A Handbook for Language Engineers

Ali Farghaly (ed.)

February 23, 2003

CENTER FOR THE STUDY
OF LANGUAGE
AND INFORMATION

Contents

1

Knowledge Representation for Language Engineering

MATTHEW STONE

1.1 Introduction

I understand KNOWLEDGE REPRESENTATION first and foremost as a technical practice: the enterprise of specifying information about the world for use in computer systems. Knowledge representation as a field also encompasses conceptual results that call practitioners' attention to important truths about the world, mathematical results that allow practitioners to make these truths precise, and computational results that put these truths to work. This chapter surveys this practice and its results, as it applies to the interpretation of natural language utterances in implemented natural language processing systems. For a broader perspective on such technical practice, in all its strengths and weaknesses, see (Agre 1997).

Knowledge representation offers a powerful general tool for the science of language. Computational logic, a prototypical formalism for representing knowledge about the world, is also the model for the level of logical form that linguists use to characterize the grammar of meaning (Larson and Segal 1995). And researchers from (Schank and Abelson 1977) to (Shieber 1993) and (Bos to appear) have relied crucially on such representations, and the inference methods associated with them, in articulating accounts of semantic relations in language, such as synonymy, entailment, informativeness and contradiction. The new textbooks (Blackburn and Bos 2002a, Blackburn and Bos 2002b) provide an excellent grounding in this research, and demonstrate how deeply computational ideas from knowledge representation can inform pure linguistic study. In this short chapter, I must leave much of

A Handbook for Language Engineers.
Ali Farghaly (ed.).
Copyright © 2003, CSLI Publications.

this work aside.

Instead, I will focus on those uses of knowledge representation that are more directly practical. In language engineering, the essential role of knowledge representation is to capture connections between language and the world. Systems that generate natural language must start from an internal representation of the information about the world that the user needs. Systems that interpret natural language must recognize the inferential links that connect the user's utterance to presupposed information about the world. Section 1.2 offers more precise illustrations of the ways we use language to describe the world, and the pragmatic reasoning that underlies such descriptions. To create systems that bridge language and the world the same way, language engineers need to be familiar with the techniques of knowledge representation, which will inform the available representations of domain knowledge; and they need to be comfortable with the practice of knowledge representation, which they must use to link these domain representations to linguistic constructions. Knowledge representation thus serves as a bridge between ONTOLOGY, which as surveyed in (Noy, Chapter 5, this volume) governs systems' domain representations, and GRAMMAR WRITING, which as surveyed in (Butt and Holloway King, Chapter 4, this volume) governs systems' grammatical description of linguistic constructions.

At the heart of knowledge representation is a methodology for the design of computer systems that allows us to understand them as encoding and using information about the world—any aspect of the world—to solve problems. However, as Section 1.3 explores in more detail, the key to this methodology actually lies in the way it aids engineers in developing a precise understanding of the world in the first place, before system-building gets underway. Our effortless abilities to act in the world ourselves can easily mask the complexity and flexibility of the information that we require. The methodology of knowledge representation provides a practical discipline for systematizing such information. Linguists will find much that is familiar from their own discipline in this methodology, for linguistics springs from similar motivations. Our effortless ability to use our native language, acquired without formal instruction, masks the complexity of the linguistic knowledge that we implicitly rely on. In fact, in developing and implementing models of pragmatic reasoning, linguists and knowledge representation researchers share a common project of identifying, describing and formalizing the knowledge that underlies the capacity for verbal communication.

In the rest of this chapter, I complement these general discussions with a survey of results that connect the practice of knowledge representation more specifically to language engineering. Section 1.4 explores the picture of the world that language presupposes; Section 1.5 surveys the formal methods that are available to reason about this picture of the world precisely and efficiently.

I close in Section 1.6 with an assessment of knowledge representation and its prospects for language engineering.

1.2 Motivation

Language is general. For the countless open-ended situations in which we might find ourselves, language offers a rather small stock of words and constructions, with rough-and-ready meanings which we must fit together creatively to suit our needs. Yet the abstract information we can convey in language allows us almost effortlessly to advance our specific projects in specific contexts. Thus, consider example (1).

(1) I would like coffee.

By uttering (1), a speaker can request an action by the hearer and thereby coordinate with the hearer to advance goals in the world. In particular, you have no doubt imagined (1) as an instruction to surrender a mug of steaming liquid to the speaker's physical control—thinking of (1) as a response to waitstaff's question (2) perhaps.

(2) Would you like a drink with dessert?

The specific content that an utterance carries when used by a speaker on a particular occasion, as with (1) in answer to (2), is its INTERPRETATION. Interpretation in this sense is part of linguistic pragmatics. By contrast, I will reserve MEANING for the general semantic description of the utterance, as provided by the grammar, abstracting away from any use of the utterance in a particular context.

The meaning of (1) in this sense is much more general than this one interpretation at first suggests. Too see this, note that (1) works just as well as a request for a clerk to scoop out an ice-cream cone in the context established by (3a), as a request for a wealthy host to make certain arrangements with the household cook on the speaker's behalf in the context established by (3b), or as a request for a coach to assign the speaker to affiliation on a particular team in the context established by (3c).

(3) a. Which of these flavors do you want?
b. What will you have to drink tomorrow morning?
c. Will you program for Team Coke or Team Coffee?

Semantically, (1) specifies the desired action just by its result: through it, the speaker must come in some sense to possess something identified as coffee. (1) looks to the context for the object the speaker intends to get, the kind of possession the speaker requires, and the kind of action the speaker expects. Here the possibilities are as limitless as the world itself, and when we link language to context, we seem quite simply to be linking language up with our full shared understanding of our present situation. In other words, interpretation may be FORMALIZED in terms of parameters from the context, describing speaker, time, place and so forth as is sometimes done in formal semantics, following (Kaplan 1989), for example; however, interlocutors' pragmatic reasoning must in fact CONSTRUCT this formal context (Thomason 1999), drawing on interlocutors' shared physical environment, their information about one another, and their expectation for the interaction. See (Clark 1996, Stalnaker 1998, Bunt 2000).

Language engineering places a natural emphasis on the interpretation of utterances. By constructing interpretations, interlocutors relate application talk to the objects that they know about in the domain and the actions that they can take there. (1), for example, might be something users would say to a virtual barista. In this application, the interpretation of (1) links the utterance to an action the system can take: *serving a cup of coffee to the user*, one of the agent's primitive capabilities. The meaning of the utterance, by contrast, may have to be articulated in more general terms, as we have just seen, and so the meaning may have to abstract away from specific connections with the application domain. By reasoning from interpretations, meanwhile, interlocutors can determine how to proceed with the conversation and the interaction. In any application, when an utterance expresses preference for some action, a cooperative agent may respond by taking that action. Here, then, by reasoning from the interpretation, the system knows to exercise its capability of *serving a cup of coffee to the user*. Again, the general meaning of utterances may have to be too abstract to support such inference in an application.

Language engineering also places a natural emphasis on keeping connections between meaning and interpretation as constrained and specific as possible. Systems do need to work both with meaning and with interpretation; in fact, they must have interpretive processes that seamlessly combine SEMANTIC information—knowledge of meaning—with PRAGMATIC information, describing the user, task and environment. This helps them to accommodate peoples' natural variability in language use (Furnas et al. 1987, Brennan to appear), to mesh with robust designs that give systems some competence in new or unpredictable environments, and to support flexible reuse of linguistic resources and modules across applications.

However, practical systems cannot implement the connection between meaning and interpretation in its full generality. For one thing, we are still uncertain about the shape such a general account will take. Some view the problem of language understanding as a problem of drawing inferences to make an utterance relevant, as in (Sperber and Wilson 1986, Hobbs et al. 1993); others view it as a problem of drawing inferences to fit an utterance into the discourse, as in (Grosz and Sidner 1986, Lascarides and Asher 1991); still others see it as a problem of drawing inferences to attribute an intention to a speaker, as in (Grice 1975, Allen and Perrault 1980); or as a problem of inferring how to participate in a collaboration, as in (Clark 1996, Lochbaum 1998). Indeed, language use might require a combination of these largely compatible inferences, as suggested in (Stone 2003).

More importantly, no matter how such theoretical questions are resolved, it seems clear that interpretation in cooperative conversation must sometimes reflect open-ended reasoning about agents' actions, choices and motivations—see (Grice 1957, Searle 1975, Grice 1975, Lewis 1979, Thomason 1990); although such general reasoning has also sometimes been pursued in models of dialogue in artificial intelligence, too (Allen and Perrault 1980), it remains prohibitively expensive for practical applications. We see the issues involved again in (1). (1) literally expresses the speaker's preference for coffee. However, the utterance functions as a request. This secondary function has a good explanation. Both interlocutors can anticipate that the hearer will act cooperatively to fulfill this interlocutors' preferences. Accordingly, in expressing the preference with (1), the speaker cannot help but convey an expectation that the hearer will follow through. It seems that, in general, such reasoning should figure in utterance interpretation. But enormous obstacles remain in implementing such inferences with the robustness, efficiency and usefulness we find in our own language use.

Rather than attempt such reasoning, a more common strategy is to harmonize meaning and interpretation to cover frequent cases of indirection. A system can be constructed so that the action it takes in response to a literal understanding of an utterance accords with its indirect interpretation. Our discussion of (1) follows this strategy—from utterances that express a preference for some action (a literal understanding), the agent responds by taking the desired action (in keeping with an indirect interpretation). Alternatively, a system can be supplied with new semantic resources that derive an indirect interpretation directly. For (1), the form *I would like* could simply be given the meaning *I request*. On occasion throughout this chapter, I will have to offer some other rather superficial linguistic analyses in the interest of illustrating the practical compromises in systems that talk about the world. I am optimistic that future implementations will be able to take fewer of these liberties with meanings and interpretations.

Overall, then, language engineering brings a natural emphasis on implementing constrained processes that derive utterance interpretation from meaning in context. The key practical and methodological challenge is to capture the connection between natural language utterances and the underlying ontology of our domain of discourse. This domain ontology is typically specified in advance of any linguistic application; see (Noy, Chapter 5, this volume). So the burden of knowledge representation here is to identify, describe and formalize the further background knowledge about this ontology that figures in the meanings and interpretations of utterances in the domain.

Linguists moving into this new intellectual sphere will find that the presupposed background of conversational utterances embraces a panoply of human concerns in rich and fascinating detail. Interpretations effortlessly draw on subtle and intricate characterizations of physical action in the world—see particularly (Crangle 1989, Crangle and Suppes 1994, Webber et al. 1995, Di Eugenio and Webber 1996, Webber 1998). Interpretations highlight the assumptions about time, action and causality that we must also use to predict, explain and plan for our changing circumstances; see (Steedman 1997). And interpretations connect with enduring regularities in our interactions with one another: our goals, beliefs, relationships and choices; for seminal studies, see (Charniak 1973, Hobbs 1978, Schank 1980, Grosz and Sidner 1990, Litman and Allen 1990, Pollack 1990). In accounting for human language use, researchers often view all such information in aggregate, as a single body of common-sense knowledge that we all share, and draw freely on in language use. That may be. As computational systems have not nearly attained such generality yet, in language processing systems such common-sense facts must usually be supplied in new, domain-specific formulations.

Even with an example as simple as (1), we see the extent to which such background knowledge must come into play. We link the words of (1) to what we talk about in (2), in (3), or elsewhere, by drawing inferences that connect general linguistic information (for example, that *would like* concerns some suitable way of getting something) to specific information about our real-world circumstances (for example, that you can get a mug of coffee by having it delivered to you). We link the information (1) provides in words to our ongoing tasks by further inferences still. We cannot enumerate the possible uses of words across the situations we face; we adapt words and constructions creatively to new situations.

The flexibility and context-sensitivity of language can seem intimidating at first. However, natural language systems can appeal to strong constraints on how they connect language and the world—The formal frameworks and case studies developed by language scientists substantiate this. For example, one strong constraint on interpretation simply comes from the hypothesis that utterances DESCRIBE a rich inventory of generalized or abstract individuals, picking these objects out from context and contributing them directly to interpretation. For example, (1) describes the virtual barista's action of *serving a cup of coffee to the user* in this sense. See Section 1.4.

At the same time, the methodology of knowledge representation also anticipates the effort you may need to get a handle on application talk in a new domain. Knowledge representation not only provides essential mathematical techniques, such as modeling languages like first-order logic; it provides guides to apply these techniques to new cases. Thus, the methods of knowledge representation will suggest abstract features and distinctions that you can reapply in describing your new domain effectively. Indeed, the central lesson from knowledge representation is that the way you represent something depends on the inferences you need to draw about it; past research in knowledge representation already offers analyses of important classes of inference that recur across many domains. Of course, these examples include complex cases of reasoning about events, change, action and agency; but they also include numerous useful simpler techniques, to characterize knowledge of categories, roles, and stereotypical situations. The simplest cases are often the most useful. See Section 1.5.

Ultimately, natural language processing and knowledge representation combine together to suggest a variety of declarative programming methods, ranging from logic programming to supervised statistical parameter estimation, that allow us to engineer systems that use language in a new application domain out of nothing more than our precise but conceptual understanding of language as we see it used there. This section has briefly outlined the nature of the problem. The rest of this chapter will sketch some solutions.

1.3 Methodology and Framework

We start by recognizing that specifying information for natural language interpretation is a complex task. In this as in the other aspects of language engineering surveyed in this volume, success depends on acknowledging and managing the complexity of the effort. The practice of knowledge representation adopts a distinctive methodology and framework to address the problem.

Intelligent systems, including typical text processing applications and dialogue interfaces, are particularly difficult to design because their functionality is typically initially very ill-defined. For example, in an dialogue application, we will typically have only a vague understanding of the actions and plans involved in carrying out the system's task, or the language that the system must master to communicate on the task. Even if we can carry out the system's role effortlessly ourselves, we will rarely have access to the intuitive knowledge which we use to do it.

Nevertheless, clear requirements are a prerequisite for the implementation of a working system. Clear requirements simplify and clarify implementation decisions; they ensure that developers can work more independently; and they suggest regimes of verification and validation that help ensure that the final implementation functions as intended. (For more on verification and validation in natural language, see (Barr and Gonzalez 2000, Barr and Klavans 2001).)

To develop these requirements, we must look to the world—we must investigate the environment in which the system must act and then work to discover the behavior the system should exhibit. The initial investigation can take a number of forms. For some applications, we will have strong initial expectations about how a system should act. We make these expectations explicit by using our intuitions to compile examples of desired behavior across the possible situations the system will face (Ratnaparkhi 2000, Walker et al. 2002, Pan and Weng 2002). In other applications, we may not know what situations the system may face, or how it should act in them. In such cases, the usual strategy is to create an interactive mock-up of the system, and use it to collect and evaluate potential interactions that we could choose to realize in a final system (Dahlbäck et al. 1993, Walker 2000). These experiments simply collect data. We get requirements by describing, analyzing, and finally modeling this data, to characterize what we have observed in terms of its precise consequences for system design. For systems that use natural language, knowledge representation is an essential part of this process.

For systems that use natural language, our requirements must specify how the system is to connect input utterances to the application domain, and how the system should in turn report facts from its application domain back to users in natural language utterances. These requirements must respond to our observed data set, so we begin by drawing on the domain ontology to characterize the observed connections between language and the world in formal terms. Because the domain ontology will be structured for application inference, the techniques of knowledge representation will already come into play at this stage. However, the requirements must abstract away from any individual examples; requirements must generalize in meaningful ways to the full range of possible interactions with the system. Knowledge representation really comes into its own in formulating these meaningful generalizations.

Using the methodology of knowledge representation, we can identify the abstract concepts and the abstract relationships that tie together domain elements that surface in language in analogous ways. We can go on to describe connections between language and the world in this general vocabulary. This constitutes the linguistic semantics that our system must implement—these general statements characterize both our attested examples and the many further utterances we expect our system to eventually deal with. In addition, we can use knowledge representation machinery to automate inferences with these general statements. For example, we can classify new elements of the domain ontology on the basis of our formal generalizations about connections between language and the world; we can thereby predict ways these domain elements could be described in application language, and implement modules for interpreting or generating this language.

The methodology of knowledge representation is only one tool of many in developing natural language systems, of course. Often, it is best combined with hypothesis-testing and data-mining tools, which help to frame, refine and validate generalizations about the data as we formulate them. This analysis may be necessary for exploring the trade-offs among different perspectives on system functionality. At the same time, final requirements may underspecify system behavior; our evidence and intuitions may leave us indifferent among alternative possible choices for the system. So there is again room to combine the techniques of knowledge representation with automatic tools that optimize system behavior, using machine-learning for example.

