}_{AC}$, with $\mathcal{I}_{AC} : \mathcal{L}_a \rightarrow \mathcal{L}_A$ and $\mathcal{I}_A : \mathcal{L}_A \rightarrow \texttt{D}$ (Definition 8.4). Then, the **intensional representation** of $\mathcal{I}$ is

$$\mathcal{I}^i =< \mathcal{L}_a, \{\mathcal{FR}\}_\mathcal{I} > \tag{9.4}$$

where $\{\mathcal{FR}\}_\mathcal{I}$ is the set of **formation rules for** $\mathcal{I}^e$ with

$$\mathcal{I}^e = \{< w, \texttt{f} >: w \in \mathcal{L}_a, \texttt{f} \in \texttt{D}, < w, \texttt{f} >\in \mathrm{C}(\{\mathcal{FR}\}_\mathcal{I}, \mathcal{L}_a)\}$$

where $\mathrm{C}(\{\mathcal{FR}\}_\mathcal{I}, \mathcal{L}_a)$ is the transitive closure of $\{\mathcal{FR}\}_\mathcal{I}$ over $\mathcal{L}_a$.

Observation 9.6 (Interpretation function, recognition rule) The recognition rules for the interpretation function $\mathcal{R}_\mathcal{I} \in \{\mathcal{FR}\}_\mathcal{I}$ are defined as follows:

$$\begin{array}{lll} \mathcal{I}_{AC}(\texttt{T}) & ::= & \mathcal{I}_A(\texttt{<T>}) \\ \mathcal{I}_{AC}(\texttt{<NT>}) & ::= & \mathcal{I}_{AC}(\texttt{<NT}_1\texttt{ >}) \mid \ldots \mid \mathcal{I}_{AC}(\texttt{<NT}_1\texttt{>}) \end{array} \tag{9.5}$$

where <NT> are nonterminal symbols for $\mathcal{I}_{AC}$, T are terminal symbols for $\mathcal{I}_{AC}$, and <T> are nonterminal symbols for $\mathcal{I}_A$.

Observation 9.7 (Interpretation function, recognition rules) In the recognition rules of the interpretation function, $\mathcal{I}_{AC}$ applies to nonterminals, while $\mathcal{I}_A$ applies to terminals of $\mathcal{I}_{AC}$, which are actual nonterminals for $\mathcal{I}_A$. To this extent, see the first production rule in Equation (9.5), where the same expression T gets transformed from terminal of $\mathcal{I}_{AC}$ to nonterminal of $\mathcal{I}_A$.

Example 9.5 (Interpretation function, recognition rules, data operators)
!!!!!!!TODO!!!!!!!!!!!!!

Example 9.6 (Atomic formulas, recognition rules, knowledge operators) Consider the set of formulas defined in Example 8.2, which are of the form $\mathcal{C}_1 \sqcap \mathcal{C}_2$, where $\mathcal{C}_i$ can be an atomic assertion as well as a complex assertion. Consider the formation rules generating them. We formalize the intuition that $\mathcal{C}_1 \sqcap \mathcal{C}_2$ denotes the intersection of the interpretations of two atomic assertions as follows:

$$\begin{array}{ll} \mathcal{I}_{AC}(\texttt{<concept>}) & \texttt{::=}\,\mathcal{I}_A(\texttt{<concept>}) \\ \mathcal{I}_{AC}(\texttt{<awff>} \sqcap \texttt{<awff>}) & \texttt{::=}\,\mathcal{I}_{AC}(\texttt{<awff>}) \cap \mathcal{I}_{AC}(\texttt{<awff>}) \end{array}$$

which can be used to implemented the following sequence of rewrites

$$\mathcal{I}(\mathcal{C}_1 \sqcap \mathcal{C}_2) = \mathcal{I}_{AC}(\mathcal{C}_1 \sqcap \mathcal{C}_2) = \mathcal{I}_A(\mathcal{C}_1) \cap \mathcal{I}_A(\mathcal{C}_2) = \mathsf{C}_1 \cap \mathsf{C}_2$$

Thus, for instance,

$$\begin{array}{ll} \mathcal{I}((\text{person} \sqcap \text{woman}) \sqcap \text{dog}) & = \\ \mathcal{I}_{AC}((\text{person} \sqcap \text{woman}) \sqcap \text{dog}) & = \\ \mathcal{I}_{AC}(\text{person} \sqcap \text{woman}) \cap \mathcal{I}_A(\text{dog}) & = \\ \mathcal{I}_A(\text{person}) \cap \mathcal{I}_A(\text{woman})) \cap \texttt{dog} & = \\ (\texttt{person} \cap \texttt{woman}) \cap \texttt{dog} & = \\ \texttt{woman} \cap \texttt{dog} & = \emptyset \end{array}$$

where we have assumed to know that women are persons and dogs are disjoint from persons.

Observation 9.8 (Application of the interpretation formation rules) Following up on what mentioned in Observation 6.11, Equation (6.7) shows how $\mathcal{I}$ is applied recursively by applying it to the components of its input assertion till assertions. In this process its components are applied as needed, that is, $\mathcal{I}_e$ to entities, $\mathcal{I}_C$ to concepts and $\mathcal{I}_P$ to properties. Example 9.6 shows (second line) how the process mentioned in Observation 6.11 generalizes to complex atomic formulas. The general idea is that $\mathcal{I}_{AC}$ gets applied till it gets to the single elements of the alphabet where, then, the proper component interpretation function is applied.

