CHAPTER

15 Logical Representations of Sentence Meaning

ISHMAEL: *Surely all this is not without meaning.*
Herman Melville, *Moby Dick*

meaning representations

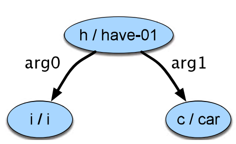
In this chapter we introduce the idea that the meaning of linguistic expressions can be captured in formal structures called **meaning representations**. Consider tasks that require some form of semantic processing, like learning to use a new piece of software by reading the manual, deciding what to order at a restaurant by reading a menu, or following a recipe. Accomplishing these tasks requires representations that link the linguistic elements to the necessary non-linguistic *knowledge of the world*. Reading a menu and deciding what to order, giving advice about where to go to dinner, following a recipe, and generating new recipes all require knowledge about food and its preparation, what people like to eat, and what restaurants are like. Learning to use a piece of software by reading a manual, or giving advice on using software, requires knowledge about the software and similar apps, computers, and users in general.

semantic parsing

computational semantics

In this chapter, we assume that linguistic expressions have meaning representations that are made up of the *same kind of stuff* that is used to represent this kind of everyday common-sense knowledge of the world. The process whereby such representations are created and assigned to linguistic inputs is called **semantic parsing** or **semantic analysis**, and the entire enterprise of designing meaning representations and associated semantic parsers is referred to as **computational semantics**.

$$\exists e,y\ Having(e) \wedge Haver(e,Speaker) \wedge HadThing(e,y) \wedge Car(y)$$




```
(h / have-01
       arg0: (i / i)
       arg1: (c / car))
```

```
Having:
     Haver:  Speaker
     HadThing:  Car
```

Figure 15.1 A list of symbols, two directed graphs, and a record structure: a sampler of meaning representations for *I have a car*.

Consider Fig. 15.1, which shows example meaning representations for the sentence *I have a car* using four commonly used meaning representation languages. The top row illustrates a sentence in **First-Order Logic**, covered in detail in Section 15.3; the directed graph and its corresponding textual form is an example of an **Abstract Meaning Representation (AMR)** form (Banarescu et al., 2013), and on the right is a **frame-based** or **slot-filler** representation, discussed in Section 15.5 and again in Chapter 17.

While there are non-trivial differences among these approaches, they all share the notion that a meaning representation consists of structures composed from a set of symbols, or representational vocabulary. When appropriately arranged, these symbol structures are taken to *correspond* to objects, properties of objects, and relations among objects in some state of affairs being represented or reasoned about. In this case, all four representations make use of symbols corresponding to the speaker, a car, and a relation denoting the possession of one by the other.

Importantly, these representations can be viewed from at least two distinct perspectives in all of these approaches: as representations of the meaning of the particular linguistic input *I have a car*, and as representations of the state of affairs in some world. It is this dual perspective that allows these representations to be used to link linguistic inputs to the world and to our knowledge of it.

In the next sections we give some background: our desiderata for a meaning representation language and some guarantees that these representations will actually do what we need them to do—provide a correspondence to the state of affairs being represented. In Section 15.3 we introduce First-Order Logic, historically the primary technique for investigating natural language semantics, and see in Section 15.4 how it can be used to capture the semantics of events and states in English. Chapter 16 then introduces techniques for **semantic parsing**: generating these formal meaning representations given linguistic inputs.

15.1 Computational Desiderata for Representations

Let's consider why meaning representations are needed and what they should do for us. To focus this discussion, let's consider a system that gives restaurant advice to tourists based on a knowledge base.

Verifiability

Consider the following simple question:

(15.1) Does Maharani serve vegetarian food?

verifiability

To answer this question, we have to know what it's asking, and know whether what it's asking is true of Maharini or not. **verifiability** is a system's ability to compare the state of affairs described by a representation to the state of affairs in some world as modeled in a knowledge base. For example we'll need some sort of representation like $Serves(Maharani, VegetarianFood)$, which a system can can match against its knowledge base of facts about particular restaurants, and if it find a representation matching this proposition, it can answer yes. Otherwise, it must either say *No* if its knowledge of local restaurants is complete, or say that it doesn't know if it knows its knowledge is incomplete.

Unambiguous Representations

Semantics, like all the other domains we have studied, is subject to ambiguity. Words and sentences have different meaning representations in different contexts. Consider the following example:

(15.2) I wanna eat someplace that's close to ICSI.