The rest of this section delineates the practice of knowledge representation precisely. My presentation proceeds through three successive layers of analysis that characterize the capabilities of intelligent systems. I start in Section 1.3.1 by considering the KNOWLEDGE LEVEL, which simply characterizes the information that an intelligent system must have to act successfully in its environment. In Section 1.3.2, I introduce the level of REPRESENTATIONS AND ALGORITHMS, where we introduce the essential features of computation—for example, symbol manipulation—while maintaining a precise correspondence with the knowledge-level theory. Finally, in Section 1.3.3, I consider the level of IMPLEMENTATION, which describes how proposed representations and algorithms are to be realized in physical mechanisms. These layers of analysis have been used not only to help create computer programs in artificial intelligence; they have also been used to frame theoretical and empirical investigation into natural computation (Marr 1982, Newell 1982). In language engineering, we adopt the distinction so that we can specify methodically, in manageable steps, how the system will use real-world information in utterance interpretation.

1.3.1 The knowledge level

To describe language use at the knowledge level, we characterize the information about the world and about language that an agent must have if it is to connect natural language utterances to specific ongoing projects and situations, as people do. The knowledge level abstracts away entirely from computation and implementation, so we can draw our account in part from research in semantics and pragmatics, which characterizes our knowledge of meaning, and in part from research in cognitive science, which characterizes our knowledge of the common-sense world. Our criterion for work at the knowledge level is simply that it give a true account of the system's language and its domain.[1]

[1]In an influential paper (Levesque 1984), Hector Levesque showed that knowledge-level analysis can usefully play a deeper role in the design and implementation of intelligent systems. Knowledge-level analysis shows how one module in an intelligent system could query the information available to another, without taking into account the representation and implementation that underlies the response. This is a particularly elegant way to think about interfaces between modules in complex, intelligent systems! But the reader should bear in mind that my appeal to the knowledge level is more modest and circumscribed—an invitation to think of system specification and implementation first and foremost in terms of the information the system will need and use.

Interpretation

At the knowledge level we aim first to document what the interpretation of utterances is. In general, speakers use utterances to describe or portray the things we are interested in—things as we conceive of them, individuated abstractly through our ideas about real-world causation, function and intention. I will say that the utterance DESCRIBES the things that it links up with and says something about.[2] In an application domain, the best way to characterize the interpretation of an utterance is in terms of the elements from the domain that the utterance describes.

Description is so useful because we intuitively understand not just what an utterance describes in general, but also the individual things that specific words describe. For example, the word *coffee*, as it is used after (2), describes something, namely **coffee**: a kind of beverage, made from certain roasted and ground seeds and known for its stimulant qualities. (I will continue to use italics when typesetting words under analysis and boldface when typesetting references to things.) And after (3), the word *coffee* describes other things: **coffee-ice-cream**, a kind of frozen confection flavored with the beverage coffee; or **team-coffee**, a group that represents its collective identity in the beverage coffee and its stimulant effects. By allowing us to factor the links between language and the world down to the atoms of linguistic structure, description naturally sets the stage for models of interpretation that generalize productively to new utterances and new situations.

The ubiquity of description means that documenting the interpretation of utterances principally involves annotating words and phrases with the things in domain that they are understood to describe. In practice, documenting the interpretation of utterances this way builds closely on given domain representations, as informed by more general investigations in system-building. To act or reason effectively, any system needs an ontology of individuals and concepts which frames its information about the world. When we understand utterances as descriptions, we take these same individuals and concepts as constituents of utterance interpretation. By allowing a system to link utterances directly to its world, we collapse the tasks of delineating the individuals and concepts the system will reason about and those it will speak about. This gives us less work to do. It also makes the work easier, because specifying interpretations directly in domain terms alleviates many of the complexities and ambiguities of more generic levels of annotation. (Many such pitfalls are considered by (Vieira and Poesio 2000).)

[2]Note here that, following philosophers, I am distinguishing description from REFERENCE; philosophers define reference as an objective real-world connection that links our thoughts or words to some constituent of the real world. See (Neale 1990) for reference or (van Deemter and Kibble 2000) for the concomitant difficulties of coreference. Reference is thus much rarer than description; we can describe Santa Claus, but cannot refer to him!

When documenting interpretation directly in domain terms, it helps to understand the process and rationale that has led to domain representations. Conversely, anticipating an analysis of domain talk early on can avoid difficulties in characterizing utterances later. The methodology presented by Brachman and colleagues (1990) represents a standard way of developing domain representations, and offers key junctures at which developers can consult evidence from language use representing domain knowledge. (See also (McGuiness et al. 1995), (Farghaly and Hedin, Chapter 2, this volume) and (Noy, Chapter 5, this volume).) To begin, we brainstorm to develop a comprehensive list of the kinds of things that we will need to reason and speak about in the domain. If you already have case studies of system reasoning, or a corpus of potential system utterances, the data provides a supplement and check for this brainstorming.

The next step is to regiment this inventory of concepts to come up with a single consistent perspective on the domain. Difficult issues are the GRANULARITY of individuals within this perspective and the ORGANIZATION of its relationships among individuals. Granularity refers to the detail and subtlety of the distinctions you draw among individuals. Your granularity determines, for example, whether you will create separate domain objects for a copy of a book, the whole edition, its text, and its content, or whether you can get by with just one perspective on the book; granularity determines whether you will inventory locations in the world at the level of seats, rooms, buildings or cities; and it determines which actions are primitive and which must decompose into a sequence of steps. Naturally, finer granularity is a source of complexity, and you want only those distinctions which worthwhile for your application.

Meanwhile, to organize relationships, you need to determine which concepts stand alone, and which are implicitly understood within frames of reference involving other objects. You then need to determine which concepts are more general and more specific. Again, these decisions respond to the needs of an application. In a real-estate application, windows might be understood relationally, as parts of a room; in lego construction, a window might just be a kind of piece independent of any others. Likewise, in relating concepts, even if science-fiction scenarios show that concepts are, in principle, logically unrelated, what matters is whether the relationship holds in a system's environment.

Evidence about a system's required functionality is thus essential to decide both the granularity of individuals and the organization of concepts; but linguistic data can also be quite useful. Linguistic usage shows what entities language users describe in the system's environment and what distinctions they make among entities there. The way people describe the environment can also reveal generalizations about the environment and relationships among concepts that language users presuppose. Thus, the annotation of the interpretation of such utterances goes hand-in-hand with more general decisions about how a system will view and reason about its domain. We will pursue the connection between linguistic data and world models through specific principles and case studies in Section 1.4.

Further steps depend on the particular formalism for describing concepts; we will explore a number of alternatives in Section 1.5. In all cases, though, this process is iterative. As you refine your concepts to implementation, you need to return to the original data as a test of your work: can you still describe the data faithfully now that your concepts are more systematic and precise? Here the answer depends not just on your implementation, but also on your interpretation of the data. An important part of this iteration, then, is to settle boundary cases of description as they arise in domain talk, and to articulate your decisions as standards for your annotation.

Meaning

Once we have fleshed out our knowledge-level account of individual interpretations, we can proceed to develop knowledge-level generalizations that characterize interpretation in a general way. These generalizations characterize not just individual utterances, but the complete range of utterances that our system will handle. They capture the meaning of language, as it is used in our application.

Developing these generalizations is a new and coherent task for knowledge representation. It requires us to identify categories of interpretation, to regiment these concepts into a single consistent perspective that determines the granularity and organization of our semantic vocabulary, and to state and to iteratively refine regularities in interpretation in these semantic terms. In pursuing this knowledge representation, we can naturally draw on general theories of interpretation. In particular, in this chapter, I will exploit the characterization of interpretation in terms of description, and formulate semantic knowledge as constraints on what words in utterances can describe. As reviewed in Section 1.2, good practice is to derive interpretation from meaning as straightforwardly as possible. By characterizing meaning in terms of description, we can formulate semantic generalizations that correspond one-to-one to the elements of interpretation and generalize directly over them.

Consider our case with example (1). A theoretically-inspired model of this case might describe the use of *coffee* by adducing the principle in (4a). And it might offer the principle in (4b) for *I* and those in (4c) and (4d) *would like*.

(4) a. *Coffee* can describe any kind of thing connected with the beverage **coffee**, and always does so.
    b. *I* can and does describe the speaker of an utterance.
    c. *Would like* can describe any kind of event that brings its grammatical subject into a certain kind of possession relation with its grammatical object, and *would like* always does so.

d. *Would like* expresses the subject's preference for such an event, and so describes a particular **state**.

These suggestions to some degree encode a range of insights from the linguistic literature. (4a) suggests Nunberg's theory of deferred reference (1979); what definite noun phrases and other descriptive expressions contribute to interpretation is often an entity related to what they literally evoke. (4b) treats *I* as an indexical much as Kaplan might (1989); *I* is interpreted by accessing specific entities from the utterance context. (Note that in wider domains deferred reference might also be appropriate for *I*!) Finally, (4c) and (4d) abstract away from any predefined link to the world in a way that recalls accounts of productive polysemy and metaphor (Lakoff and Johnson 1980, Pustejovsky 1991). Observe that the meaning given in (4c) does not simply say that the subject will have the object; rather it suggests that the context will supply the relationship that results between subject and object, and uses possession as an abstract concept to characterize these possible interpretations.

Articulating the principles in (4) involves the typical methods of knowledge representation. We have identified abstract concepts that link utterances to interpretation; these are concepts like **being-connected-with**, **speaker-of**, **bring-about**, **possession**, and **preference**. We will need to come to a precise understanding of these concepts in order to determine the full inventory of concepts that we need, and their relationships to one another. (In fact, since pursuing and presenting any case study to this level would be beyond the scope of this chapter, readers should recognize that specific examples such as (4) must be illustrative and provisional, not definitive!) Finally, we have applied these concepts to state constraints that characterize interpretations in a general way.

In articulating the principles in (4), we have also drawn quite closely on a simple general approach to interpretation. For the suggested principles in (4) are ready abstractions away from the observed interpretations of (1) in the contexts of (2) and (3). To accommodate the different interpretations we observe for (1), we need only characterize the various things the utterance might describe in the general terms of our semantic vocabulary. (5) and (6) illustrates this:

(5) a. The beverage **coffee** is connected with the beverage **coffee** (in virtue of being it).

b. The group **team-coffee** is connected with the beverage **coffee** (in virtue of representing its collective identity in terms of that beverage and its stimulant effects).

(6) a. The relation of **physical-control** is a kind of possession and **bringing-a-mugful** is a kind of event that can generally result in the speaker's **physical-control** of the beverage **coffee**.

b. The relation of **team-affiliation** is a kind of possession and

> **assignment-to-the-team** is a kind of event that can generally result in the speaker's **team-affiliation** to **team-coffee**.

At the knowledge level, then, interpretations simply combine semantic generalizations, as in (4), with common-sense generalizations, as in (5) and (6), to derive specific descriptive interpretations. From (4), (5a) and (6a), it follows that (1) can express the speaker's preference to be brought a mug of coffee. From (4), (5b) and (6b), it follows that (1) can express the speaker's preference to be assigned to Team Coffee.

A model of this kind starts by predicting what an utterance can readily describe, but this should be enough to make PREDICTIONS for why apparently unambiguous utterances have the interpretations they do. The actual interpretation of an utterance in context has to satisfy all the semantic constraints on interpretation simultaneously. In typical cases, this rules out the vast majority of descriptive possibilities, which are licensed by the semantics of some words in the utterance but ruled out by others. When we restrict consideration to the real-world objects and semantic characterizations that are most relevant to the context—those that fit the speaker's known intentions and the agreed purposes of the conversation—even fewer possibilities will remain. In fact, it is quite likely that there will be just one.

Actually, it is unclear how we would ever be able to understand one another if this kind of model was not effective. Think of our rational coordination, or of our uncertain inference, as we infer the interpretation of an utterance in context. Generalizations like (4) ought to be able to capture our full knowledge of meaning (we can reformulate them, after all). They can instruct us to link utterances as deeply as possible with our information about the utterance situation. If so, we MUST take the utterance to describe the possible matches that are most relevant to context—given that we have applied our full knowledge of meaning and linked it fully with our knowledge of the situation, what strategy, what convention, what evidence could we have which would allow the utterance to describe something else?

In fact, the perennial difficulty of operationalizing pragmatic predictions remains, until we spell out precisely how alternative interpretations fit the context. This is the kind of optimization problem which may require automatic data-analysis and machine-learning. Nevertheless, if our framework admits too many possibilities, we should be sceptical that we can find a ranking for them; many principled difficulties arise (Norvig and Wilensky 1990). This worry inspires us to formulate knowledge-level generalizations about meaning as narrowly as we can, and to be optimistic about the role such generalizations can play in natural language systems.

1.3.2 Representations and algorithms

The level of REPRESENTATIONS AND ALGORITHMS offers us an abstract view of computation that helps to mediate between a knowledge-level account of a system's environment and the specific implementation mechanisms that we devise to realize the system. REPRESENTATIONS are formal structures that we can use to define computational operations but that we can also view as encoding information about the world. To give this meaning to representations, we assign them an INTENDED INTERPRETATION, which is a correspondence with constituents of the knowledge-level theory. In natural language applications, we need representations to formalize interpretations, and to formalize the information about language and the world that a system needs to construct interpretations. Meanwhile, ALGORITHMS are abstract but explicit and mechanical descriptions of the operations that the implementation will carry out. The algorithms in natural language applications include those that can construct candidate interpretations for utterances from representations of linguistic form and meaning, and from representations of the context. Representations and algorithms together provide a formal specification for a computational system. In creating representations and algorithms for natural language, the key project in knowledge representation is to formalize our account of the background information and domain concepts that figure in the interpretations of application utterances and the meaning of application language.

Because of the intended interpretation of representations, we can also characterize algorithms in knowledge-level terms. We can view an algorithm as a function that produces meaningful results across a meaningful range of real-world circumstances. This gives us an abstract framework in which to prove concisely that our computational operations respect their interpretations in the knowledge-level theory. From then on, we can focus on the level of representations and algorithms. For example, to link a specific implementation to the knowledge-level theory, we need only consider whether the implementation correctly supports the representations and algorithms we have defined for it. This breakdown allows us to analyze the implementation with a facility that might otherwise be impossible. In these cases, it is the theory of representations and algorithms that allows us to understand our implementation as possessing and using knowledge.

Interpretation

Let us first explore how to formalize the interpretations of individual utterances. Since at the knowledge level we have characterized the interpretation of an utterance in terms of the domain elements that the utterance describes, we must now formalize these descriptive connections.

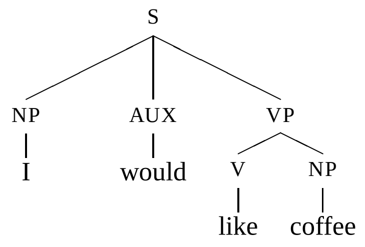
First, we must have representations that correspond to described domain elements. Commonly, these representations are SYMBOLS. A symbol is the atomic primitive of a representation; it is an unstructured element with an immediate and arbitrary correspondence to something in the domain. For example, we could use the symbols in (7) to formalize the things *I would like coffee* describes on its default interpretation. (I use fixed-width font to typeset symbols.)

(7) a. The symbol `u`, with an intended correspondence to the speaker of the utterance (the user of the system).
   b. The symbol `k_c`, with an intended correspondence to **coffee**, the kind of beverage.
   c. The symbol `r_p_c`, with an intended correspondence to the relation of **physical-control**.
   d. The symbol `k_b_m`, with an intended correspondence to the event-type **bringing-a-mugful**.
   e. The symbol `s_p`, with an intended correspondence to the speaker's **state of preference** for an event of type **bringing-a-mugful**.

Evidently, once a knowledge-level account of an utterance's interpretation gives you a precise understanding of its described individuals, you can translate that understanding almost directly into representations. The main challenge is consistency. For example, since analysis of the underlying domain usually precedes investigation into application talk, you may have to reuse representations that have been developed already, determining and respecting these representations' existing interpretations.

Next, we must formalize how description plays out in the utterance itself. This means integrating representations of utterance interpretation with our other linguistic representations. For example, to make precise how, on the intended interpretation of an utterance, its words fit together into a grammatical structure, we typically develop linguistic representations for utterances in the form of syntactic trees. (8) is a suggestive example.

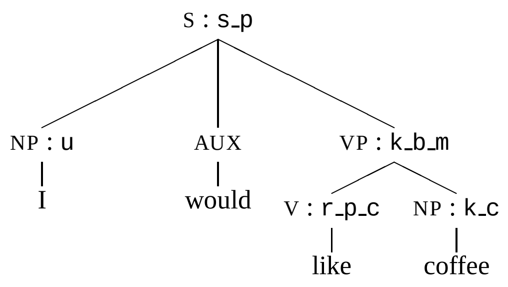
(8)



See also (Butt and Holloway King, Chapter 4, this volume) and (Megerdoomian, Chapter 6, this volume) for more sophisticated structures and a range of specific representations for them. We must formalize the disambiguated interpretation of the utterance in a way that builds on and extends this representation of linguistic structure.