Observation 9.9 (Nesting of formation rules) The process highlighted above can be nested at any level. In fact $\mathcal{I}_A$ could be again a set of formation rules allowing for more refined complex atomic rules and so on, for any level of nesting. This allows for the generation of more and more complex manipulations of world models.

Example 9.7 (Atomic formulas, formation rules, data operators) !!DA FARE!! FUNZIONI ESEMPIO CON "friendOf" e "nearTo"

9.4 Entailment

Let us proceed with the intensional definition of the entailment relation.

Definition 9.6 (Entailment relation, intensional representation) Let $\texttt{M} \subseteq \texttt{D}$ be a model and $\mathcal{T} \subseteq \mathcal{L}$ a theory. Let the entailment relation be defined as

$$\models\ \subseteq\ \texttt{M} \times \mathcal{T}$$

as from Definition 8.5. Then, the **intensional representation** of $\models$ is

$$\models^i = < \mathcal{L}, \mathtt{D}, \{\mathcal{FR}\}_{\models} > \tag{9.6}$$

where $\{\mathcal{FR}\}_{\models}$ is the set of **formation rules of** $\models^e$, with

$$\models^e \ = \{< \mathtt{f}, w >: \mathtt{f} \in \mathtt{M}, w \in \mathcal{L}, < \mathtt{f}, w > \in \mathrm{C}(\{\mathcal{FR}\}_{\models}, \mathtt{D}, \mathcal{L})\}$$

where $\mathrm{C}(\{\mathcal{FR}\}_{\models}, \mathcal{L})$ is the transitive closure of $\{\mathcal{FR}\}_{\models}$ over $\mathcal{L}$.

Definition 9.7 (Entailment relation, formation rule) A formation rule for the entailment relation $\mathcal{R}_{\models} \in \{\mathcal{FR}\}_{\models}$ is defined as:

$$\begin{array}{lll} \mathtt{M} \models \mathtt{T} & ::= & \mathcal{I}(\texttt{<T>}) \\ \mathtt{M} \models \texttt{<NT>} & ::= & \mathtt{M} \models \texttt{<NT}_1\texttt{ >} \ \mid \ldots \mid \mathtt{M} \models (\texttt{<NT}_1\texttt{>} \end{array} \tag{9.7}$$

where <NT> are nonterminal symbols for $\models$, T are terminal symbols for $\models$, and <T> are nonterminal symbols for $\mathcal{I}$.

Observation 9.10 (Entailment relation, formation rules) Read Observation 9.7. The same considerations apply, *mutatis mutandis*.

Example 9.8 (Entailment) Take $\mathcal{L}_a, \mathtt{D}, \mathcal{I}$ as from Example 9.5. Take $\mathcal{L} = \mathcal{L}_a \cup \mathcal{L}_c$ to be the representation language defined in 9.4 (with $\mathcal{L}_a$ being the same as in Example 9.5), where the formation rules for $\mathcal{L}_c$, are as follows (as from Example 9.4 reported here for completeness):

$$\begin{array}{l} \texttt{<wff>} ::= \texttt{<awff>} \\ \texttt{<wff>} ::= \texttt{<wff>} \ \underline{\texttt{xor}} \ \texttt{<wff>} \end{array}$$

Then we define the entailment relation with the following recognition rules:

$$\begin{array}{lll} \mathtt{M} \models \texttt{<wff>} & ::= & \mathcal{I}(\texttt{<awff>}) \\ \mathtt{M} \models \texttt{<wff1>} \ \underline{\texttt{xor}} \ \texttt{<wff2>} & ::= & \mathtt{M} \models \texttt{<wff1>} \text{ and } \mathtt{M} \not\models \texttt{<wff2>} \mid \\ & & \mathtt{M} \not\models \texttt{<wff1>} \text{ and } \mathtt{M} \models \texttt{<wff2>} \end{array}$$

We have the following examples (where a, a_i are atomic formulas).

$$\begin{array}{ll} \mathtt{M} \models a & \text{if } \mathcal{I}(a) \in \mathtt{M} \\ \mathtt{M} \not\models a & \text{if } \mathcal{I}(a) \notin \mathtt{M} \\ \mathtt{M} \models a_1 \ \underline{\texttt{xor}} \ a_2 & \text{if } \mathcal{I}(a_1) \in \mathtt{M} \text{ and } \mathcal{I}(a_2) \notin \mathtt{M} \ \mid \ \mathcal{I}(a_1) \in \mathtt{M} \text{ and } \mathcal{I}(a_2) \notin \mathtt{M} \\ \mathtt{M} \models \{w_1, w_2\} & \text{if } \mathtt{M} \models w_1 \text{ and } \mathtt{M} \models w_2 \end{array}$$

Observation 9.11 (Theory) The notion of theory, as from Definition 8.2, and its characterization as a set of axioms, as from Terminology 8.3 does not capture the idea of a theory consisting all theorems, that is all the statements which are decided to hold as part of the closure over the entailment formation rules.

Definition 9.8 (Theory, finite presentation) A **theory** is a set of formulas closed under the logical consequence, that is $\mathcal{T}$ is a theory if and only if $\mathcal{T} \models w$ implies that $w \in \mathcal{T}$. The set of axioms $\mathcal{T}_0$ from which $\mathcal{T}$ is generated via entailment is called a **finite presentation of** $\mathcal{T}$.

The two notions of theory are used interchangeably. But not all theories can be generated from a finite set of axioms.

Definition 9.9 (Theory, finite axiomatization) A theory T is **finitely axiomatizable** if it can be generated from a finite set of axioms.