This sentence can either mean that the speaker wants to eat *at* some nearby location, or under a Godzilla-as-speaker interpretation, the speaker may want to devour some nearby location. The sentence is ambiguous; a single linguistic expression can have one of two meanings. But our *meaning representations* itself cannot be ambiguous. The representation of an input's meaning should be free from any ambiguity, so that the the system can reason over a representation that means either one thing or the other in order to decide how to answer.

vagueness

A concept closely related to ambiguity is **vagueness**: in which a meaning representation leaves some parts of the meaning underspecified. Vagueness does not give rise to multiple representations. Consider the following request:

(15.3) I want to eat Italian food.

While *Italian food* may provide enough information to provide recommendations, it is nevertheless *vague* as to what the user really wants to eat. A vague representation of the meaning of this phrase may be appropriate for some purposes, while a more specific representation may be needed for other purposes.

Canonical Form

canonical form

The doctrine of **canonical form** says that distinct inputs that mean the same thing should have the same meaning representation. This approach greatly simplifies reasoning, since systems need only deal with a single meaning representation for a potentially wide range of expressions.

Consider the following alternative ways of expressing (15.1):

(15.4) Does Maharani have vegetarian dishes?

(15.5) Do they have vegetarian food at Maharani?

(15.6) Are vegetarian dishes served at Maharani?

(15.7) Does Maharani serve vegetarian fare?

Despite the fact these alternatives use different words and syntax, we want them to map to a single canonical meaning representations. If they were all different, assuming the system's knowledge base contains only a single representation of this fact, most of the representations wouldn't match. We could, of course, store all possible alternative representations of the same fact in the knowledge base, but doing so would lead to enormous difficult in keeping the knowledge base consistent.

Canonical form does complicate the task of semantic parsing. Our system must conclude that *vegetarian fare*, *vegetarian dishes*, and *vegetarian food* refer to the same thing, that *having* and *serving* are equivalent here, and that all these parse structures still lead to the same meaning representation. Or consider this pair of examples:

(15.8) Maharani serves vegetarian dishes.

(15.9) Vegetarian dishes are served by Maharani.

Despite the different placement of the arguments to *serve*, a system must still assign *Maharani* and *vegetarian dishes* to the same roles in the two examples by drawing on grammatical knowledge, such as the relationship between active and passive sentence constructions.

Inference and Variables

What about more complex requests such as:

(15.10) Can vegetarians eat at Maharani?

This request results in the same answer as the others not because they mean the same thing, but because there is a common-sense connection between what vegetarians eat and what vegetarian restaurants serve. This is a fact about the world. We'll need to connect the meaning representation of this request with this fact about the world in a knowledge base. A system must be able to use **inference**—to draw valid conclusions based on the meaning representation of inputs and its background knowledge. It must be possible for the system to draw conclusions about the truth of propositions that are not explicitly represented in the knowledge base but that are nevertheless logically derivable from the propositions that are present.

inference

Now consider the following somewhat more complex request:

(15.11) I'd like to find a restaurant where I can get vegetarian food.

This request does not make reference to any particular restaurant; the user wants information about an unknown restaurant that serves vegetarian food. Since no restaurants are named, simple matching is not going to work. Answering this request requires the use of **variables**, using some representation like the following:

variables

$$Serves(x, VegetarianFood) \tag{15.12}$$

Matching succeeds only if the variable x can be replaced by some object in the knowledge base in such a way that the entire proposition will then match. The concept that is substituted for the variable can then be used to fulfill the user's request. It is critical for any meaning representation language to be able to handle these kinds of indefinite references.

Expressiveness

Finally, a meaning representation scheme must be expressive enough to handle a wide range of subject matter, ideally any sensible natural language utterance. Although this is probably too much to expect from any single representational system, First-Order Logic, as described in Section 15.3, is expressive enough to handle quite a lot of what needs to be represented.

15.2 Model-Theoretic Semantics

What is it about about meaning representation languages that allows them to fulfill these desiderata, bridging the gap from formal representations to representations that tell us something about some state of affairs in the world?

model

The answer is a **model**. A model is a formal construct that stands for the particular state of affairs in the world. Expressions in a meaning representation language can be mapped to elements of the model, like objects, properties of objects, and relations among objects. If the model accurately captures the facts we're interested in, then a consistent mapping between the meaning representation and the model provides the bridge between meaning representation and world. Models provide a surprisingly simple and powerful way to ground the expressions in meaning representation languages.

First, some terminology. The vocabulary of a meaning representation consists of two parts: the non-logical vocabulary and the logical vocabulary. The **non-logical vocabulary** consists of the open-ended set of names for the objects, properties, and relations that make up the world we're trying to represent. These appear in various schemes as predicates, nodes, labels on links, or labels in slots in frames, The **logical vocabulary** consists of the closed set of symbols, operators, quantifiers, links,

non-logical vocabulary

logical vocabulary