One way to do this is to INDEX each node in the tree by the domain elements that this constituent is intended to describe. Formally, an index is just a label or feature of the node; the terminology recalls and generalizes that of the referential index in syntactic accounts of language structure. (9) shows how the structure from (8) could be indexed to record its intended interpretation, by appending the node category with corresponding representations of domain elements:

(9)



Such structures represent not only what individual words describe but also how those words fit together to create a complex sentence that describes many things simultaneously.

We must be systematic in creating representations such as those in (9) that link linguistic structures to domain information. Inevitably, the effort requires an understanding of the practicalities of achieving coverage and consistency in linguistic representations, as described by (Butt and Holloway King, Chapter 4, this volume); and the effort requires a crisp and precise understanding of domain individuals and concepts, as motivated in Section 1.3.1 and described more fully in (Noy, Chapter 5, this volume). In many cases, though, this just means relying on data and standards that others have prepared as documentation of the available resources; to create representations such as those in (9), we will not need to adapt the structure and design of the system as a whole.

What we will need is our own standards about connections between language and the world. The interpretation of utterances often takes place against a framework which implicitly involves a range of domain individuals and concepts; we have to choose which individuals to make explicit, and where. Since the effort calls for consistency, it is essential to assemble a range of criteria that constrain decision making. For example:

(10) a. When the interpretation of the utterance is different in different contexts simply because we take the utterance to describe a different salient individual, the understood individual should figure explicitly in the utterance representation.

b. When contexts following the utterance allow something implicit in the utterance to be evoked with a pronoun or other elliptical expression, the understood individual should figure explicitly in the utterance representation.

c. When we choose to make an individual explicit, it should appear at every node where modifiers that describe it can attach (as determined syntactically).

The contrast in interpretation of (1) in the contexts of (2) and (3) justifies each of the annotations added to (9) according to these criteria.

The criteria in (10) grow out of theoretical arguments from linguistics. (10a) is a consequence of an anaphoric theory of presupposition, in which context-dependence is achieved by unifying contextual representations and linguistic representations; see (Saeboe 1996). (10b) meshes with a general and uniform analysis of anaphora in discourse, which links reduced forms to repeated description of some domain entity; see (Webber 1983). And (10c) is required to use a compositional semantics for language in deriving observed interpretations. Such theoretical considerations already figure in linguistic accounts of implicit arguments; see (Larson 1986) for example. But a natural language processing system that implements any of these theories will be dependent on its conditions being met in its representations; see (Stone et al. 2001) for discussion.

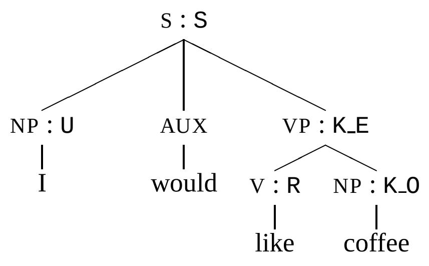
These general insights about interpretation are only half the story, however; as always, our standards must acknowledge the inevitably circumscribed functionality of our implementation. For example, there is no point in parameterizing interpretation in ways that never change in our domain, even if analysis of language use more generally might call for it. More challenging is the need to build on available syntactic and domain representations, which can limit computational semantics through their own autonomous practical constraints. For example, to implement a general formal semantics, we should represent that *would* describes a possibility; see Section 1.4.5. That is, according to the most general semantic account, *I would like coffee* gets its interpretation by evoking a hypothetical possibility where an event of type **bringing-a-mugful** occurs. The utterance reports the speaker's preference indirectly, by describing the pleasure with which the speaker reacts in this hypothetical possibility. We could formalize that account by indexing *would* in (9) to a representation intended to correspond with this possibility. The problem is that the system's formal model of its domain is very unlikely to already provide such representations of possibilities. The abstract reasoning required to make sense of possibilities is generally too difficult to implement. There is little value in modeling the possibility only at the level of interpretation, without the reasoning to go with it. So rather than fleshing out the interpretation in such cases, it makes sense to short-circuit the theoretical account. In (9), we treat the modal *would* as a semantically-empty function word, and directly indicate the state that the whole utterance describes.

Meaning

Once we have characterized interpretation of application utterances, we can proceed to formalize the meaning of application language. At the knowledge level, we describe meaning by introducing categories of interpretation; now, we set up further symbols that correspond to these categories. In particular, we may have PREDICATES, symbols that correspond to semantically-significant domain relationships; and OPERATORS, symbols that correspond to semantically-significant bodies of domain information or restricted domain situations. We complement these new representations with inference algorithms that enable us, for example, to classify domain and utterance elements into semantic categories, or to apply semantic generalizations to flesh out the representation of specific utterances. These algorithms implement the links we have identified between meaning and interpretation in knowledge-level analysis. Again, this process, because it may require system-builders to connect autonomous descriptions of domain reasoning and linguistic structure in new ways, calls for the practice and techniques of knowledge representation.

A perspicuous and representative illustration is the use of VARIABLES in representing semantic generalizations and formalizing semantic inference. Variables are symbols that serve as placeholders; they can be instantiated, or replaced, by any of a set of alternative symbols, as long as that same symbol is used consistently everywhere in place of the variable. (For variables, I adopt the Prolog convention of using symbols with initial capital letters.) Variables express generalizations over the things that those symbols represent. For example, (11) provides a structure analogous to (9) in which variables abstract away from the things any particular utterance of the sentence might describe:

(11)



We can obtain (9) from (11) by the following instantiation:

(12) The variable `U` for the subject is instantiated to `u`; the variable `R` for a possession relation is instantiated to `r_p_c`; the variable `K_O` for a kind of object is instantiated to `k_c`; the variable `K_E` for a kind of event is instantiated to `k_b_m`; and the state `S` is instantiated to `s_p`.

Instantiations can themselves be assigned corresponding formal representations, as in (13), so that we can readily formulate algorithms that construct them, operate on them, or apply them to rewrite our other representations.

(13) [`U=u`, `R=r_p_c`, `K_O=k_c`, `K_E=k_b_m`, `S=S_p`]

Following the knowledge-level accounts of (4), (5), and (6), we could also derive representations for the other interpretations of of (1) by instantiating the variables of (11) to appropriate alternative symbols.

A model of interpretation predicts how variables will be instantiated to values; simply put, the problem is to classify which variables take which values. Formal representations, using predicates and operators, provide a way to describe this classification problem using the semantic concepts and generalizations that we have developed in a knowledge-level account of meaning. For example, the generalizations about meaning presented in (4) depend on a few key relationships. We can create representations of these relationships by using predicates and operators, and so formalize conditions on the variables in (11), as in (14).

(14) a. `connected(K_O, k_c)`, using a predicate `connected` to indicate that the thing described by `K_O` is connected by a close association with the beverage **coffee**.

b. `speaker(U)`, using a predicate `speaker` to indicate that `U` is the speaker of the utterance.

c. `possession(R)`, using a predicate `possession` to indicate that `R` is a kind of possession relation.

d. `result(K_E, holds(R, U, K_O))`, using an operator `result` to describe a view of the world which restricts attention just to the effects of `K_E`, and using a predicate `holds` to indicate that this view of the world finds `U` in `R` with `K_O`.

e. `preference(S, U, K_E)`, using a predicate `preference` to indicate that `S` represents `U`'s interest in seeing an event of kind `K_E`.

To formalize the semantics of (11), we pair it with the constraints on values of variables in (14).

A suitable representation will now allow us to classify domain elements relevant to the context of utterance in semantically-meaningful terms. This will allow a system to derive the representations in (15) algorithmically; these representations are the formal counterpart of the knowledge-level classification (5) of domain elements in our semantic vocabulary.

(15) a. `connected(k_c, k_c)`.

b. `speaker(u)`.

c. `possession(r_p_c)`.

d. `result(k_b_m, holds(r_p_c, u, k_c))`.

e. `preference`(`s_p`, `u`, `k_b_m`).

These representations are identical with those in (14) under the instantiation represented in (13)—a fact which can be readily calculated. Then by applying (13) to the structure in (11) we get a representation of the intended interpretation (9). Overall, then, we arrive at an algorithmic procedure that uses the meaning of (11) as (14) to construct a correct candidate interpretation (9).

An important thing to note about such algorithms is that we can show that they exactly implement our knowledge-level account. As a case study, we will look more explicitly at the account of *I would like coffee* in terms of the constraints in (14). Both the knowledge-level account and our representations involve CONSTRAINT-SATISFACTION PROBLEMS. In any constraint-satisfaction problem, we are given a set of variables (like the variables in (11)), a set of constraints on the values of the variables (like the constraints in (14)), and a database of the instances that satisfy each of the constraints (like the classification results in (15)). In the simplest case, the solution to the problem is just an instantiation for the variables under which each constraint is listed as satisfied in the database (like the instantiation in (13)).[3] For the theory of constraint-satisfaction see (Mackworth 1987); for applications in natural language, see (Mellish 1985, Haddock 1989, Stone and Webber 1998, Schuler 2001).

Constraint-satisfaction procedures depend on MATCHING operations that derive possible instantiations for the variables in a constraint using the instances in the database, and on SEARCH operations that explore these possible instantiations in turn. (One strength of symbolic representations as a bridge between theory and implementation is that these operations can be implemented straightforwardly and with low overhead on digital computers.) Specifically, to describe a basic constraint-satisfaction solver, we draw on two kinds of statements:

(16) a. *Set* ← `match`(*Instantiation*, *Constraint*, *DB*)

b. `search` *Set* `for` *Element* `where` *Expr* `returns` *Result*

(16a) succeeds when we can augment the values for variables in *Instantiation* in such a way that the term *Constraint* has a matching instance in database *DB*. *Set* records the set of matching instantiations. (16b) searches through *Set* to find some result *Result*; *Element* provides a placeholder for each element of the set as the search considers it, and *Expr* defines the operation considered in

[3]In a more general form of constraint-satisfaction problem, instantiations that make the constraints satisfied are merely potential solutions, and these potential solutions are ranked. In particular, a potential solution might be ranked more highly if a variable is instantiated to a preferred value, or if a constraint matches a preferred instance from the knowledge base. Among all potential solutions, we aim to find one that is ranked as highly as possible. When we adopt the knowledge-level principle that utterances describe what they most readily can, we commit to representations that encode these more general constraint-satisfaction problems, and to algorithms that solve them.

the search.

Given these primitives, we can define a constraint-solving algorithm that scans through the list of constraints, accumulating the solution instantiation step-by-step. We represent the algorithm as a procedure `solve`(S,L), assuming that S is an instantiation that satisfies all the constraints thus far considered, and L is the list of constraints that remain to be tried. We define the procedure by cases, according to whether no constraints remain, and L is the empty list $[]$ (17a); or whether at least one constraint remains, and L is a list $[C|Cs]$ whose first element is C and whose remainder is a list Cs (17b). We assume `db` names the database of instances to consider.

(17) a.
```
solve($S$, $[]$)
    return $S$
```
b.
```
solve($S$, $[C|Cs]$)
    $Set \leftarrow$ match($S$, $C$, db)
    search $Set$ for $E$ where solve($E$,$Cs$) returns $R$
    return $R$
```

To solve the complete list of constraints L, we construct a null instantiation `Z` that gives no values to any variables, and run `solve`(`Z`, L). Here is an explicit demonstration that the algorithm computes the consequences of the knowledge-level theory. First, see that anything returned must be a solution. We have assumed that each instantiation returned by `match`$(S,C,$`db`$)$ extends S and finds an instance for C in `db`. In running the procedure in (17), then, the instantiation passed to `solve` always represents a solution to each constraint that has already been considered. When the procedure finally finishes, all the constraints have been considered. So any instantiation returned by `solve` must satisfy all the constraints in L. Conversely, we can also check that `solve` cannot fail when there is an instantiation that satisfies all the constraints. For we have assumed that `match` returns all results and `search` is capable of exploring each in turn.

The precision and accessibility that we have just seen make knowledge representation a key tool in developing computational approaches to meaning. But knowledge representation need not be used on its own. Many machine-learning methods can also approach the problem of classifying new semantic structures on the basis of prior labeled instances; see (Megerdoomian, Chapter 6, this volume) and (Callison-Burch and Osborne, Chapter 7, this volume). In such approaches, we might create a training example for each variable in a sample of data that we have collected. We represent each example in terms of available features; these features will be properties of words and surface structure, as exhibited in (11) for example, which we expect to provide meaningful information about the value of the variable. If we can, we will also include straightforward properties of the context as additional features. In training, we also specify the example class: the domain individual that the variable is understood to describe. Machine-learning builds a model of this classification task from the whole collection of training examples. We can then apply the model to test examples, by formulating each test example in terms of its features and applying the learned model.

We can use such learning methods as a supplement to knowledge representation—we can use logical techniques to assemble a judicious range of alternative candidate interpretations and use learned models to evaluate which candidate is most likely. Alternatively, we can use knowledge representation as a supplement to learning. We can treat relationships like those in (14) simply as additional features of training and test instances, by classifying instances in semantic terms and supplying this information to machine-learning methods. By taking these more meaningful features into account, learned models often may make classification decisions more reliably, or generalize more effectively (requiring less training data).

1.3.3 Implementation

Implemented resources support every aspect of the practice of knowledge representation—not just the implementation itself. For knowledge-level work, ANNOTATION tools, such as the AGTK tools described in (Cotton and Bird 2002) and similar tools described in (Butt and Holloway King, Chapter 4, this volume), help engineers articulate the elements of linguistic interpretation. Meanwhile, KNOWLEDGE-ACQUISITION tools, such as the PROTÉGÉ tools described in (Grosso et al. 1999) and (Noy, Chapter 5, this volume), help engineers articulate the elements of a model of the world. Equally useful are HYPOTHESIS-TESTING and MODELING tools that help researchers explore and refine their assumptions about meaningful distinctions in language and the world; see (Witten and Frank 2000) or (Megerdoomian, Chapter 6, this volume) for a practical introduction to these tools. For work with representations and algorithms, implementation contributes tools for PROOF, SIMULATION and EXPLORATION, which help developers evaluate their representations and algorithms. Tools such as the MOCHA system (Alur et al. 1998, Alur and Henzinger 1999) are increasingly general ingredients of programming practice, though they are not yet widely used in language engineering.

First and foremost, of course, the level of IMPLEMENTATION comprises a repertoire of tools and ideas about how proposed representations and algorithms are to be realized in physical systems. To take a simple case, the programming language Prolog is a convenient tool for realizing constraint-satisfaction solvers on digital computers. (Sterling and Shapiro 1994) presents a high-powered investigation of Prolog as a computer science tool.

Prolog contains primitives for matching and search that are equivalent to those defined in (16) and used in (17). Prolog always maintains an instantiation of values for variables as part of the current state of program execution; so you can think of any Prolog operation as implicitly extending an input instantiation to alternative output instantiations. (Actually, instantiations are the only kind of results that statements can return in Prolog!) Prolog also has a built-in database. So for matching, Prolog uses a procedure `clause`$(C,$`true`$)$, which extends the (implicit) current instantiation of values for variables by looking for a fact that matches C in the (implicit) database. Meanwhile, whenever an operation might return alternative results, Prolog automatically carries out a systematic search of those results. After a call to `clause`$(C,$`true`$)$, Prolog will search over all the alternative matching assignments. In all then, (18) recasts (17) as an equivalent Prolog program—a simple implementation of constraint-satisfaction for any digital computer.

(18) a.
```
solve([]).
```
b.
```
solve([C|Cs]) :-
    clause(C,true),
    solve(Cs).
```

(18a) represents a successful path of execution, which returns the implicit input instantiation as its implicit result (compare the explicit (17a)). (18b) extends its implicit input instantiation by matching the first constraint against the database, then automatically searchers over the alternative results, using `solve` recursively to construct and return an assignment that takes the rest of the constraints into account.

You run a Prolog program by posing a query. Consider the query in (19) for example:

(19)
```
solve([connected(K_O, k_c),
       speaker(U),
       possession(R),
       result(K_E, holds(R, U, K_O))
       preference(S, U, K_E)])
```

The query asks for an instantiation in which `K_O` is stuff related to coffee, `U` is the speaker, `R` is a kind of possession, and `K_E` is the kind of event that will get the speaker to have the stuff in this sense. Thus, the program in (18) and the query in (19)—with suitable further clauses describing the domain, as in (15)—together implement the knowledge-level model of interpretation from (4). They compute the link between an utterance of (1) and its context.

Evidently, if you anticipate and exploit the resources of the implementation level—like we used Prolog as a tool here—you can translate your representations and algorithms almost directly into a specification from which an implementation follows automatically. In tandem with efforts at the knowledge level and the level of representations and algorithms, this step establishes a tight connection between your intuitive understanding of some real-world circumstances (in language interpretation, say) and your understanding of a system you have built that acts within those same circumstances.

1.3.4 The Results of Knowledge Representation

This chapter presupposes this framework—of knowledge-level theory, representation and algorithms, and implementation—in the further results it presents. Those results themselves reflect not just this general framework, but specific insights about the world and about language.