9.5 Logic

Let us see how to use logic in practice, and how they extend what can be done with world models, as from Section 6.5.

Definition 9.10 (Logic, intensional representation) Let $\hat{\mathcal{W}}^i =< \mathcal{L}^i_A, \mathtt{D}^i, \mathcal{I}^i_A >$ be a world model with $\mathtt{D}^i =< \mathtt{E}, \{\mathtt{C}\}, \{\mathtt{R}\} >$. Then, let

$$\hat{\mathcal{L}} =< \mathcal{L}, \mathtt{D}, \mathcal{I}, \models >,$$

be a logic defined for the same domain of interpretation $\mathtt{D}^i$ of $\hat{\mathcal{W}}^i$. Then, the **intensional representation** $\hat{\mathcal{L}}^i$ of $\hat{\mathcal{L}}$ is defined as:

$$\hat{\mathcal{L}}^i =< \mathcal{L}^i, \mathtt{D}^i, \mathcal{I}^i, \models^i >, \quad \text{with} \quad \mathcal{L}^i =< \mathcal{L}^i_a, \mathcal{L}^i_c > \tag{9.8}$$

and

$$\begin{array}{lll}
\mathcal{L}^i_a & = < \mathcal{A}_a & , \{\mathcal{FR}\}_a > \\
\mathcal{L}^i_c & = < \mathcal{L}^e_a & , \{\mathcal{FR}\}_c > \\
\mathcal{I}^i & = < \mathcal{L}^e_a & , \{\mathcal{FR}\}_\mathcal{I} > \\
\models^i & = < \mathcal{L}^e & , \{\mathcal{FR}\}_\models >
\end{array}$$

$\hat{\mathcal{L}}^i, \mathcal{L}^i_a, \mathcal{L}^i_c, \mathcal{I}^i, \models^i$ are the **stencil** of the logic $\hat{\mathcal{L}}$, of the language of atomic formulas $\mathcal{L}_a$, of the language of complex formulas $\mathcal{L}_c$, of the interpretation function $\mathcal{I}$ and of the entailment relation $\models$, respectively.

Terminology 9.1 (Data, knowledge, mixed) We talk of **data, knowledge** and **mixed logics, theories** and **models** with the obvious meaning.

Observation 9.12 (Logic, design constraints and choices) The Logic stencil $\hat{\mathcal{L}}^i$ is all we need in order to implement a logic and use it to perform reasoning. But there are constraints to be kept in mind, in particular, those concerning the interpretation function and also the entailment relation. There are also important choices to be made, in particular about which world model and logic should be selected, how to implement them, and how to integrate them.

Observation 9.13 (Defining a logic) The definition of a logic is articulated in the following steps:

1. **Define** $\mathcal{A}_a$. This provides the language for making assertions about the facts in the world. In the following $\mathcal{A}_a$ will be defined in terms of a $\hat{\mathcal{KG}}$.
2. **Define** $\mathcal{L}^i_a$ and, specifically, $\{\mathcal{FR}\}_a$. This provides the language for producing complex descriptions of the facts in the world.
3. **Validate the definition of** $\mathcal{A}_a$ **and** $\mathcal{L}^i_a$. This is done via the definition of $\mathcal{I}^i$ where the goal is to make sure that the notation used $\mathcal{A}_a$ and $\mathcal{L}^i_a$ is intuitive and representative of the fact of the world and that all the constraints on $\mathcal{I}^i$ are satisfied.
4. **Define** $\mathcal{L}^i_c$. This provides the language for building complex descriptions about the facts in the world, as represented in the $\hat{\mathcal{KG}}$. This is usually done using a logical language which formalizes some targeted fraction of natural language.
5. **Define** $\models^i$, namely the rules for reasoning, based on the definition of $\mathcal{L}^i_c$. In particular the key idea is that the definition of entailment is compositional, following the compositionality of the generation rules for $\mathcal{L}^i_c$. !!THIS POINT MUST BE MADE MORE GLOBAL ... CHECK!!

Given a logic, one can use it to reason about the world. This is the topic of the next section.

9.6 Using a logic

Logics allow to do complex reasoning about world models. Let us see how this can be implemented in practice.

Intuition 9.1 (Solving problems using logic) This is the quite the same as with world model. Seen from the outside, the user will see no difference, only an increase in reasoning power. See Intuition 6.6

Terminology 9.2 (Language of the world model, language of reasoning) In the following we assume that $\mathcal{L}_Q = \mathcal{L}_A = \mathcal{L}_\mathcal{W}$, where $\mathcal{L}_\mathcal{W}$ is the representation **language of the world model**, and that $\mathcal{L}_T = \mathcal{L}_\mathcal{R}$, where $\mathcal{L}_\mathcal{R}$ is the representation **language of reasoning**. Notice that, compared to the Terminology 6.3, we assume the existence of $\mathcal{L}_\mathcal{R}$ and that $\mathcal{L}_\mathcal{W} \neq \mathcal{L}_\mathcal{R}$. This is motivated by the fact that the task of representing the world in the world model and that of reasoning about the world are usually implemented by two independent systems with a third system explicitly dedicated to bridging the two systems and translating $\mathcal{L}_\mathcal{W}$ into $\mathcal{L}_\mathcal{R}$. See also Intuition 9.3

Intuition 9.2 (Logics, reasoning problems) But which questions and which answers? All logics, independently of the specifics of their definition, as from Definition 9.10, provide answers to six foundational questions that we state below as **reasoning problems**. Let us assume that we have the stencil $\hat{\mathcal{L}}^i =< \mathcal{L}^i, \mathtt{D}^i, \mathcal{I}^i, \models^i>$ and that we have a set $\{\mathtt{M}\}$ of models with $\mathtt{M} \subseteq \mathtt{D}$ and a set theories $\mathcal{T} \subseteq \mathcal{L}_A$. Then we have the following:

Reasoning Problem 9.1 (Model checking) Given $\mathcal{T}$ and $\mathtt{M}$, check whether $\mathtt{M} \models \mathcal{T}$.

Reasoning Problem 9.2 (Satisfiability) Given $\mathcal{T}$, check whether there exists $\mathtt{M}$ such that $\mathtt{M} \models \mathcal{T}$.

Reasoning Problem 9.3 (Validity) Given $\mathcal{T}$, check whether for all $\mathtt{M}$, $\mathtt{M} \models \mathcal{T}$.

Reasoning Problem 9.4 (Unsatisfiability) Given $\mathcal{T}$, check whether there is no $\mathtt{M}$ such that $\mathtt{M} \models \mathcal{T}$.

Reasoning Problem 9.5 (Logical consequence) Given $\mathcal{T}_1$, $\mathcal{T}_2$ and a set of reference models $\{\mathtt{M}\}$, check whether $T_1 \models_{\{\mathtt{M}\}} T_2$.

Reasoning Problem 9.6 (Logical equivalence) Given $\mathcal{T}_1$, $\mathcal{T}_2$ and a set of reference models $\{\mathtt{M}\}$, check whether $T_1 \models_{\{\mathtt{M}\}} T_2$ and $T_2 \models_{\{\mathtt{M}\}} T_1$.

The validity, satisfiability and unsatisfibiality of a formula are correlated as follows:

Proposition 9.1 *If a formula is valid, then it is also satisfiable, and it is also not unsatisfiable. That is:*

$$\textit{Validity} \supset \textit{Satisfiability} \supset \textit{not Unsatisfiability}$$

Proposition 9.2 *If a formula is unsatisfiable, then it is also not satisfiable, and also not valid. That is:*

$$\textit{Unsatisfiability} \supset \textit{not Satisfiable} \supset \textit{not not Valid}$$

Observation 9.14 (logics vs world models) World models (see Intuition 6.7) feature the first four reasoning problems of logic, stated exactly in the same way. The key and fundamental difference is that in world model entailment reduces to set inclusion, that is, to checking whether the interpretation of a formula belongs to a model (see Definition 6.11). Though much simpler than logical reasoning, it provides the baseline on top of which logical reasoning is implemented.

Observation 9.15 (Reasoning problems, logic dependence) As the following sections will make clear, different logics feature different and specific instances of the reasoning problems defined in Intuition 9.2. This is in fact the main reason why there are multiple world models. Despite the fact that they all solve the same six foundational problems, they do it in specific contexts and problem spaces for which they are tuned. Showing and discussing these specifics will be the topic of the following sections.

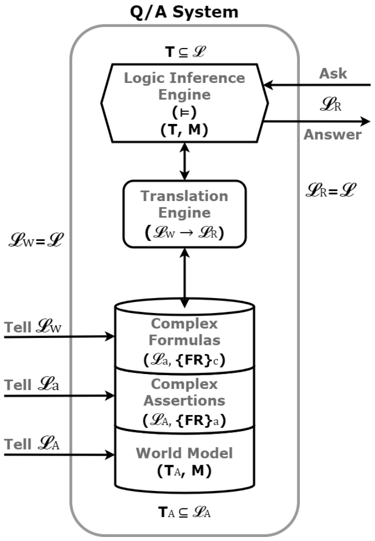

Q/A System
$\mathbf{T} \subseteq \mathscr{L}$
Logic Inference Engine
$(\models)$
(T, M)
Ask
$\mathscr{L}_R$
Answer
$\mathscr{L}_R = \mathscr{L}$
Translation Engine
$(\mathscr{L}_W \rightarrow \mathscr{L}_R)$
$\mathscr{L}_W = \mathscr{L}$
Tell $\mathscr{L}_W$
Complex Formulas
$(\mathscr{L}_a, \{\mathbf{FR}\}_c)$
Tell $\mathscr{L}_a$
Complex Assertions
$(\mathscr{L}_A, \{\mathbf{FR}\}_a)$
Tell $\mathscr{L}_A$
World Model
$(\mathbf{T}_A, \mathbf{M})$
$\mathbf{T}_A \subseteq \mathscr{L}_A$

Fig. 9.1 Solving problems using logics.

Intuition 9.3 (An architecture for solving problems using logic) The architecture supporting the use of logic, as specified in Intuition 9.1, is depicted in Figure 9.1. We can identified three main components, as follows

- A **world model** which encodes the available data and knowledge about the world and allows for minimal reasoning about them (see the reasoning problems in Intuition 6.7);
- An **logic inference engine** implementing one or more of the reasoning problems defined in Intuition 6.7;
- A representation **language translation engine** implementing the bidirectional translation between $\mathcal{L}_{\mathcal{W}}$ and $\mathcal{L}_{\mathcal{R}}$.

Some observations. The world model as well as the reasoners are selected at the beginning, before the system is put in operation. The choice of which wold model stencil and which logic stencil depends on the specifics of the problem be solved. Observation 9.18 below describes the issues and trade-offs which must are taken into account when performing the choice of the logic stencil.