Section 1.4 returns to the knowledge-level treatment of DESCRIPTION in utterances. I argue that to support language use, we must describe key real-world subjects—matter, space, time, action, mind—in terms of a rich inventory of entities; we must also make explicit important underlying properties of these entities, in terms of causation, function and agency. These properties prove essential for expressing deep generalizations about the world, and so, not surprisingly, they provide the basis for a wide range of linguistic distinctions.

Section 1.5 surveys the representations and algorithms required to create models of description. We look at the use of logic to provide computational representations of real-world relationships constituents of interpretation; and we explore what is involved in deriving such formal representations from generalizations about language and the world.

Finally, Section 1.6 offers a summary of the results and an assessment of the requirements and prospects for knowledge representation in language engineering. Inference about interpretation only makes sense in the context of systems that have reliable, language-independent information, represented with a certain richness that makes linguistic distinctions meaningful. We can expect such representations to remain expensive for quite a while—but increasingly feasible.

Although our language use is intimately tied up with the sophistication of our own understanding of the world, the structure of language places fewer constraints on the representation of world knowledge (that is, on thought) than one might have supposed (contrast (Whorf 1956, Vygotsky 1962)). Utterances can express any relationship in any domain, as long as the relationship is cashed out using elements that can be retrieved from a domain-specific ontology, and as long as the distinctions made in the utterance distinguish the required values in that ontology from the alternative possible values there. This flexible link between language and other representations of the world not only explains the success of language as a natural system of communication, but at the same time motivates the attention and the optimism with which we can continue to explore the practice of knowledge representation for language processing.

1.4 Elements of Interpretation

Natural language in no way limits us to describe just the familiar middle-sized physical objects of everyday experience. Indeed, any natural language utterance characterizes a wide range of GENERALIZED INDIVIDUALS, including such abstract things as categories and kinds; actions, events and times; places and paths in space; and scales, quantities and measurements. Our analysis of (1), *I would like coffee*, as spelled out in the discussion of (4–7), already illustrates this key fact. We took (1) to describe a KIND of beverage, a RELATION of possession, a STATE of preference.

The importance of generalized individuals to natural language interpretation rules out a number of simple and effective forms of knowledge representation. In particular, in many domains you can go quite far with representations that only use unanalyzed proposition symbols. Agre (1997), for example, formulates sophisticated agent behavior in terms of richly indexical atoms such as *the-bee-I-am-attacking-is-running-away-from-me*. The propositions Agre uses correspond directly to the recurring features of the environment that the agent must consult in its decisions. This approach is heavily geared toward success on specific tasks in specific applications, because each possible feature of a new situation has to be formalized with a completely distinct representation. However, when the approach fits, it is simple and efficient because rules for the agent can be written straightforwardly and the agent can realize them with minimal inference.

More complex structure is simply required for an agent that interacts using linguistic descriptions of its world. Such agents must have explicit representations of, and information about, the real-world individuals that human users attach significance to and talk about. Jackendoff (1983, (3.9) and (3.10), p. 53) crisply illustrates the diversity of these individuals. He proposes to analyze each of the exchanges in (20) through a represented individual that provides the topic of the question and the content of the answer.

(20) a. What did you buy? A fish.
b. Where is my coat? In the bathtub.
c. Where did they go? Into the bathtub.
d. What did you do? Go to the store.
e. What happened next? Billy fell out the window.
f. How did you cook the eggs? Slowly.
g. How long was the fish? Twelve feet (long).

It is an ordinary object in (20a), a place in (20b), a path in (20c), an action in (20d), an event in (20e), a method in (20f) and a measurement in (20g). To underscore the link of such examples with represented individuals, Jackendoff also notes the possibility of nonlinguistic answers as well: “producing the fish, pointing to the place or in the direction, performing the action, pointing to the event (or its result), demonstrating the manner, or demonstrating the size."

The practice of knowledge representation for natural language demands an unequivocal understanding of such generalized individuals. Analysts must often draw on this understanding to identify the individuals described in a particular utterance. More generally, analysts may also be called upon to represent language-independent real-world generalizations in which such abstract individuals figure. Accordingly, I now undertake a brief survey of generalized individuals and the principles that govern them. I will adopt a linguistic perspective, whose most direct antecedents may be Hobbs's discussions of generalized individuals (1985, 1995) in computational linguistics and Bach's (1986) in formal semantics. But where appropriate I will suggest connections to more computational work, particularly the increasingly rich and diverse systems of representation under development for the semantic web (Fensel et al. 2002).

1.4.1 Objects and Their Associates

Let us begin by considering the constituents of the physical world, where already we find a broader range of things to talk about than one first suspects. I'll take as an example one of the pushpins I use in my office; it's a round bead of blue plastic affixed to a short spike of steel wire ending in a point. My observations will simply be those required to justify the description of the pin I just gave.

For cognitive scientists, there is just one object here, namely the pin itself. An OBJECT is something that can move coherently as a unit, and maintains its internal physical relationships while in motion. So understood, objects are the basic participants in unfolding physical events: pushing, pulling, dropping and the like. They are also typical participants in physical relations, including support and containment. Not surprisingly, we seem predisposed to draw extensive inferences about such events and relations, in terms of objects that move consistently and coherently, from infancy; see (Spelke 1994) for example. Indeed, such objects seem fundamental to the operation of the visual system in particular; see (Scholl 2001) for review and (Siskind 2001) for a computational realization.

Language is not limited to talk of objects; we can zoom in and focus on parts of objects. Above, I described three parts of the pushpin: the head, the spike, and the point. In the typical case, a PART identifies some but not all of an object because that part has its own shape, its own history and its own action. Take the pushpin. Its parts are distinguished by shape—the head is round where the spike is straight and the point sharp. In manufacture, the head is made separately out of a different material and then affixed to the spike, while the point also must be specially cut into the spike. In use, the head allows the pin to be pushed and pulled comfortably by hand, while the point pierces surfaces and the spike transfixes them. In general, the parts we perceive may not be the parts we create or the parts we use; language allows us to talk about any of them. However, the conspiracy between shape, history and action is genuine: structurally, we seem to segment shapes along boundary curves where the shapes bend most sharply (Hoffman and Richards 1984). These boundaries would actually mark the history and extent of shaping processes that produced protruding parts (Leyton 1992). Of course, in the objects we typically encounter, both natural and artificial, the processes that create an object and its parts closely conform to what the object and its parts must eventually be able to do.

Because we can identify parts by this family of distinct but related criteria, it can be especially difficult to agree on part-whole relationships and to formalize them consistently. To represent domains of parts and wholes successfully (or other similarly difficult cases), we need precise ways of identifying and characterizing the alternative domain relationships. One effective discipline is to articulate principles to use in formalization that help to pin down the concepts under analysis in meaningful causal terms. For example, we may want a part-concept that covers only physical constituents of the whole that that contribute to its basic function or design. We can help get a fix on this relation by insisting that whenever the object is destroyed, its parts must be too. We can use such tests to help us establish, for example, that the muffler is part of a car in a way that its registration or its ice-scraper are not. See (Artale et al. 1996).

As easily as we can use language to zoom in on parts, we can use it to zoom out, and describe sets of things. Descriptions of sets can provide convenient summaries of the properties of individuals, as in (21a); such readings are known as DISTRIBUTIVE. Descriptions of sets also provide convenient overviews of complex interrelationships among objects, as in (21b); this leads to COLLECTIVE interpretations which ascribe properties to a set as a whole.

(21) a. The package contains some yellow pushpins, but I never use them.

b. Two blue pushpins attach the announcement securely to the board.

In (21a), each yellow pushpin has found its individual way into the package, and remains there individually unused. In (21b), though, it is likely that neither blue pushpin suffices on its own create a secure attachment—both are required. A good expedient in representing simple domains to to predefine meaningful groups of individuals directly, just as you predefine meaningful parts; these groups inherit the distributive properties of their members. Unfortunately, this does not capture the full subtleties of interpretation.

Summarization and overview are compatible, it turns out. To see this, consider (22):

(22) a. I briefly secured the announcements with pushpins.

b. But they eventually tore holes in them.

(22a) might describe a situation in which each announcement requires two pushpins—and I might sometimes economize by overlapping adjacent announcements and using a single pin for both. See (Gillon 1987, Verkuyl and van der Does 1994, Schwarzschild 1994, Schwarzschild 1996). By following up (22a) with (22b), we make these complex interrelationships among objects the topic of an extended discourse; (22b) suggests that each announcement got holes from the pushpins it was attached with. See (van den Berg 1993, van den Berg 1996).

Perhaps paradoxically, the expressive flexibility illustrated in (22) provides a strong incentive to tie descriptions of sets directly to represented properties of individuals. That would involve, for example, taking (22) as a report of the contribution that individual pushpins make to securing or tearing individual announcements. This has the advantage of making explicit the interrelationships on which discourse connectivity depends. Linguistic theories along these lines (e.g., (Schein 1993)) are surprisingly effective; so are computational ones (e.g., (Stone 2000, Horacek 2002)).

Language describes a final constituent of the physical world: the MATERIAL that makes things up. The material that makes up the head of the pin is a quantity of plastic; the rest of the pin contains a quantity of steel. One might have thought that the pin simply WAS the plastic and the steel, and for many applications this would no doubt suffice. Descriptions of material would then call for new properties but not for new individuals. However, we shall see that the contrast in perspective, between thing and substance, must be maintained more broadly across descriptions of time, space and measurement (Jackendoff 1991). We capture this straightforwardly only by taking the different perspectives systematically to describe different kinds of things.

1.4.2 Categories

In natural language descriptions, kind terms do not just supply predicates, and are not just used to characterize other things. Kinds are entities with their own distinctive properties. (23) illustrates this.

(23) a. Dodos are extinct.

b. Paper clips were invented in the early 1900s.

For (23a), the individual dodos are merely dead; only the species, **the-dodo**, is extinct. Likewise for (23b), individual paper clips are merely made; it is the kind of office supply, **the-paperclip**, that is invented. A classic source on descriptions of kinds is (Carlson 1980); see also (Carlson 1992, Krifka et al. 1995).

Once we represent kinds as things, to interpret sentences such as those in (23), we are free to exploit kinds elsewhere as well. Indeed, it may be that language always represents that domain individual **x** has domain property **p** as a relationship between autonomous elements, **x** and **p**, drawn from a background model of the world. So to say something has **p**, an utterance would always include a constituent that describes **p**. One argument for this derives from the context-sensitivity we explored extensively in Section 1.2—the properties by which we characterize domain individuals belong to that domain, not directly to language. For example, in different domains we might use *tape* in (24) to instruct our audience to use any of a variety of kinds of material.

(24) Cover the edge with some tape.

In office work, we might expect thin transparent tape, for the edge of a piece of paper; in craft work, we might expect masking tape, for the edge of a surface to be painted; or, in repair, we might expect duct tape, for the edge of a piece of metal or plastic. (The alternative interpretations of (1) make a similar point.)

It is easy enough to represent the facts about the world that make a particular kind of tape appropriate to a particular task. What is problematic is to draw these facts into the processes of language use in a regime that keeps understanding and generation symmetric and avoids open-ended reasoning in either. Modeling utterances as descriptions of domain kinds as well as domain objects solves this problem in a direct and straightforward way; generation and understanding processes then recover the described kinds by matching linguistic constraints against the context. In (24), then, we take *tape* to describe some kind of material from the context: specifically, the material is required to be a long thin fabric with a sticky side. Each context supplies its own kind of tape to satisfy this requirement.

Discourse connectivity provides an independent argument. Read (25) in the context established by (24).

(25) Isn't it the most versatile fastener ever invented?

As in (23b), the word *invented* in (25) signals that the utterance describes some kind, rather than some individual thing. Here that kind is picked out by the pronoun *it*. In the context of (24), *it* is naturally understood as whatever kind of tape we have just been instructed to use. On an analysis that implicates kind reference in attributions like those in (24)—but only on such an analysis—this interpretation of *it* in (25) is completely consistent with other uses of *it* (Webber 1983). In general, *it* can describe anything explicitly described in an earlier utterance, and little else.

Although language impels us to represent kinds pervasively in the interpretation of utterances, language provides little guidance about what kinds there are. In (24), for instance, language tells us that some kinds of things are *tape*. But domain knowledge actually supplies the specifics: there is this particular kind of transparent tape, this kind of masking tape, that kind of duct tape. Formalizing kinds in interpretation therefore involves investigating meaningful categories as constituents of the real world; this investigation draws on judgments about such real-world matters as causation and function, not judgments about language.

The literature of cognitive science calls our attention to three sorts of kinds in particular: NATURAL KINDS, ARTIFACT CATEGORIES and SOCIAL CATEGORIES. All of them crucially involve an explanatory understanding that puts us into contact with meaningful real-world categories. None of these categories are reducible to a boolean combinations of, say, perceptual features or other primitives, and none can be described purely in terms of definitions, theories and inferences in isolation from our connection to the world. See (Fodor 1998). What distinguishes the different sorts of kinds is the character of the underlying causal properties and relationships that determine category membership.

Things form a NATURAL KIND when they are created by the same general causal regularities in the natural world, and when they derive certain common characteristics from those regularities. Our talk of the natural world—with its *water*, its kinds of *jade*, its *tigers*, its *elms* and the rest—abounds in terms that describe natural kinds.

Our judgment that a set of objects forms such a kind represents a hypothesis that the objects' shared features have a uniform, natural explanation. There are alternatives to such a judgment. Apparent similarities may have disparate explanations within the natural world. Or they may result from the intervention of agents, acting individually and intentionally. They may derive from some accident or other special and disruptive circumstance. Accordingly, natural kind membership ultimately depends just on the presence of the right underlying causal processes, not the exact details of something's present state. Zebras are four-legged and striped, but a zebra remains a zebra even without those legs or stripes. See (Kripke 1982, Putnam 1975, Carlson 1992).

Things form an ARTIFACT CATEGORY when they recognizably support a designated use in virtue of some effective structure, composition or design. Our sculpting of our environment is manifest in the many terms for artifact categories we draw upon in describing our surroundings—our *chairs* and *tables*, our *pens* and *ink*, or *paper*.

Artifacts are not categorized by the processes that create them, but by their current conditions. A comfortable horizontal surface in a rock formation counts as a seat whether it has been shaped by the elements or by a hammer and chisel. And either way, with enough damage to the surface, the rock will be a seat no longer. Nevertheless, membership in an artifact category is a causal judgment about what can clearly be done with something, not a direct report of perceptual features. The chair, for example, remains an enduring inspiration to designers because of the elasticity of its possible characteristics: an object can be a chair, as long as its form clearly preferentially accommodates a single person, sitting. For more on the contrast between our understanding of artifacts and natural kinds, see (Keil 1989).

SOCIAL CATEGORIES classify things in terms of the roles, powers and responsibilities people give them in specific culturally-sanctioned practices. Our words for ourselves often describe social categories—*student* and *teacher*; *performer* and *audience*; *teams*, *committees* and *organizations*. But in addition things in the world also fall into social categories, like those we describe with such words as *money*, *signals* and *laws*. See (Searle 1997). In both sorts of cases, social categorization just reflects decisions people have to make about individuals in order to act together. Ultimately, we have social categories because the powers that be designate things as such, and we all agree. Although these decisions can be motivated by other properties of things, they are not determined by them. So membership in social categories is also divorced from what we see explicitly in the world.

These taxonomies call attention to the kinds of explanations that we use to make our real-world distinctions. They invite us to pursue the practice of knowledge representation for language by spelling out interpretations using represented categories that crystallize the same kinds of explanation. Naturally, we may extend this effort beyond the categories of physical entities I have emphasized so far. As we consider events, places and measurements, for example, we expect to find linguistic descriptions of kinds of events, kinds of places, and kinds of measurements. We expect likewise as we consider relations among generalized individuals. Moreover, we expect all these kinds to articulate the same variety of meaningful distinctions, in terms of causality, function and agency, that we see in kinds of physical entities.

It is important to note that these taxonomies are not taxonomies of words, however. Words describe categories, and as we have seen from (1) to (25), they do so extremely flexibly. In fact, for many words, we may be able to say quite little about the categories they describe in context in taxonomic terms. For example, depending on the what matters in a particular case, we may take *vinegar* to describe a kind of result of fermentation (a natural kind, identified just by the preponderance of certain chemicals), as a kind of ingredient for imparting sourness to food (an artifact category, requiring a wholesome present state), or as a category of import, subject to particular taxes or controls (a social category, perhaps motivated by, but not identified with, specific methods of manufacture).

1.4.3 Events and their associates

Objects and kinds are things with an enduring identity; language also describes things that play themselves out over time. To start, we can describe time itself. We have expressions like *Tuesday* which describe intervals of time, and predicates like *was rainy* that categorize these temporal intervals.

(26) Tuesday was rainy.

Moreover, we find many sentences that implicitly characterize an interval of time, and spell out the condition of the world during it. Sentences such as these are called STATIVE. (27a), for example, fits an interval from August, 1992, to August, 1998 (and any part of that time).