Observation 9.16 (Decision, procedure, logic inference engine) A **decision procedure** is an algorithm which, for a certain logic, solves one of the six foundational problems specified in Intuition 6.7. An inference engine is usually a decision procedure with two added features:

- it may solve more than one reasoning problems, exploiting the fact that, in certain logics (not all), the solution of a reasoning problem can be reduced to the solution of another reasoning problem;

- it implements various **heuristics** whose main goal is to speed-up the computation time.

Observation 9.17 (Representation language translation engine) As it will be discussed in detail the work on inference engines is most advanced for certain logics where there exist very fast, off-the-shelf- open source implementations. In these cases the translation engine is used to rewrite a reasoning problem stated in a source world model or logic to a reasonig problem stated in the target logic.

Intuition 9.4 (The process of using logic) Logics are used according to the `Tell`, `Ask`, `Reason`, `Answer` pattern defined in Intuition 9.1. However, prior to that, the system depicted in Figure 9.1 must be specified (and implemented). The specification pattern for this to happen is as follows

1. Selection of the world model $\mathcal{W}$ and specifically the **representation Language** $\mathcal{L}_{\mathcal{W}}$ and corresponding interpretation function. This, in turn, amounts to the following three steps:
   - Selection of the **assertional language** $\mathcal{L}_A$ (and interpretation function) used to specify the domain of interpretation;
   - Selection of the **language of atomic formulas** $\mathcal{L}_a$ (and interpretation function) used to specify atomic formulas and the corresponding set of formation rules $\mathcal{R}_a$;
   - Selection of the **representation language** $\mathcal{L}$ used to specify formulas and the corresponding set of formation rules $\mathcal{R}_a$;
2. Selection of the **entailment relation** $\models$, which amounts to the following two steps:
   - Selection of the reasoning **representation language** $\mathcal{L}_{\mathcal{R}}$;
   - Selection of the **Interpretation function** for $\mathcal{L}_{\mathcal{R}}$;
   - Selection of the **consequence relation** $\models_{\mathcal{L}_{\mathcal{R}}}$ used to implement reasoning and the corresponding reasoning problems;
   - Selection of the **inference engine** used to solve the target reasoning problems.
3. Selection of the **translation procedure** used to translate assertions $a \in \mathcal{L}_a$ describing the facts stored in the world model to the formulas $w \in \mathcal{L}$ of the language of the reasoner.
4. Selection of the **inference engine**.

Observation 9.18 (Logic, selection trade-offs) Any logics can be characterized by two main parameters:

- **Expressivity**, that is, the level of detail at which the problem is expressed, depending on the syntax of the language of the logic;
- **Computational efficiency**, that is how much it costs, in terms of space and time, to reason and answer queries in that language.

More expressivity allows for a more refined and precise modeling of the problem but it also generates longer and more complicated formulas. There is a crucial trade-off in that, the more expressive a logic is, the less computationally tractable it is. The modeler must find the right trade-off between *expressiveness* and *computational complexity*. Here the choice of the representation language $\mathcal{L} =< \mathcal{L}_a, \mathcal{L}_c >$ is crucial. The computational complexity of both $\mathcal{L}_a$ and $\mathcal{L}_c$ ranges in fact from polynomial to exponential and beyond. There is also an issue of *(un)decidability*, namely the possibility for the reasoner, on certain queries, to get into an infinite loop, never terminate and, therefore, never return an answer. However this issue, while very relevant in mathematical logic and theory of computation, has no practical relevance here, the main reason being that in AI the domains of interpretation are (almost) always finite, and this guarantees termination, also in the case of $\mathcal{LOL}$, on of the most expressive logic that we consider.

9.7 Exercises

TODO

Part III
The logics of recognition

Use the template *part.tex* together with the document class SVMono (monograph-type books) or SVMult (edited books) to style your part title page and, if desired, a short introductory text (maximum one page) on its verso page.

Chapter 10
$\mathcal{LOE}$ A Logic of Entities

10.1 Logic

The **Logic of Entities ($\mathcal{LOE}$)**, as from its name, allows to describe and reason about entities also as a function of the main class to which they belong, and their properties. All entities are assumed to have a name which uniquely identifies them, which may be unknown. $\mathcal{LOE}$ is the simplest logic allowing to reason about $\hat{\mathcal{EG}}$'s. It is conceptually similar in spirit to an earlier logic called the Assertion Box (ABOX) of Description Logics (DL).

Definition 10.1 (Reminder Definition 7.1: $\hat{\mathcal{EG}}$) $\mathcal{LOE}$ support the following $\hat{\mathcal{EG}}$ fact elements:

- An **Entity** is anything to which we give a name;
- A **Concept** is the main class to which an entity belongs;
- A **(Data) Value** is anything which has a (predefined) name;
- A **Datatype** is a class of (data) values;
- A **Data Property**, also called **Attribute**, is a characteristic of an entity;
- An **Object Property** describes a relation between two entities.

Example 10.1 ($\hat{\mathcal{EG}}$) We have the following:

- Examples of entities are: a location of name Trento, an event of name "Festival dell'Economia, 2023", a person a name Fausto, a dog of name Fido, my car, the tree in front of my room;
- Examples of concepts are: location, event, person, dog, car, tree, room;
- Examples of values: the natural number 3, the real number 3.14..., the string *"Fausto"*, 82 as the weight of Fausto in kgs, the URL `http://knowdive.disi.unitn.it/`.
- Examples of datatypes are: Integer, Real, String, Weight in kilograms, Weight in grams, Distance in meters, Identifier;
- Examples of Data properties are: the name of Fausto, a unique identifier for Fausto, the age of Fausto, the weight of Fausto, the color of my car, the max speed of my car;

- Example of object properties are: Fausto is a friend of Fido, Fausto is the owner of his car, the tree in front of my room is in front of my room, Fausto is a friend of Mario.