(27) a. I lived in Philadelphia.
b. I lived in Philadelphia in 1994.
c. When I lived in Philadelphia, I had no car.

The tense of the sentence, if nothing else, describes the time in question. It is a short step from this implicit description to an explicit one. In (27b), we add a modifier that specifies the interval at issue; in (27c), we make a *when*-clause out of the sentence to contribute the interval to another characterization. See (Cresswell 1985) for a thorough linguistic discussion of temporal intervals and (Allen 1983) for a classic computational one.

Descriptions of times, however, play a surprisingly small part in our accounts of change in the world. EVENTS seem to be more basic. With events, we isolate a pattern of change, initiated by a coherent causal process and effected within a definite time, as something in the world. Like the parts we find within objects or the categories we assemble out of them, the events that we recognize in our changing world go beyond what the world might actually seem to contain. Nevertheless, like parts and categories, events are essential to our knowledge of the world, in the organization they bring to our explanations of our surroundings. Events package some differences between one time and another together with the rationale that grounds those differences within stable schemes of autonomous cause and effect, individual goals and methods, and social coordination.

The best way to make sense of events is to insist on a strict parallel between events and the ordinary things in the physical world. See (Davidson 2002b). The parallel begins with the observation that we have many different ways to describe the same thing, whether it is an object or an event. For example, whether we call it *a rock*, *a seat*, or *a throne*, the object is the same; the difference lies in whether we categorize its nature, its function, or its social significance. To pursue this analogy: consider what happens when John says (1) in the context of (2). John makes a noise; John says *I would like coffee*; John answers. One thing happens; there is one event; but we categorize that same event within different arenas (Davidson 2002c).

Among events, ACTIONS have a special significance for our own practical deliberations, and for our judgments of one another. ACTIONS are the events that we choose; and AGENTS are things that can take actions. More precisely, each action is a material change in an agent, which that agent initiates with a commitment to its significant consequences as the agent foresees them

(Davidson 2002a). Such commitments are INTENTIONS; actions are what an agent does intentionally, that is, with some intention. But of course, an intentional action need not achieve its intended effect. (On intention more generally, see (Bratman 1987).)

A characterization in terms of material changes in an agent agrees with many direct accounts we can give of what we do, as when: I reach my hand into the handle of a mug of coffee; I contract my reach upward and inward; I initiate some motions of my lips, tongue, jaws and throat. We must remember, though, that any such action can, like events more generally, be described in many different ways. So it is that, with these same actions, I grab the mug, pick it up, and drink from it. In general, as long as two expressions implicate the same package of causal connections, playing out in the same stretch of space and time, we should take these expressions to describe the same actions, and the same events.

Many actions bring about effects indirectly. I press the *on* button on my laptop; it goes about its business in response, and finally displays its login screen fifteen seconds later. I ignite the burner beneath a kettle for tea; the fire heats, and then in a few minutes the whistle of escaping steam announces that the water is ready. In such cases, we parse change in the world into successive distinct events: I press, and the computer boots up; I ignite, and the water boils. At the same time, in such cases, we also group the action with its consequences into larger events: I boot up the computer; I boil water. We readily integrate the regular and predictable causal relations among these events into a single package, one thing. Complex events that aggregate actions with their effects over time are often called ACCOMPLISHMENTS, while instantaneous changes are called ACHIEVEMENTS; the terminology is due to (Vendler 1967).

Repackaging events involves changes in perspective that call for subtle and precise language. The EVENT of pressing a button is not the event of booting up the computer; the first is almost instantaneous while the second takes fifteen seconds. But my ACTION of pressing the button is my action of booting up the computer. That is, I do one and the same thing, but we can talk about that action in different ways by characterizing it in the context of different happenings.

These perspectives show how we can understand events, like objects, as wholes composed of parts. Other perspectives show how we can also understand events, like objects, as wholes comprised of quantities of stuff. Within events, we recognize ACTIVITIES or PROCESSES, each a recurring dynamic that in any instance can continue indefinitely and that can play out again in other instances. There is one process at work as the computer is booting; another as the water is heating.

Linguistic descriptions are remarkable for the complex perspectives that individual utterances can encode. See (Steedman 1997) for a review from the perspective of computational semantics. Take (28) for example.

(28) I am boiling water for tea.

(28) characterizes an interval of time during which some specific process is underway: the fire is heating a full kettle, let us assume, adapting our earlier discussion. And this process is characterized in turn by the role it plays in an ongoing, as yet incomplete, pattern of change. My action is already done as this process unfolds; I must have ignited the burner with the intention of keeping it lit until the water boils. On the other hand, the envisaged culmination of the process—the water actually boiling—remains in the future and in fact, if I am interrupted or change my mind, perhaps will never be realized. This is the imperfective paradox observed by (Dowty 1979).

Linguistic descriptions are also remarkable for the flexibility with which they reconcile such different perspectives. *Drinking* describes a process, *coffee* describes a kind of substance, and so *drinking coffee* naturally describes an activity that can be continued indefinitely. By contrast, a mug of coffee is just an object, with a definite quantity of stuff. And so of course *drinking a mug of coffee* describes an event—a complex one that begins with the activity of drinking coffee from the mug and concludes with a finishing event that marks the change when the mug empties and the coffee is drunk. (Jackendoff 1991) is an accessible discussion of this problem of aspectual composition.

The real-world parallel between objects and events extends to a natural parallel in knowledge-representation methodology. We work first with the world. We individuate events at the level of detail we independently require to support our system's real-world reasoning and action; we organize events according to the meaningful explanatory connections in the system's environment. In fleshing out this model of the world, we can draw both on established representation formalisms for talking about real-world processes and events, such as the DAML-S standard ontology for describing action in the semantic web (Ankolekar et al. 2002), and established practice for applying those formalisms to new applications (Brachman et al. 1990). But then we understand utterances as literal descriptions of the real events. We can add individuals where language use absolutely requires it; language use might for example motivate us to make processes explicit that we might otherwise gloss over in system-building. But although these linguistic descriptions may be complex and flexible, they must aim ultimately to connect with the real happenings and the real structure of the world in which the system and the user act.

1.4.4 Space and Quantity

Now across several semantic domains we have encountered the same key principles of description. As users of language, we talk about a wide range of things, we individuate them in ways that serve our explanations of the world, and we characterize them into underlying categories with explanatory significance. We use description to link utterances directly to these meaningful individuals—both as users of language and as engineers. Our discussion remains far from exhausting the full range of generalized individuals, but from now on we can consider further ones only briefly—there will be just the same few principles to apply.

We can describe PLACES and say things about them. A place is a location in space. (29a) describes the place that is *under the bed*.

(29) a. Under the bed is a good place to hide.
b. From New York to Philadelphia is 150km.

We can also describe PATHS, and say things about them. A path is a progression of places, traversed along a dimension of space or time. (29b) describes the path *from New York to Philadelphia*.

Places and paths are delimited according to the causal processes by which objects interact in space. Of course! But note then how places are not determined by mere geometry; contrast (30a) with (30b) and (30c).

(30) a. Chris stood at the window.
b. Chris stood at the door.
c. Chris stood at the mirror.

In all cases we locate Chris by reference to a rectangular plane, indeed perhaps one made in each case of framed glass set in a wall. But in (30a) we locate Chris inside, quite close to the window, looking out; in (30b), though, we probably locate Chris outside the door, quite close, looking in. As for (30c), Chris is positioned to look into the mirror, while dressing perhaps, and so, in contrast to (30a) or (30b), Chris most likely retains a comfortable distance from the mirror. Our descriptions of place in (30) say as much about what Chris is doing, and how, and why, as they say about where Chris is. Of course the interactions we describe are quite plausible; they offer us an easy glimpse of Chris's history and motivations. But in understanding (30) we are not just imposing our expectations on a situation described in purely geometric terms. Replace *at* by *near* in (30) and, continued geometric proximity notwithstanding, much of the subtlety of interpretation disappears.

We illustrate the same property of paths in (31).

(31) Chris went { to the window, to the door, to the mirror }.

The alternative prepositional phrases in (31) describe Chris's path of motion in going. They characterize the path in terms of its endpoint—basically, the motion ends with Chris *at* the landmark Chris goes *to*, with its full geometric and causal implications.

Roughly following (Hobbs 1995), I will use the term GEOMETRIC SYSTEM for a pattern of causal interaction played out in space. (The notion has an analogue in many theories, such as the SITUATION, or meaningful spatiotemporal slice of the world, from Situation Theory (Barwise and Perry 1983); see also (Kratzer 1989).) So, places and paths are identified as parts of geometric systems, and, in turn, locating a figure at a place characterizes the meaningful causal relations of the figure within the system. (30a) locates Chris in a geometric system where the right position affords looking out; (30b) locates Chris in a geometric system where the right position affords going through (and in); (30c) locates Chris in a geometric system where the right position affords looking at an image of oneself.

Naturally, places and paths are also characterized in causal terms. Place prepositions, such as *at*, *in*, *over*, *above*, and the rest, categorize a place in terms of the spatial structure of the interactions that go on there (in the system). Path prepositions, such as *to*, *into*, *through* and *across* characterize paths in similar terms. For example, as we just saw, *at* fits places (and systems) where action depends just on being in the right position, and *to* fits paths in similar systems. A contrasting example is *in*, which fits places (and systems) where action depends on containment, and *into*, which fits corresponding paths.

Where places and paths overlay the causal order onto the geometric structure of space, MEASUREMENTS overlay that causal order more generally across the continuous dimensions of the physical world. Of course, descriptions of measurements can readily make quantity explicit, as in (32):

(32) The Empire State Building is 443 meters tall.

In (32), the word *tall* seems to relate an individual, the Empire State Building, to a measurement of its height; the word *meters* in turn relates the measurement to a tally (counted in meters), and the numeral *443* describes the tally.

However, constructions that describe measurements more typically individuate and characterize them by causal, functional or intentional standards, as in (33).

(33) a. The Empire State Building is tall enough to see from New Jersey.
b. Choose a tall glass to accommodate plenty of ice.
c. Give me one of the tall glasses.

In (33a), the word *enough* explicit relates a measurement of height to a causal result: because the Empire State Building has the height it does, it is possible to see it from an appreciable distance. In (33b), the use of *tall* by itself receives a similar interpretation: the height of a glass counts as *tall* in (33b) in part because you can use such a glass that size to fit the ice you need. In (33c), finally, what counts as a tall measurement follows simply from the speaker's intention to distinguish two sets of glasses in the context; *tall* picks the intended set out by height but does not pick out the other set, the short ones.

To summarize the systematic analogy between descriptions of space and description of measure, we can understand measurements as constituents of SYSTEMS OF VALUES—patterns of meaningful interaction determined by a measured quantity of something. So for example, to learn how tall something is to learn what its height enables, within some system of interaction; and to say that one object is as tall as another is to say that, within some system, their height affords them the same interactions. Thus, we find once again that we are led to the individuals we need to describe, and the kinds of characterizations we need for them, by our explanatory understanding of meaningful real-world differences.

1.4.5 Abstract Objects

Finally, no discussion of linguistic description could be complete without a discussion of the range of further abstract objects that grammar implicates, such as STATES, POSSIBILITIES, FACTS and PROPOSITIONS. But there is perhaps one notable difference between these abstractions and the individuals we have already considered. A specification or system may be equipped with an inventory of concrete individuals to describe, by listing the objects or places in its environment, the meaningfully different quantities it can reason about, and even the events in some history or plan. But to describe abstract individuals in interesting ways, a system needs a generative understanding of them, and needs to create representations of them by inference, on the fly. Such inference in turn requires a more sophisticated implementation. At the same time, the linguistic expressions that describe abstractions are more rarefied. This gives two reasons why knowledge representation does not approach abstractions with the same urgency as other individuals. Of course, abstract individuals remain quite intriguing, as we shall now discover.

A STATE is a fine-grained aspect of the enduring condition of the world at an interval of time. States are part of the causal order, as the examples in (34) show.

(34) a. Winning the lottery caused Terry's happiness.
b. Terry's happiness lasted three years.
c. Chris saw Terry's happiness.

In each of these examples, the noun phrase *Terry's happiness* seems to describe an aspect of Terry's condition. We see in turn that such states can originate with events (34a), and any termination to a state is also triggered by a corresponding event. States are localized and extended in time (34b). And states can influence how things turn out, and play a role in ongoing events (34c). See (Higginbotham 2000).

It is common, especially in computer science, to use the word state to describe a very coarse-grained condition. The state of an automaton, for example, refers to a snapshot that completely characterizes an unfolding computation.

More colloquially, the “state of the union” is a full account of the United States and its institutions. But the sentences in (34) cannot describe Terry’s state—Terry’s ABSOLUTE state—in this coarse-grained sense. At any time, Terry’s absolute state is only partially due to the lottery, and has countless more immediate antecedents. Maybe Terry is hungry from skipping breakfast: the lottery wouldn’t have caused that, nor would that last three years, nor could Chris be expected to see that. Evidently, in this case, Terry has a state of hunger, which is distinct from the state of happiness as described in (34). This is the sense in which states, as described in language, must be fine-grained: Terry is simultaneously in many states!

Talk of states meshes readily with approaches to knowledge representation which characterize objects using a small number of fundamental attributes. Any time an object has any of these attributes, it must have a corresponding state. But states are trickier to work with when our information about the world is less decisively regimented. Take Terry’s case: would Terry’s state of happiness coincide with Terry’s state of glee? And what about simply Terry’s emotional state? For such questions there are few guidelines, as the answers so far seem to have rather few consequences, save that, as always in formal work, absolute consistency is a must.

When we talk about our plans for the future or our explanations of the past, we are as much concerned with what could be as with what is. And we seem simply to DESCRIBE these possibilities:

(35) Terry might win the lottery. Chris would be jealous.

That is, *might* in (35) describes a possibility where Terry wins the lottery, and introduces it into the discourse for discussion. Then *would* describes that same possibility and characterizes it further: in the possibility where Terry wins the lottery, Chris is jealous. Roberts (1989) called attention to the semantic difficulties of descriptions of possibilities in discourse.

Talk of possibilities in planning meshes readily with approaches to knowledge representation which characterize the effects of actions across alternative possible histories. Agents that make plans must represent these alternative histories anyway; they must choose among them. Once we represent the possibilities, we can easily take sentences like (35) to describe them. (Isard 1974) offers the classic demonstration. His program plays and talks about a game of tic-tac-toe; the program interprets conditionals as descriptions of points in a space of possible game-histories that branches move-by-move. With less decisively regimented information, though, possibilities like states can become unmanageable: we can find ourselves required to reason about an new and unexplored possibility, described with no simple standard to judge how it might have arisen or what further information would characterize it. See (Ortiz 1999) for a recent exploration of the computational difficulties involved in constructing these open-ended possibilities.

In our explanations of the world, we also often somehow appeal to the FACTS about the world that ground our claims:

(36) a. The fact that Terry departed early meant that most of the party was fun.

b. Terry's early departure meant that most of the party was fun.

One might think of facts as the immutable data which constitute the world and its history, both in all its coherent regularity and in all its contingent detail. Such data compose into chains of reasons and evidence, but they can only ACT in the world indirectly, through their representations (for example in the minds and statements of agents). Our descriptions of this data can be accomplished by explicit talk of facts, as in (36a), which apparently identifies something as *the fact that Terry departed early*. But perhaps our descriptions of this data can also be mediated by the grammar, as (36b) suggests. (36b) has much the same interpretation as (36a), and much the same structure; this argues that *Terry's early departure* can describe the same thing as *the fact that Terry departed early*.

The distinction between facts on the one hand and events and states on the other is subtle at best. Noun phrases such as *Terry's departure* do not always describe facts; they must sometimes describe events, as in (37a). Events occur in time and facts do not.

(37) a. Terry's departure occurred at ten.

b. Terry's departure disturbed Chris.

A corresponding ambiguity is perceptible in (37b). We can use (37b) to describe a situation where Terry packs noisily, stomps out the door and slams it, and all these sounds keep Chris awake. Chris need not know that Terry is leaving; when we use (37b) this way, we describe the causal connection between Chris and the event itself. However, we can also use (37b) to describe a different kind of situation. Terry slinks out silently unnoticed. Only later does Chris notice that Terry is gone, and infer that Terry must have departed. But then Chris becomes wildly upset at the implications. In such a scenario, (37b) can correctly characterize Chris's relationship to the fact that Terry departed. The difference between these two interpretations highlights the distinction between events and states as things inside the world, with powers to act on their own within the world, and facts as things that constitute the world, which act through their representations.

Talk of facts is naturally tied to talk of PROPOSITIONS. A proposition reifies the content of a representation; it is a claim about the world that is either true or false. (When a proposition is true, we might think that there is some fact in the world which satisfies the proposition.) As with facts, we can talk about propositions explicitly, as in (38a). In fact, we can describe propositions in many ways, depending on the kind of information they embody: *ideas*, *thoughts*, *views*, *theories*, and so forth.

(38) a. Chris suggested the idea that Terry had departed.

b. Chris suggested that Terry had departed.