Definition 10.2 ($\hat{\mathcal{EG}}$) An $\mathcal{EG}$ is a $\mathcal{KG}$ where:

- Nodes, which are of two types.
  - entity nodes which are labeled by entity names and their concept;
  - value nodes which are labeled by values and their datatype;
- edges, which are labeled by a data property name or by an object property name

Example 10.2 (Reminder Example 7.5: An entity Graph $\mathcal{EG}$) Consider the $\mathcal{EG}$ represented in Figure 7.3. This $\mathcal{EG}$ represents the following facts: *Paolo and Stefania are people. Max, Argo, Toby are dogs. Spain and Italy are places. 15/05/1990 is a date of birth. 1.78m and 1.85m are heights. Stefania has Paolo as her friend. Paolo's birthplace is Spain. Paolo has Max and Argo as his pet. Paolo has a height of 1.85m. Paolo's website has the Url http://www.paolo.it. Stefania's birthplace is Italy. Stefania's date of birth is 5/15/1990. Stefania's pet is Toby. Stefania's height is 1.78m.*

We formally define $\mathcal{LOE}$ as follows

Definition 10.3

$$\mathcal{LOE} = \langle \hat{\mathcal{EG}}, \models_{\mathcal{LOE}} \rangle \tag{10.1}$$

with

$$\hat{\mathcal{EG}} = \langle \mathcal{L}_{\mathcal{LOE}}, \mathtt{D}_{\mathcal{LOE}}, \mathcal{I}_{\mathcal{LOE}} \rangle \tag{10.2}$$

When no confusion arises, below we drop the subscripts.

10.2 Domain

The intensional definition $\mathtt{D}^i$ is as follows:

Definition 10.4 (Domain, its intensional definition $\mathtt{D}^i$)

$$\mathtt{D}^i \ = <\mathtt{E}, \{\mathtt{C}\}, \{\mathtt{R}\}> \tag{10.3}$$

where:

$$\begin{aligned} \mathtt{E} \ &= \ \{e\} \cup \{v\} \\ \{\mathtt{C}\} &= \ \ \mathtt{ET} \cup \mathtt{DT} \\ \{\mathtt{R}\} &= \{\mathtt{OR}\} \cup \{\mathtt{DR}\} \end{aligned}$$

where $\mathtt{ET} = \{\mathtt{E_T}\}$, $\mathtt{E_T} = \{e\} \subseteq \mathtt{E}$ and $\mathtt{DT} = \{\mathtt{D_T}\}$, $\mathtt{D_T} = \{v\} \subseteq \mathtt{E}$ are **sets of etypes** and **dtypes**, respectively, and $\mathtt{OR}, \mathtt{DR}$ are **(binary) object** and **data relations**.

Observation 10.1 (Concepts, etypes, dtypes) The set of etypes $\mathtt{ET}$ and dtypes $\mathtt{DT}$ are a priori defined, application dependent, organization of $\mathtt{E}$ (see the section on Knowledge Graphs for more details). Here we have defined them as part of $\{\mathtt{C}\}$ to make the definitions more streamlined. From a practical point of view, concepts, etypes, dtypes are all subsets of $\mathtt{E}$, and, as such, behave exactly in the same way.

Definition 10.5 (The facts $\{\mathtt{f}\}$, with $\{\mathtt{f}\} \in \mathtt{D}^e$) $\mathcal{LOE}$ allows for the following facts:

- Every etype or dtype applied to its argument is a fact.
- Every relation $\mathtt{R}$ applied to its two arguments is a fact.

Observation 10.2 (The form of facts) Facts only have one of four possible forms: $\mathtt{ET}(\mathtt{e})$, $\mathtt{DT}(\mathtt{v})$, $\mathtt{R}(\mathtt{e}_i, \mathtt{e}_j)$ and $\mathtt{R}(\mathtt{e}, \mathtt{v})$.

Example 10.3 (A domain $\mathtt{D}$ *of the* $\mathcal{EG}$ *of Example in Fig.7.3)*

$$\begin{aligned}
\mathtt{E} &= \{\#1, \#2, \#3, \#4, \#5, \#6, \#7, \#8, \#9, 1{,}85\text{m}, \\
&\quad\; 1{,}78\text{m}, 15/05/1990, \text{http://www.paolo.it}\} \\
\mathtt{ET} &= \{\text{P}, \text{D}, \text{L}\} \\
\mathtt{DT} &= \{\text{Real}, \text{Boolean}, \text{String}\} \\
\{\mathtt{R}\} &= \{\text{hF}, \text{hD}, \text{hH}, \text{hB}, \text{hL}, \text{hU}\}
\end{aligned}$$

Observation 10.3 (Alphabet and domain components) The cardinality of each of the components of $\mathtt{D}$ is bigger than or equal to the cardinality of the corresponding elements of $\mathcal{S}^e$. It is a fact that in most situations the language used to describe a domain does not have names for all the elements of the domain it describes.

10.3 Language

$\mathcal{LOE}$ allows only for assertions, so its language $\mathcal{L}^e$ is extensionally defined as follows.

Definition 10.6 (Language $\mathcal{L}^e$)

$$\mathcal{L}^e = \{\mathcal{L}^e_A \cup \emptyset\} \cup \emptyset \tag{10.4}$$

where $\mathcal{L}^e_A$ is a set of assertions.