Again, the parallel structure and interpretation of closely related sentences, such as (38b) here, suggests that we might treat certain grammatical constructions as descriptions of propositions, too.

It may be that, with effort, we can cultivate the linguistic intuitions suggested by examples such as (36), (37) and (38), and develop a systematic account of facts and propositions in discourse; see (Asher 1993), for example. However, developing a formal ontology of facts or propositions is fraught with pitfalls, because of the further domain inferences these abstract objects must support. Many seemingly innocuous assumptions about the form and content of factual inference bring the surprising and devastating consequence that there can be only one fact: “the True”. Apparently, inference with facts must be constructive, not classical, to make sense of the idea that an inference draws on particular facts; see (Girard et al. 1989). For similar reasons, these inferences must also expose the information that identifies individuals to the logical structure of facts and argument; see (Neale 1995). When constructing propositions, meanwhile, apparently natural ways of describing propositions turn out to be inconsistent, because they open up the possibility of paradoxes of self-reference; see for example (Priest 1994). For the moment, these complexities put a systematic treatment of facts and propositions beyond the scope of practical implementation.

1.5 Meaning and Inference

The generalized individuals surveyed in Section 1.4 let us formalize the relationship between application talk and an underlying domain model. They provide an index into application knowledge, as captured by domain representations and portrayed in language. However, to capture the general meaning of application talk, we need to go further; we need to express the generalizations over interpretation that constitute semantic knowledge, and we need to reason from these generalizations automatically in constructing particular interpretations. This section offers a brief survey of the strengths and limitations of a few notable formalisms. I particularly emphasize the generalizations that figure most prominently in semantic knowledge and that therefore contribute most significantly to the applicability of formal techniques to semantic representation.

1.5.1 Description Logic

We begin with DESCRIPTION LOGIC, a particularly simple formalism with many good implementations; see (Baader et al. 2003). Description logic systematizes the generalizations over individuals that can be expressed in terms of simple relationships among abstract concepts. It comes with algorithms that can assess a variety of relationships among concepts in terms of the available generalizations; and because the logic is constrained, these algorithms can be quite efficient.

Saying that description logic is a logic of concepts means that all the logical statements abstract away from particular individuals. For example, rather than identifying the individual that is currently speaking by a name or variable, in description logic we would use use a concept *speaker*. We can combine concepts with boolean operators to describe individuals logically. For instance, we can create the concept $\mathit{speaker} \vee \mathit{hearer}$, true of an individual that fits *speaker* or fits *hearer*, to characterize the participants in a conversation. Inference often proceeds through definitions for concepts; for example, we could DEFINE *participant* as $\mathit{speaker} \vee \mathit{hearer}$. When we conjoin together these concepts, there may be interesting logical relationships among them; and we can pursue them to infer new consequences and more fully characterize any object that falls under the concept. For example, any individual that satisfies $\mathit{participant} \wedge \neg \mathit{hearer}$ in fact satisfies *speaker*.

The expressive power of description logic comes in its ability to construct complex concepts using ROLES that abstract the relationships between individuals. Using roles, we can build concepts that characterize individuals indirectly, through a recursive description that follows the link to a related object and characterizes it in terms of another concept. In fact, description logic lets us quantify over the related objects universally, using *all*; or existentially, using *some*. Take (39) for example.

(39) $\mathit{state} \wedge (\mathit{some\ type\ possession})$

(39) is a complex concept that describes a state; the second conjunct describes the state indirectly—it says that you must be able to find some other related individual, a *type* of the state; this new object must be a kind of possession. The notation suggests a more compact, and somewhat more natural reading for the content of (39): this is a state some type of which is a possession. This more compact wording epitomizes the elegance of the recursive concepts of description logic.

Many roles are attributes that take exactly one value; here description logic offers the quantifier *the* which describes the value of the attribute. Thus, (40) encodes a state of possession whose logical subject is the speaker and whose logical object is associated with coffee.

(40) $\textit{state} \wedge (\textit{some type possession}) \wedge (\textit{the logical\_subject speaker}) \wedge$
$(\textit{the logical\_object } (\textit{some associate\_of coffee}))$

Even the iteration of roles here seems quite natural and effective.

Despite its simplicity, description logic provides a good starting point for natural language applications. Once we populate our representations with a plethora of generalized individuals, as suggested in Section 1.4, it becomes possible to characterize those individuals using just a small number of concepts and roles. For example, to map out the details of a specific scenario in semantic terms, we might need only to link the generalized individuals of the domain to the semantic categories they belong to, and flesh out domain relationships such as the roles ordinary individuals play in complex events and states. Our semantic knowledge is then stated in terms of the concepts characterizing individuals and the roles that link individuals together. Description logic is often a convenient formalism for expressing this information, since description logic formulas are particularly good at defining FRAMES, whose instances package together a number of related objects into a complex using roles and attributes. In effect, (40) is a simple frame.

At the same time, though, many definitions and inferences fall outside description logic. Since description logic emphasizes concepts, its focus is on concept-level inferences like SUBSUMPTION, whether one concept is more specific than another, or CONSISTENCY, whether there is any possible way a concept could be instantiated. There are many extensions to description logic, which offer additional ways to construct concepts; these extensions increase the complexity of reasoning but still generally allow constrained algorithms to decide questions of subsumption or consistency. However, once you can freely describe multiple objects at once, as needed to construct complex terms, for example, you get too much expressive power to reason with algorithms of such limited complexity. Thus, none of the extensions have a robust notion of equations that require objects to be the same, and they cannot readily perform unification to construct complex structures that satisfy multiple constraints. This brings us squarely into the realm of first-order logic.

1.5.2 First-order logic

First-order logic is the logic of arbitrary relationships among individuals. First-order logic uses terms and variables to represent individuals and predicates to represent relationships among them; the illustrations of Section 1.3.2 are first-order representations and Prolog, as described in Section 1.3.3, is a inference mechanism for an important subset of first-order logic.

In first-order logic, predicates combine with terms and variables to form SENTENCES; sentences make claims that the specified objects stand in specified relationships and so can be true or false. First-order logic continues to offer complex sentences involving propositional connectives; it adds quantifiers that explicitly describe the possible values of variables: $\forall xp$ claims that p is true no matter what object x is taken to stand for; $\exists xp$ claims that p is true for some value of x. We can also define a quantifier $\exists!xp$ which says that p is true for exactly one value of x. See (Lepore 2000) for a linguistically-oriented introduction to first-order logic; for applications in computer science see (Gallier 1986) and for the representation of the common-sense world in particular see (Davis 1990).

(41) is the cumbersome equivalent of (40) in first-order logic.

(41) $state(s) \wedge \exists r(type(s,r) \wedge\ possession(r)) \wedge \exists!u(logical\_subject(s,u)) \wedge$
$\exists u(logical\_subject(s,u) \wedge speaker(u)) \wedge \wedge \exists!v(logical\_object(s,v)) \wedge$
$\exists v(logical\_object(s,v) \wedge \exists c(associate\_of(v,c) \wedge coffee(c)))$

It characterizes a state s, where s has a type r that is a possession; where s has exactly one logical subject, where that logical subject u is the speaker; and where s has exactly one logical object, where that logical object v has an associate c that is coffee. Evidently, abbreviation and naturalness are not virtues of first-order logic! The constraint-satisfaction representations of meaning and interpretation presented in Section 1.3 in fact distill substantial practical wisdom aimed at making first-order representations accessible and useful.

The value of explicit first-order representations is the flexibility with which they let you state real-world generalizations. For example, consider how to characterize an agent u's action of getting a physical object o. Call this kind of action k_a. There is a corresponding kind of possession state k_s, where u has o under control; k_a has k_s as a generic consequence. First-order logic lets us flesh this situation out precisely as in (42).

(42) $\forall u \forall o(agent(u) \wedge object(o) \supset$
$\exists k\_a \exists k\_s(possession(k\_s) \wedge generic\_consequence(k\_a, k\_s)))$

We use universal quantifiers $\forall u \forall o$ to indicate our concern for all agents and objects explicit, and we use existential quantifiers $\exists k\_a \exists k\_s$ to relativize our description of the kind of action and the kind of state to the choice of agent and object. Concretely, first-order logic goes beyond description logic by making it possible to express simultaneous interrelated descriptions of multiple objects, as (42) illustrates.

An alternative but equivalent first-order representation is based on a technique called Skolemization that eliminates quantifiers by introducing suitable complex names for objects. The technique is intuitively motivated as follows. Think of narrow-scope existential quantifiers like $\exists k\_a \exists k\_s$ in (42) as implicitly describing functions. The functions require values for the outer variables; they convert these values into an object that makes the existential sentence true. For (42), the function for $\exists k\_a$ takes the values of u and o as arguments and delivers u's action of getting o. We can represent this explicitly by naming the function with an arbitrary new symbol, say f_{k_a} and constructing a term describing the function's result: $f_{k\_a}(u,o)$. This is called a Skolem term.

We can now eliminate existential quantifiers by replacing the quantified variable with a suitable Skolem term everywhere it would otherwise occur free. Meanwhile, we can adopt the convention that any remaining variables are implicitly universally quantified. In this scheme, (42) becomes (43):

(43) $agent(u) \wedge object(o) \supset$
$possession(f_{k\_s}(u,o)) \wedge generic\_consequence(f_{k\_a}(u,o), f_{k\_s}(u,o))$

This is the convention that underlies the logical interpretation of Prolog clauses; in practice, these Skolem terms aggregate data effectively into complex structures subject to constraint and unification.

First-order inference coincides with computability. That is why Prolog can implement any algorithm, for example. The result is that drawing inferences in first-order logic is undecidable; it can involve an arbitrarily long computation involving arbitrarily many instantiations of clauses and the construction of arbitrarily many data structures. The complexity of first-order inference can be a real practical problem.

First-order logic matches well with a rich ontology of generalized individuals. There are more powerful logics, but the expressive resources of these logics can generally be captured quite naturally in first-order logic by introducing abstract individuals for properties, sets, or situations. This is common practice in knowledge representation, as evidenced for example in current formalizations of knowledge, action and time (Levesque et al. 1997, Reiter 2001). One reason to avoid the more powerful logics is that some logical consequences of their intended models may not even be computable! It is still worth seeing how more expressive languages work, however. These languages can support some practical inference; and, more importantly, they can highlight insights about content and inference that do not come through so clearly in first-order translations or approximations.

1.5.3 Modal logic

First-order modal logic systematizes the inference about generalized individuals that we can draw from specified and limited information. For natural language, this gives us the ability to to formalize the causal, functional and intentional distinctions by which utterances construe the world. (Fitting and Mendelsohn 1998) is a recent accessible and comprehensive treatment of first-order modal logic from a general standpoint that combines mathematics, philosophy and computation. Other classic references for modal logic include (Chellas 1980, van Benthem 1983, Hughes and Cresswell 1996). Note that many properties of modal languages are especially clear for propositional languages; see (Blackburn et al. 2001) for a recent survey of propositional modal logic from the point of view of representation and (Fagin et al. 1995) for a study of propositional modal logic from the point of view of applications in computer science.

First-order modal logic extends first-order logic by adding MODAL OPERATORS; applying a modal operator to a sentence p gives a new sentence which describes the truth of p relative to a specific BODY OF INFORMATION or MODALITY. There are two kinds of modal operators: operators of NECESSITY, written $[\mathbf{INFO}]$ for body of information **INFO**; and operators of POSSIBILITY, written $\langle\mathbf{INFO}\rangle$ for body of information **INFO**. $[\mathbf{INFO}]p$ claims that p follows from **INFO**. $\langle\mathbf{INFO}\rangle p$ claims that p is consistent with **INFO**.

You can use modal operators to describe any of the wide range of bodies of information that figure in rich descriptions of the world. A particularly important motivation originates with causal reasoning, and its connection to our explanations and predictions about the world. Explanations and predictions inevitably involve an appeal to a limited body of information, because they show how a narrow range of facts and circumstances suffice to determine some observation or outcome. For example, we can explain why the grass is wet by observing that it has rained. The explanation is convincing because it draws on just a few facts and generalizations: the grass is outside; hence, the rain has fallen on the grass and wet it; and the subsequent meteorological conditions that have since prevailed can have done nothing to dry the grass. Since the explanation draws on this narrow body of information, it could apply generally to many similar situations, and thus locates our observations within a recurring natural pattern. Indeed, typical explanations presuppose a collection of generalizations that are always in force: a MECHANISM in the sense of (Pearl 2000). Here it is the dynamics of the weather. Explanations combine these generalizations with a simple description of a particular case; here it is the rain. The conclusion follows.

Causal relationships in the world embody these explanatory connections; thus, they must also express judgments based on limited information. In fact, the many diverse theories of causality are united in their focus on a scheme for identifying the presupposed background and the circumstantial facts that determine causal connections and give them explanatory power. For example, Lewis's account of causation in terms of counterfactuals locates the persistence of explanatory generalizations in his notion of similarity between possible worlds (Lewis 1973). Kratzer explores the circumstantial facts behind causal and counterfactual judgments, and argues that we "lump these facts together" in evaluating what else might have happened (Kratzer 1989). Meanwhile, in Goldman's analysis of the GENERATION relation of bringing about a result by means of an action, the result must follow from the action by a principled explanatory connection established by specified background circumstances (Goldman 1970). Pearl talks directly of mechanisms that embody generalizations and surgeries on mechanisms that determine a particular case (Pearl 2000).

As it turns out, though, typical natural language constructions explicitly specify the explanatory connections that link a cause to its effects. Resultative verbal expressions have been a standard example since (Fodor 1970). Let us return to the rain falling on the grass. A raindrop *r* falls and hits a blade of grass *g*, beating it down flat and making it wet. We would happily understand this as a single complex event, uniting a preparatory process *p* of falling with a culmination *c* of physical interaction involving an exchange of force and mass, and a consequent state *s* where the newly wet grass has taken on a new position.

With this event in mind, contrast the alternative descriptions we might give of it in (44).

(44) a. The rain made the grass flat and wet.
b. The rain hit the grass flat.
c. The rain hit the grass wet.
d. The rain wet the grass.
e. The rain wet the grass flat.
f. The rain caused the grass to become flat and wet.

With *made*, we can happily describe the event as a whole, highlighting any and all of its direct effects on the grass as in (44a). With *hit*, though, not all descriptions are appropriate: (44b) fits this situation and (44c) does not; indeed (44c) seems rather incomprehensible. Apparently, when we characterize a result from *hitting*, we link the result to a particular causal mechanism; the result must be a change of configuration accomplished by the ballistics of physical interaction (and nothing can get wet that way). Symmetrically, while (44d) describes this situation, (44e) does not; (44e) suggests that the grass has been brought down by the very weight of the water, not by the impulse of the raindrops as is actually the case. So when we characterize an object's change of state from *wetting*, we must link the change to the general properties of that object when wet. In comparison with (44c) or (44e), it makes sense to think of (44a) as reflecting the combination of causal mechanisms which offer stock perspectives on events. Famously, of course, only (44f) invites the hearer to supply an arbitrary mechanism leading indirectly from the action to its effect, as through an open-ended interpretation of causation. For example, when we describe a fanciful indoor greenhouse designed with an array of lights and sprinklers and sensors to mirror outdoor precipitation automatically, only (44f) would fit an event where the rain triggers the sprinklers of the greenhouse and indirectly accomplishes the watering of the grass inside.

A typical first-order representation of the event, as in (45), effectively captures only example (44f).

(45) $result(c,s) \wedge wet(s,g) \wedge flat(s,g)$

(45) just says that a state s of g being wet and flat is somehow the result of culmination c. Observe that changing the relation between c and s does not help to distinguish the different causal semantics in (45); as long as the other conjuncts describe the same state s, they will have the same truth value.

To articulate the distinction, we have to use modal operators. For example, we can introduce an operator $[\textbf{BALLISTICS}]$ describing what follows from the causal mechanism involved in hitting and other interactions of physics; we can introduce an operator $[\textbf{INUNDATION}]$ describing wetness by its causal mechanisms. Now we can contrast (46a) and (46b) meaningfully both with one another and with (45).

(46) a. $[\textbf{BALLISTICS}](result(c,s) \wedge flat(s,g))$

b. $[\textbf{INUNDATION}](result(c,s) \wedge wet(s,g))$

(46a) corresponds to (44b); (46b) corresponds to (46d). In modal logic, even when all three (46a), (46b) and (45) are true, it may be that $[\textbf{BALLISTICS}](wet(s,g))$ and $[\textbf{INUNDATION}](flat(s,g))$ are false, if these aspects of s do not follow for the right reason.

Modal representations might look rather exotic. But in fact, cutting-edge knowledge representation systems increasingly treat collections of models and working assumptions as first-class elements of the logic, since (Guha 1991). The most common device is a variant and extension of a modal operator called, rather ambiguously, a CONTEXT (McCarthy 1993). There are also many ways to approximate model inference effectively in first-order representations. When you do not need logic within the scope of modal operators, you can rewrite the modal formulas using structured atomic propositions; the use of (47) for (46a) suggests the correspondence.

(47) $nec(ballistics, result(c,s)) \wedge nec(ballistics, flat(s,g))$

Meanwhile, when logical inferences are required within modal scopes but typically not between scopes, we can add a POSSIBLE WORLD or SITUATION argument to predicates, as suggested by modal semantics (Kripke 1963, Fitting 1983, Wallen 1990) and AI techniques (McCarthy and Hayes 1969, Reiter 2001).

(48) $result(c,s,pw(ballistics,w,reality)) \wedge flat(s,g,pw(ballistics,w,reality))$

(48) suggests this representation—if you are familiar with possible worlds semantics, you will recognize that the term $pw(ballistics,w,reality)$ can represent an arbitrary possible world reached from the real world by a transition w of ballistics-accessibility. For reasoning about possibilities, transitions are represented by Skolem terms rather than variables.

The opportunities for using these modal representations are very broad.

Our explanations of function, as embodied in our characterizations of objects as artifacts, involve a parallel dependence on explicitly-restricted bodies of information. Think of a wooden bar stool, built solidly, with a round top cut with the gentle indentations of a seat. When the object stands under a window with a sprawling spider-plant on top, it does not thereby become a plant stand (except as a jest); it remains a stool used as a plant stand. We do not characterize the stool based on its successful function in its full present context. Instead, we assess the stool on its own, restricting our attention to its individual properties and the broader principles that apply generally to similar objects. Concretely, we continue to judge the object as stool, for sitting, because we base our judgments about the object on a restricted subset of what we know about it—properties such as its shape, its structure, and its design—information which in fact determines what the object can do, by a public and persistent standard, and which thereby allows us to explain the object, rather than just redescribing its actual condition.

Moreover, many individuals come with genuine perspectives of the world, including agents that see or know the world they experience; and images, recordings and other sources that depict events in history. We can describe the content of these real-world representations using the same realistic bodies of information that we use to formalize judgments of causation and function (Halpern and Moses 1985, Cohen and Levesque 1990). By further introducing bodies of hypothetical information, describing consistent scenarios potentially unrelated to the world as it is, we gain the ability to describe the fallible beliefs and unrealized goals of agents, and the more rarefied contents of fiction and imagination.

1.5.4 Developing and Using Knowledge

Description logic, first-order logic and modal logics are some of the most-used formalisms for knowledge representation, but they are not the only ones. Logics exist at all level of complexity, aimed at providing the right expressive power to automate important inferences as conveniently and efficiently as possible. Hybrid logic (Blackburn 2000, Areces et al. 2001), for example, stands between description logic and first-order logic. As in description logic, it expresses general knowledge of concepts and offers inference algorithms to decide how concepts are related; however, its formulas do allow you, in restricted ways, to refer back to individuals you have already described. Higher-order logic, by contrast, adds even more expressive power—its terms can stand for arbitrary functions and properties. Despite the complexity, higher-order logic is amenable to logic programming implementations comparable to Prolog (Nadathur and Miller 1998) and supports the inference required to characterize general relationships in natural language discourse such as ellipsis and parallelism (Dalrymple et al. 1991, Gardent 1999).

Finally, there are formalisms whose inference is based on making assumptions and reconciling competing explanations: nonmonotonic logics. See e.g., (Gabbay et al. 1994, Brewka et al. 1997). One role for these formalisms in language engineering is to capture probabilistic inference with tools continuous with deductive reasoning (Poole 1993); another is to specify default inferences that can flesh out interpretations in salient ways when more specific information is unavailable (Asher and Lascarides 2003). Clearly, there are lots of alternative representations to choose from! You can expect to use the representation that best supports the inferences that are most important to your application.

Even within a specific representation, however, formalization can seem unconstrained. For language engineering, a coarse high-level model, like the constraint-satisfaction model sketched in Section 1.3, can provide important further guidance. Any general model allows formalization to proceed in tandem with an assessment of the impact of knowledge on the interpretation of domain utterances. A simple but executable model in fact makes precise predictions about problems in the treatment of domain utterances such as ambiguities and missing information.

As the description of domain information starts to converge, you can explore more flexible ways of incorporating knowledge into the system's decision-making. You can start to write rules that specify the system's behavior in terms of the system's knowledge; see (Shoham 1993, Fagin et al. 1995, Poole et al. 1998). You can of course continue to test out these rules on the development data for the system, to help track the viability and progress of your developing strategies. Alternatively, you can start to associate individual examples with features of the system's knowledge, and thereby supply background knowledge to automatic machine-learning experiments. This gives learners access to underlying similarities of instances that are potentially more meaningful than surface correlations. Learned models, rules and classifiers can then generalize in more principled ways.

1.6 Closing Thoughts

The methods of Sections 1.3, 1.4 and 1.5 make it possible to articulate and formalize information about the world for language use. But in many respects, what we have learned serves to quantify the received wisdom that such knowledge is extremely expensive, not to challenge it. It is quite difficult to formulate representations that are consistent, and even more difficult to formulate ones that are true! And for an actual application, truth is not enough—what you say has to be the right information to do useful work in the system's actual environment.

Understanding knowledge representation as a practical methodology can minimize the difficulty. You can take care to do automatic checks to make sure you describe the world and its inferences consistently. You can start from a representative sample of real-world situations to make sure that you describe them correctly. And you can check as you go along that the representations you develop are in fact used economically in describing these real-world situations.

Nevertheless, an appreciation of the difficulties of knowledge representation is required to put knowledge representation to work most effectively. Typical natural language systems are built to collaborate with partners who bring their own prodigious and inimitable real-world expertise to the interaction—human users. It is foolish to attempt to duplicate their strengths in a system when you could just as well build on those strengths. In using speech systems, for example, users often can quickly learn not to expect inferences of the system, and accommodate by spelling out facts explicitly step-by-step. And in using text systems, users can often quickly learn to sieve through the rough suggestions of the system to find those that mesh with their task and context. What knowledge you need to represent depends on which burdens users are willing to share, and which can be readily lifted from them.

As language engineering matures, we expect more and more systems whose knowledge representation is a distinctive ingredient in their success—accessible interfaces, for users who do not have the knowledge to bear the collaborative burden in interacting with a system, as in (Yates et al. 2003) for example; or decision-support systems, where users and system must both have their act together, because efficiency of interaction is of the utmost importance, as in (Allen et al. 2001) for example. In these cases, knowledge and inference brings natural utterances, flexible and robust processing, and efficient dialogue.

1.7 Further information

If you are representing knowledge for natural language systems, you can find useful ideas across a wide range of work in computer science, psychology, linguistics and philosophy (as the preceding discussion should already have made clear). You can expect to consult this literature yourself for inspiration. But typically, you will find only inspiration, not concrete suggestions, for the research you find will almost always address itself to the narrow concerns of one academic discipline, rather than to more general enterprise of computational modeling of communication, representation and reality.

People who aim to build things that really work are computer scientists, and no other discipline offers the mix of practice and precision that natural language engineers must achieve. The web is a great archive of tools and resources for knowledge representation. At the time of writing, Patrick Lamblix maintains pointers to description logic information, including systems, applications and tutorials (49a). Michael Kohlhase and Carolyn Talcott maintain pointers to reasoning systems for more expressive logics, including first-order, higher-order and modal logic (49b).

(49) a. http://www.ida.liu.se/labs/iislab/people/patla/DL/index.html
b. http://www-formal.stanford.edu/clt/ARS/systems.html

Tools and resources for data management in computational linguistics and ontology management generally, as described in (Noy, Chapter 5, this volume) and (Megerdoomian, Chapter 6, this volume), will also be helpful.

To get acquainted with more general results of knowledge representation in computer science, you might consult (Davis 1990, Sowa 1999), which provide introductions to knowledge representation itself, and (Genesereth and Nilsson 1987, Poole et al. 1998, Minker 2000), which situate logic and knowledge representation more broadly within the enterprise of building intelligent systems. Natural language engineers could also benefit from introductions to formal specification for and in software systems such as (Potter et al. 1996, Huth and Ryan 1999). Nowadays, increasing effort in knowledge representation is geared towards the development of reusable collections of concepts and terminology for large knowledge-based systems; see (Noy, Chapter 5, this volume) or (Fensel et al. 2002).

In AI, a major thrust of knowledge representation research has been the treatment of action and change, and the FRAME PROBLEM at is heart—capturing the common-sense notion that things stay the same when unaffected by ongoing events. One strategy is to adopt special logics of persistence and change; see for example (Sandewall 1994, Shanahan 1997). An alternative is to articulate assumptions about persistence explicitly; see (Schubert 1990, Reiter 1991, Reiter 2001). (Steedman 1997) offers an excellent discussion of this literature in its implications for natural language interpretation.

Computational linguistics has also seen extensive research into representing discourse context, and the abstract concepts that are implicated in the meanings of discourse-sensitive constructions. Such concepts include information units, the open questions raised by the discourse, and their possible and actual answers (Ginzburg 1995, van Kuppevelt 1995, van Kuppevelt 1996, Roberts 1996, Steedman 2000). They include discourse relations, the inferential relationships among propositions that connect discourse together; see (Kehler 2001) and references therein. And they include alternative sets, collections of salient entities and propositions which the concerns of the discourse naturally bracket together, and which may therefore need to be distinguished from and related to one another (Hirschberg 1985, Rooth 1985, Rooth 1992, Prevost and Steedman 1994, Bierner and Webber 2000). Increasingly, researchers have succeeded in organizing such concepts into comprehensive representations of information state for dialogue applications; see (Larsson and Traum 2000, Matheson et al. 2000, Ginzburg and Cooper 2001). Implementing more formal theories, particularly from linguistic semantics, is challenging, but tutorials are available for it, notably (Blackburn and Bos 2002a, Blackburn and Bos 2002b). More formal proposals are invaluable for the close connection between syntax and semantics they provide. Such research naturally involves a close connection with other aspects of grammar development, as described in (Butt and Holloway King, Chapter 4, this volume). The papers in (Iwańska and Shapiro 2000) directly explore aspects of the overlap between knowledge representation and natural language.

The formalism of AI methods can obscure the real-world content of researchers' proposals—the true and important inferences about the world that agents need to be successful. By comparison, the psychology literature teems with case studies of the real-world inferences people draw: on objects and categories, see for example (Gopnik and Meltzoff 1997); on belief and intention and mind more generally, see for example (Sperber 2000, Malle et al. 2001); on language, see for example (Bloom 2000). However, it largely remains to systematize these case studies with the precision and coverage necessary for implementation. The cognitive semantics literature, as represented for example by (Langacker 1999, Talmy 2000), offers a similarly intriguing catalogue of inferences in language interpretation—again, without the explicitness about meaning required for implementation (sometimes to the point of rejecting formalism and computation outright). For this reason, the rather comparable proposals of Jackendoff, e.g. (1983, 1990), and Pustejovsky, e.g., (1995), have been much more influential in computational linguistics. In addition, many of the best ideas about language and the world are to be found in the philosophy of language, and original sources such as (Davidson 2002b, Grice 1991, Kripke 1982) can be surprisingly accessible.

1.8 Acknowledgments

Thanks for comments and discussion to Alex Borgida, David DeVault, Ali Farghaly, Mark Steedman, John Velman and Lan Yu. The preparation of this chapter was supported in part by NSF CISE CDA 9818322, and a sabbatical leave from Rutgers.

References

Agre, Philip E. 1997. *Computation and Human Experience*. Cambridge.

Allen, James, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In *Proceedings of Intelligent User Interfaces (IUI)*.

Allen, James F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11):832–843.

Allen, James F., and C. Raymond Perrault. 1980. Analyzing intention in utterances. *Artificial Intelligence* 15:143–178.

Alur, Rajeev, and Thomas A. Henzinger. 1999. Reactive modules. *Formal Methods in System Design* 15:7–48.

Alur, Rajeev, Thomas A. Henzinger, F. Y. C. Mang, Shaz Qadeer, Sriram K. Rajamani, and Serdar Tasiran. 1998. Mocha: Modularity in model checking. In *Proceedings of the Tenth International Conference on Computer-aided Verification (CAV)*, 521–525. LNCS, No. 1427. Springer.

Ankolekar, Anupriya, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Drew McDermott, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terence Payne, and Katia Sycara. 2002. DAML-S: Web service description for the Semantic Web. In *Proceedings of the International Semantic Web Conference*.

Areces, Carlos, Patrick Blackburn, and Maarten Marx. 2001. Hybrid Logic: Characterization, interpolation and complexity. *Journal of Symbolic Logic* 66(3):977–1010.

Artale, Alessandro, Enrico Franconi, Nicola Guarino, and Luca Pazzi. 1996. Part-Whole relations in object-centered systems: An overview. *Data and Knowledge Engineering* 20:347–383.

Asher, Nicholas. 1993. *Reference to Abstract Objects in Discourse*. Kluwer.

Asher, Nicholas, and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge.

Baader, Franz, Diego Calvanese, Deborah McGuiness, Daniele Nardi, and Peter Patel-Schneider (ed.). 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge.

Bach, Emmon. 1986. The algebra of events. *Linguistics and Philosophy* 9:5–16.

Barr, Valerie, and Alvaro Gonzalez. 2000. Valildation and verification of intelligent systems—what are they and how are they different? *Journal of Experimental and Theoretical Artificial Intelligence* 12(4).

Barr, Valerie, and Judith Klavans. 2001. Verification and validation of language processing systems: Is it evaluation? In *ACL 2001 Workshop on Evaluation Methodologies for Language and Dialogue Systems*, 34–40.

Barwise, Jon, and John Perry. 1983. *Situations and Attitudes*. MIT.

Bierner, Gann, and Bonnie Webber. 2000. Inference through Alternative-set semantics. *Journal of Language and Computation* 1:277–292.

Blackburn, Patrick. 2000. Representation, reasoning and relational structures. *Logic Journal of the IGPL* 8(5):653–679.

Blackburn, Patrick, and Johan Bos. 2002a. Representation and Inference for Natural Language: A First Course in Computational Semantics. http://www.comsem.org.

Blackburn, Patrick, and Johan Bos. 2002b. Working with Discourse Representation Theory: An Advanced Course in Computational Semantics. http://www.comsem.org.

Blackburn, Patrick, Maarten De Rijke, and Yde Venema. 2001. *Modal Logic*. Cambridge.

Bloom, Paul. 2000. *How Children Learn the Meanings of Words*. MIT.

Bos, Johan. To appear. Implementing the binding and accommodation theory for anaphora resolution and presupposition. *Computational Linguistics*.

Brachman, Ronald, Deborah McGuinness, Peter Patel Schneider, Lori Alperin Resnick, and Alexander Borgida. 1990. Living with CLASSIC: when and how to use a KL-ONE-like language. In *Principles of Semantic Networks*, ed. J. Sowa. Morgan Kaufmann.

Bratman, Michael E. 1987. *Intention, Plans, and Practical Reason*. Harvard.

Brennan, Susan E. To appear. The vocabulary problem in spoken language systems. In *Automatic Spoken Dialog Systems*, ed. Susan Luperfoy. MIT.

Brewka, Gerhard, Jürgen Dix, and Kurt Konolige. 1997. *Nonmonotonic Reasoning: An Overview*. CSLI.

Bunt, Harry. 2000. Dialogue pragmatics and context specification. In *Abduction, Belief and Context in Dialogue: Studies in Computational Pragmatics*, ed. Harry Bunt and William Black. 81–150. John Benjamin.

Carlson, Gregory. 1992. Natural kinds and common nouns. In *Handbook of Semantics*, ed. A. von Stechow and D. Wunderlich. 370–398. de Gruyter.

Carlson, Gregory N. 1980. *Reference to Kinds in English*. Garland.

Charniak, Eugene. 1973. Jack and Janet in search of a theory of knowledge. In *Proceedings of IJCAI*, 337–343.

Chellas, Brian F. 1980. *Modal Logic: An Introduction*. Cambridge.

Clark, Herbert H. 1996. *Using Language*. Cambridge.

Cohen, Philip R., and Hector J. Levesque. 1990. Intention is choice with commitment. *Artificial Intelligence* 42:213–261.

Cotton, Scott, and Steven Bird. 2002. An integrated framework for treebanks and multilayer annotation. In *Third International Conference on Language Resources and Evaluation*.

Crangle, Colleen. 1989. On saying 'stop' to a robot. *Language and Communication* 9(1):23–33.

Crangle, Colleen, and Patrick Suppes. 1994. *Language and Learning for Robots*. CSLI.

Cresswell, Max J. 1985. *Adverbial Modification: Interval Semantics and its Rivals*. Reidel.

Dahlbäck, Nils, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz studies—Why and how. In *Intelligent User Interfaces*, 193–200.

Dalrymple, Mary, Stuart Shieber, and Fernando Pereira. 1991. Ellipsis and higher-order unification. *Linguistics and Philosophy* 14(4).