$\mathcal{L}^e_A$ is constructed from an alphabet $\mathcal{A}_a$ defined as follows.

Observation 10.4 (Language $\mathcal{L}^e$) The language of $\mathcal{LOE}$ consists only of assertions. The language of complex assertions and the language of complex formulas are both empty.

Definition 10.7 (Alphabet $\mathcal{A}_a$)

$$\mathcal{A}_a =< \mathcal{E}, \{\mathcal{C}\}, \{\mathcal{P}\} > \tag{10.5}$$

where $\mathcal{E}$ is a set of **(names of) entities and values**, $\{\mathcal{C}\} = \mathcal{ET} \cup \mathcal{DT}$ is a set of **(names of) etypes** and of **dtypes**. $\{\mathcal{P}\}$ is a set of **properties**, also called **roles**, with $\{\mathcal{P}\} = \{\mathcal{OP}\} \cup \{\mathcal{DP}\}$, where $\mathcal{OP}$ is an **object property** and $\mathcal{DP}$ is a **data property**.

$\mathcal{L}^e_A$ is a set of assertions, constructed from the alphabet $\mathcal{A}_a$.

Definition 10.8 (BNF generation rules) The $\mathcal{L}^e_A$ BNF generation rules are as follows.

$$\begin{array}{lll}
\texttt{<awff>} & ::= & \texttt{<etype>(<nameEntity>)} \hfill | \\
 & & \texttt{<dtype>(<value>)} \hfill | \\
 & & \texttt{<objProp>(<nameEntity>, <nameEntity>)} \; | \\
 & & \texttt{<dataProp>(<nameEntity>, <value>)} \\
\texttt{<etype>} & ::= & \mathcal{ET}_1 \mid \ldots \mid \mathcal{ET}_n \\
\texttt{<dtype>} & ::= & \mathcal{DT}_1 \mid \ldots \mid \mathcal{DT}_n \\
\texttt{<objProp>} & ::= & \mathcal{OP}_1 \mid \ldots \mid \mathcal{OP}_n \\
\texttt{<objProp>} & ::= & \mathcal{DP}_1 \mid \ldots \mid \mathcal{DP}_n \\
\texttt{<nameEntity>} & ::= & e_1 \mid \ldots \mid e_n \\
\texttt{<value>} & ::= & v_1 \mid \ldots \mid v_n
\end{array}$$

with $e_i, v_i \in \mathcal{E}$, $\mathcal{ET}_i \in \mathcal{ET}$, $\mathcal{DT}_i \in \mathcal{DT}$, $\mathcal{OP}_i \in \{\mathcal{OP}\}$ and $\mathcal{DP}_i \in \{\mathcal{DP}\}$.

Observation 10.5 (The form of formulas) Assertions have one of four possible forms, as stated above: $\mathcal{ET}_i(e)$, $\mathcal{DT}_i(v)$, $\mathcal{OP}_k(e_i, e_j)$and $\mathcal{DP}_k(e, v)$.

Example 10.4 (An alphabet $\mathcal{A}_a$ and a language $\mathcal{S}^e$ of the $\mathcal{EG}$ of Example in Fig.7.3) The elements of $\mathcal{A}_a$:

$$\begin{array}{ll}
\mathcal{E} & = \{\text{Paolo, Sofia, Stefania, Argo, Max, Toby, Balto, Spain, Italy, Balto,} \\
 & \quad \text{1,85m, 1,78m, 15/05/1990, http://www.paolo.it}\} \\
\{\mathcal{P}\} & = \{\text{hasFriend, hasDog, hasHeight, hasBirthdate, hasBirthPlace} \\
 & \quad \text{Person, Dog, Place, Measurement, Date}\}
\end{array}$$

The sentences in $\mathcal{S}^e$:

$$\begin{array}{lll}
\mathcal{S}^e = \{ & \text{Person(Paolo),} & \text{hasDog(Paolo, Max),} \\
 & \text{hasHeight(Paolo, 1,85m),} & \text{hasDog(Paolo, Argo),} \\
 & \text{hasUrl(Paolo, http://www.paolo.it),} & \text{hasBirthplace(Paolo, Spain)} \\
 & \}
\end{array}$$

Observation 10.6 (Formalising the alphabet of an $\mathcal{EG}$) The alphabet of an $\mathcal{EG}$ names an entity or a value in each node, each associated to its etype or dtype, and one role per property. All the elements of the alphabet of the $\mathcal{EG}$ in in Fig.7.3 belong to $\mathcal{L}^e_A$ but not vice versa. It is also possible to use a language which does not contain all the elements of an $\mathcal{EG}$. This is usually done for large $\mathcal{EG}$'s where one is interested in modeling and reasoning about a subgraph of the $\mathcal{EG}$. This is a quite frequent situation given that the state-of-the art $\mathcal{EG}$'s can have millions of nodes and more. See also Observation 5.5.

Observation 10.7 (Formalizing the formula of an $\mathcal{EG}$) The same considerations made in Observation 10.6 apply to assertions.

Observation 10.8 (Scope of formulas compared to facts) Similarly to what was explained earlier, the collection of formulas within a language typically, and specifically in this context, does not encompass all the details within the domain. This occurs because languages used for description often focus on capturing key aspects and general patterns, and might not account for every specific detail or instance present in the domain. In other words, while the language provides a structured representation, it might not capture every single fact or nuance of the complex real-world domain it aims to represent.

10.4 Interpretation function

Definition 10.9 (Interpretation function $\mathcal{I}$)

$$\mathcal{I} =< \mathcal{I}_e, \mathcal{I}_\mathcal{C}, \mathcal{I}_\mathcal{P} > \tag{10.6}$$

For more details see Section 6.3.

Example 10.5 (An Interpretation function $\mathcal{I}$ of the $\mathcal{EG}$ of Example in Fig.7.3) We have the following

$$\begin{aligned}
\mathcal{I}(\text{Paolo}) &= \#1\\
\mathcal{I}(\text{hasFriend(Paolo, Sofia)}) &= \text{hf}(\#2, \#1)\\
\mathcal{I}(\text{Person(Stefania)}) &= \text{P}(\#3)\\
\mathcal{I}(\text{hasDog(Stefania, Toby)}) &= \text{hD}(\#3, \#6)\\
\mathcal{I}(\text{hasDog(Paolo, Argo)}) &= \text{hD}(\#1, \#4)\\
\mathcal{I}(\text{hasDog(Paolo, Max)}) &= \text{hD}(\#1, \#5)\\
&\ldots
\end{aligned}$$

Where for instance:

$$\mathcal{I}(\text{hasDog(Paolo, Argo)}) = \text{hasDog}^{\mathcal{I}}(\text{Paolo}^{\mathcal{I}}, \text{Argo}^{\mathcal{I}}) = \text{hD}(\#1, \#4)$$

10.5 Entailment relation

Since we have only assertions, the entailment relation collapses into the interpretation function.

Definition 10.10 (Entailment $\models$)

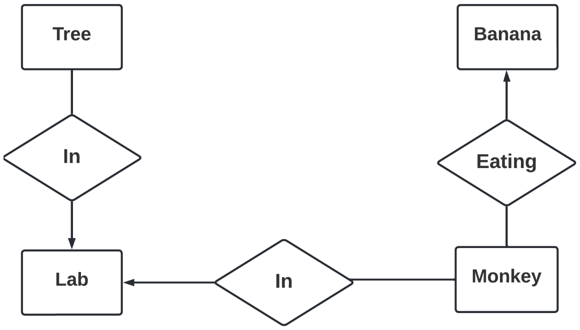
$$\mathtt{M} \models w \iff I(w) \in \mathtt{M} \tag{10.7}$$

with $w \in \mathcal{L}^e$

Solutions

Exercises of Chapter 5 & 6

Solution **6.1 (ER Creation).** We can create an ER diagram from the theory in this way:



[H]

Solution **6.2 (Complete and Correct?).** The model is **incorrect** respect to the theory.

Solution **6.3 (Complete and Correct?).** The model is **correct** respect to the theory.

Solution **6.4 (Complete and Correct?).** The model is **incomplete** respect to the theory.

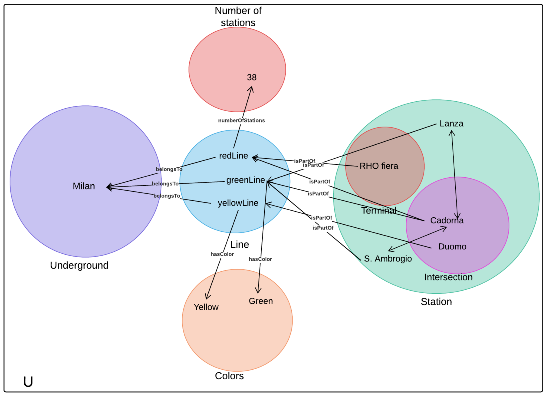
Solution **6.5 (Complete and Correct?).** The model is **complete** respect to the theory.

Solution **6.6 (Complete and Correct?).** The model is **correct and complete** respect to the theory.

Exercises of Chapter 7

Solution **7.1 (Create a set diagram).** Using set theory to represent the Milan subway we have the following diagram:

[H]



Exercises of Chapter 8 & 9

TODO

References

1. Lingelbach B., Block B., Hatzky B., and Reisinger E. The hermann grid illusion – retinal or cortical? *Perception*, 1985.
2. Stephen F. Barker. *The Elements of Logic - Chapter 6*. New York, NY: McGraw-Hill, sixth edition, 2003.
3. S. Morris Engel. *With Good Reason: An Introduction to Informal Fallacies*. New York: St. Martin's Press, 1982.
4. Robert M Entman. Framing: Toward clarification of a fractured paradigm. *Journal of communication*, 43(4):51–58, 1993.
5. Erving Goffman. *Frame analysis: An essay on the organization of experience.* Harvard University Press, 1974.
6. Daniel Kahneman and Amos Tversky. The psychology of preferences. *Scientific American*, 246(1):160–173, 1982.
7. Alfred Korzybski. *Science and Sanity. An Introduction to Non-Aristotelian Systems and General Semantics*. The International Non-Aristotelian Library Pub., 1933.
8. Elizabeth F Loftus and John C Palmer. Reconstruction of automobile destruction: An example of the interaction between language and memory. *Journal of verbal learning and verbal behavior*, 13(5):585–589, 1974.
9. Elliott Mendelson. *Introduction to Mathematical Logic*. Chapman & Hall, sixth edition, 2015.
10. Maarten de Rijke Patrick Blackburn. *Modal Logic*. Cambridge University Press, first edition, 2001.
11. Wolfgang Rautenberg. *A Concise Introduction to Mathematical Logic*. Springer New York, second edition, 2006.
12. Hoffrage U. *Cognitive Illusions: a handbook on fallacies and biases in thinking, judgement and memory*. Psychology Press, 2004.