Davidson, Donald. 2002a. Agency. In *Essays on Actions and Events*. 43–62. Oxford.

Davidson, Donald. 2002b. *Essays on Actions and Events*. Oxford. Second edition.

Davidson, Donald. 2002c. The individuation of events. In *Essays on Actions and Events*. 163–180. Oxford.

Davis, Ernest. 1990. *Representations of Commonsense Knowledge*. Morgan Kaufmann.

Di Eugenio, Barbara, and Bonnie Webber. 1996. Pragmatic overloading in natural language instructions. *Internationl Journal of Expert Systems* 9(2):53–84.

Dowty, David R. 1979. *Word Meaning and Montague Grammar*. Reidel.

Fagin, Ronald, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. 1995. *Reasoning About Knowledge*. MIT.

Fensel, Dieter, James A. Hendler, Henry Lieberman, and Wolfgang Wahlster (ed.). 2002. *Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential*. MIT.

Fitting, Melvin. 1983. *Proof Methods for Modal and Intuitionistic Logics*. Synthese Library, Vol. 169. Reidel.

Fitting, Melvin, and Richard L. Mendelsohn. 1998. *First-order Modal Logic*. Synthese Library, Vol. 277. Kluwer.

Fodor, Jerry A. 1970. Three reasons for not deriving ‘kill’ from ‘cause to die’. *Linguistic Inquiry* 1:429–438.

Fodor, Jerry A. 1998. *Concepts: Where Cognitive Science Went Wrong*. Oxford.

Furnas, G. W., T. K. Landauer, L. M. Gomez, and S. T. Dumais. 1987. The vocabulary problem in human-system communications. *Communications of the ACM* 30:964–971.

Gabbay, Dov, Christopher J. Hogger, and J. A Robinson (ed.). 1994. *Handbook of Logic in Artificial Intelligence, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*. Oxford.

Gallier, Jean H. 1986. *Logic for Computer Science: Foundations of Automated Theorem Proving*. Harper and Row.

Gardent, Claire. 1999. Unifying Parallels. In *Proceedings of ACL*, 49–56.

Genesereth, Michael R., and Nils J. Nilsson. 1987. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann.

Gillon, Brendan. 1987. The Readings of Plural Noun Phrases in English. *Linguistics and Philosophy* 10(2):199–299.

Ginzburg, Johnathan. 1995. Resolving Questions, I & II. *Linguistics and Philosophy* 18(5;6):459–527;567–609.

Ginzburg, Jonathan, and Robin Cooper. 2001. Clarification, Ellipsis and the Nature of Contextual Updates in Dialogue. King’s College London and Göteborg University.

Girard, Jean-Yves, Yves Lafont, and Paul Taylor. 1989. *Proofs and Types*. Cambridge.

Goldman, Alvin I. 1970. *A Theory of Human Action.* Prentice Hall.

Gopnik, Alison, and Andrew N. Meltzoff. 1997. *Words, Thought and Theories*. MIT.

Grice, H. P. 1957. Meaning. *The Philosophical Review* 66(3):377–388.

Grice, H. P. 1975. Logic and Conversation. In *Syntax and Semantics III: Speech Acts*, ed. P. Cole and J. Morgan. 41–58. Academic Press.

Grice, H. Paul. 1991. *Studies in the Way of Words*. Harvard.

Grosso, William E., Henrik Eriksson, Ray W. Ferguson, John H. Gennari, Samson W. Tu, and Mark A Musen. 1999. Knowledge modeling at the millennium: The design and evolution of Protégé-2000. Technical Report SMI-1999-0801. Stanford Medical Informatics.

Grosz, Barbara, and Candace Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics* 12:175–204.

Grosz, Barbara J., and Candace L. Sidner. 1990. Plans for discourse. In *Intentions in Communication*, ed. Philip Cohen, Jerry Morgan, and Martha Pollack. 417–444. MIT.

Guha, Ramanathan V. 1991. *Contexts: A Formalization and Some Applications*. Doctoral dissertation, Stanford. TR STAN-CS-91-1399.

Haddock, Nicholas. 1989. *Incremental Semantics and Interactive Syntactic Processing*. Doctoral dissertation, Edinburgh.

Halpern, Joseph Y., and Yoram Moses. 1985. A guide to the modal logics of knowledge and belief: preliminary draft. In *Proceedings of IJCAI*, 480–490.

Higginbotham, James. 2000. On events in linguistic semantics. In *Speaking of Events*, ed. James Higginbotham, Fabio Pianesi, and Achille C. Varzi. 49–79. Oxford.

Hirschberg, Julia. 1985. *A Theory of Scalar Implicature*. Doctoral dissertation, University of Pennsylvania.

Hobbs, Jerry, Mark Stickel, Douglas Appelt, and Paul Martin. 1993. Interpretation as abduction. *Artificial Intelligence* 63:69–142.

Hobbs, Jerry R. 1978. Resolving pronoun references. *Lingua* 44:311–338.

Hobbs, Jerry R. 1985. Ontological promiscuity. In *Proceedings of ACL*, 61–69.

Hobbs, Jerry R. 1995. Sketch of an ontology underlying the way we talk about the world. *International Journal of Human-Computer Studies* 43:819–830.

Hoffman, Donald, and Whitman Richards. 1984. Parts of recognition. *Cognition* 18:65–96.

Horacek, Helmut. 2002. Aggregation with Strong Regularities and Alternatives. In *International Natural Language Generation Conference*, 105–112.

Hughes, G. E., and M. J. Cresswell. 1996. *A New Introduction to Modal Logic*. Routledge.

Huth, Michael, and Mark Ryan. 1999. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge.

Isard, Stephen. 1974. What would you have done if... *Theoretical Linguistics* 1:233–255.

Iwańska, Łucja M., and Stuart C. Shapiro (ed.). 2000. *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language*. MIT.

Jackendoff, Ray. 1991. Parts and boundaries. *Cognition* 41:9–45.

Jackendoff, Ray S. 1983. *Semantics and Cognition*. MIT.

Jackendoff, Ray S. 1990. *Semantic Structures*. MIT.

Kaplan, David. 1989. Demonstratives. In *Themes from Kaplan*, ed. J. Almog, J. Perry, and H. Wettstein. Oxford.

Kehler, Andrew. 2001. *Coherence, Reference and the Theory of Grammar*. CSLI.

Keil, Frank C. 1989. *Concepts, Kinds and Cognitive Development*. MIT.

Kratzer, Angelika. 1989. An investigation of the lumps of thought. *Linguistics and Philosophy* 12:607–653.

Krifka, Manfred, Francis Jeffrey Pelletier, and Gregory Carlson. 1995. Genericity: An introduction. In *The Generic Book*, ed. G. N. Carlson and F. J. Pelletier. 1–124. Chicago.

Kripke, Saul. 1982. *Naming and Necessity*. Harvard.

Kripke, Saul A. 1963. Semantical analysis of modal logic. I. Normal modal propositional calculi. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 9:67–96.

Lakoff, George, and Mark Johnson. 1980. *Metaphors We Live By*. Chicago.

Langacker, Ronald W. 1999. *Foundations of Cognitive Grammar.* Stanford.

Larson, Richard. 1986. Implicit arguments in situation semantics. *Linguistics and Philosophy* 11:169–201.

Larson, Richard, and Gabriel Segal. 1995. *Knowledge of Meaning: An Introduction to Semantic Theory*. MIT.

Larsson, Staffan, and David Traum. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering* 6:323–340.

Lascarides, Alex, and Nicholas Asher. 1991. Discourse relations and defeasible knowledge. In *Proceedings of ACL*, 55–62.

Lepore, Ernest. 2000. *Meaning and argument: an introduction to logic through language*. Blackwell.

Levesque, Hector J. 1984. Foundations of a functional approach to knowledge representation. *Artificial Intelligence* 23(2):155–212.

Levesque, Hector J., Raymond Reiter, Yves Lespérance, Fangzhen Lin, and Richard Scherl. 1997. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming* 31:59–84.

Lewis, David. 1979. Scorekeeping in a language game. In *Semantics from Different Points of View*, ed. R. Bäuerle, U. Egli, and A. von Stechow. 172–187. Springer.

Lewis, David K. 1973. *Counterfactuals*. Harvard.

Leyton, Michael. 1992. *Symmetry, Causality, Mind*. MIT.

Litman, Diane J., and James F. Allen. 1990. Discourse processing and commonsense plans. In *Intentions in Communication*, ed. Philip R. Cohen, Jerry Morgan, and Martha E. Pollack. 365–388. MIT.

Lochbaum, Karen E. 1998. A collaborative planning model of intentional structure. *Computational Linguistics* 24(4):525–572.

Mackworth, Alan. 1987. Constraint satisfaction. In *Encyclopedia of Artificial Intelligence*, ed. S.C. Shapiro. 205–211. John Wiley and Sons.

Malle, Bertram F., Louis J. MOses, and Dare A. Baldwin (ed.). 2001. *Intentions and Intentionality: Foundations of Social Cognition.* MIT.

Marr, David. 1982. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman.

Matheson, Colin, Massimo Poesio, and David Traum. 2000. Modelling grounding and discourse obligations using update rules. In *Proceedings of the North American Assocation of Computational Linguistics*.

McCarthy, J., and P.J. Hayes. 1969. Some philosophical problems from the standpoint of Artificial Intelligence. In *Machine Intelligence*, ed. B. Meltzer and D. Michie. 473–502. Edinburgh.

McCarthy, John. 1993. Notes on formalizing context. In *Proceedings of IJCAI*, 555–560.

McGuiness, Deborah L., Lori Alperin Resnick, and Charles Isbell. 1995. Description logic in practice: A CLASSIC application. In *Proceedings of IJCAI*.

Mellish, Christopher S. 1985. *Computer Interpretation of Natural Language Descriptions*. Ellis Horwood.

Minker, Jack (ed.). 2000. *Logic-Based Artificial Intelligence*. Kluwer.

Nadathur, Gopalan, and Dale Miller. 1998. Higher-order logic programming. In *Handbook of Logics for Artificial Intelligence and Logic Programming*, ed. Dov Gabbay, C. J. Hogger, and J. A. Robinson. 499–590. Oxford.

Neale, Stephen. 1990. *Descriptions*. MIT.

Neale, Stephen. 1995. The philosophical significance of Gödel’s slingshot. *Mind* 104:761–825.

Newell, Allen. 1982. The knowledge level. *Artificial Intelligence* 18:87–127.

Norvig, Peter, and Robert Wilensky. 1990. A critical evaluation of commensurable abduction models for semantic interpretation. In *Proceedings of COLING*, 225–230.

Nunberg, Geoffrey. 1979. The non-uniqueness of semantic solutions: Polysemy. *Linguistics and Philosophy* 3(2):143–184.

Ortiz, Charles L. 1999. Explanatory update theory: applications of counterfactual reasoning to causation. *Artificial Intelligence* 108(1-2):125–178.

Pan, Shimei, and Wubin Weng. 2002. Designing a speech corpus for instance-based spoken language generation. In *Second International Natural Language Generation Conference*, 49–56.

Pearl, Judea. 2000. *Causality: Models, Reasoning and Inference*. Cambridge.

Pollack, Martha E. 1990. Plans as complex mental attitudes. In *Intentions in Communication*, ed. Philip Cohen, Jerry Morgan, and Martha Pollack. 77–103. MIT.

Poole, David. 1993. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence* 64(1):81–129.

Poole, David, Alan Mackworth, and Randy Goebel. 1998. *Computational Intelligence: A Logical Approach*. Oxford.

Potter, Ben, Jane Sinclair, and David Till. 1996. *An Introduction to Formal Specification and Z.* Prentice Hall. Second edition.

Prevost, Scott, and Mark Steedman. 1994. Specifying Intonation from Context for Speech Synthesis. *Speech Communication* 15:139–153.

Priest, Graham. 1994. The structure of the paradoxes of self-reference. *Mind* 103:25–34.

Pustejovsky, James. 1991. The generative lexicon. *Computational Linguistics* 17(3):409–441.

Pustejovsky, James. 1995. *The Generative Lexicon*. MIT.

Putnam, Hilary. 1975. *Mind, Language and Reality*. Cambridge.

Ratnaparkhi, Adwait. 2000. Trainable methods for surface natural language generation. In *First meeting of the North American Chapter of the Association for Computational Linguistics*, 194–201.

Reiter, Raymond. 1991. The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. In *Artificial Intelligence and Mathematical Theory of Computation*, ed. Vladimir Lifschitz. 359–380. Academic Press.

Reiter, Raymond. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems.* MIT.

Roberts, Craige. 1989. Modal subordination and pronominal anaphora in discourse. *Linguistics and Philosophy* 12(6):683–721.

Roberts, Craige. 1996. Information structure in discourse: Towards an integrated formal theory of pragmatics. In *Ohio State University Working Papers in Linguistics*, 91–136.

Rooth, Mats. 1985. *Association with focus*. Doctoral dissertation, University of Massachusetts.

Rooth, Mats. 1992. A theory of focus interpretation. *Natural Language Semantics* 1(1):75–116.

Saeboe, Kjell Johan. 1996. Anaphoric presuppositions and zero anaphora. *Linguistics and Philosophy* 19(2):187–209.

Sandewall, Erik. 1994. *Features and Fluents: Representation of Knowledge about Dynamical Systems.* Oxford.

Schank, Roger. 1980. Language and memory. *Cognitive Science* 4(3):243–284.

Schank, Roger, and Robert Abelson. 1977. *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum.

Schein, Barry. 1993. *Plurals and Events*. MIT.

Scholl, Brian J. 2001. Objects and attention: the state of the art. *Cognition* 80:1–46.

Schubert, Lenhart K. 1990. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. In *Knowledge Representation and Defeasible Reasoning*, ed. H. E. Kyburg, R. P. Loui, and G. N. Carlson. 23–67. Kluwer.

Schuler, William. 2001. Computational properties of environment-based disambiguation. In *Proceedings of ACL*, 466–473.

Schwarzschild, Roger. 1994. Plurals, presuppositions, and the sources of distributivity. *Natural Language Semantics* 2:201–248.

Schwarzschild, Roger. 1996. *Pluralities*. Kluwer.

Searle, John R. 1975. Indirect speech acts. In *Syntax and Semantics III: Speech Acts*, ed. P. Cole and J. Morgan. 59–82. Academic Press.

Searle, John R. 1997. *The Construction of Social Reality*. Simon and Schuster.

Shanahan, Murray. 1997. *Solving the Frame Problem.* MIT.

Shieber, Stuart M. 1993. The problem of logical form equivalence. *Computational Linguistics* 19(1):179–190.

Shoham, Yoav. 1993. Agent-oriented programming. *Artificial Intelligence* 60:51–92.

Siskind, Jeffrey Mark. 2001. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artficial Intelligence Research* 15:31–90.

Sowa, John F. 1999. *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks/Cole.

Spelke, Elizabeth. 1994. Initial knowledge: six suggestions. *Cognition* 50:431–445.

Sperber, Dan (ed.). 2000. *Metarepresentations: An Interdisciplinary Perspective*. Oxford.

Sperber, Dan, and Deirdre Wilson. 1986. *Relevance: Communication and Congition*. Harvard.

Stalnaker, Robert. 1998. On the representation of context. *Journal of Logic, Language and Information* 7:3–19.

Steedman, Mark. 1997. Temporality. In *Handbook of Logic and Language*, ed. Johan van Benthem and Alice ter Meulen. 895–935. Elsevier.

Steedman, Mark. 2000. Information structure and the syntax-phonology interface. *Linguistic Inquiry* 31:649–689.

Sterling, Leon, and Ehud Shapiro. 1994. *The Art of Prolog*. MIT. Second edition.

Stone, Matthew. 2000. On identifying sets. In *First International Confernence on Natural Language Generation*, 116–123.

Stone, Matthew. 2003. Communicative Intentions and Conversational Processes in Human-Human and Human-Computer Dialogue. In *World-Situated Language Use: Psychological, Linguistic and Computational Perspectives on Bridging Product and Action Traditions*, ed. John Trueswell and Michael Tanenhaus. MIT.

Stone, Matthew, Christine Doran, Bonnie Webber, Tonia Bleam, and Martha Palmer. 2001. Microplanning with Communicative Intentions: The SPUD System. *http://www.arxiv.org/abs/cs.CL/0104022*.

Stone, Matthew, and Bonnie Webber. 1998. Textual economy through close coupling of syntax and semantics. In *Proceedings of International Natural Language Generation Workshop*, 178–187.

Talmy, Leonard. 2000. *Toward a Cognitive Semantics*. MIT.

Thomason, Richmond H. 1990. Accommodation, meaning and implicature. In *Intentions in Communication*, ed. Philip R. Cohen, Jerry Morgan, and Martha E. Pollack. 325–363. MIT.

Thomason, Richmond H. 1999. Type-theoretic foundations for context, part I: Contexts as complex type-theoretic objects. In *Modeling and Using Context*, ed. P. Bouquet, L. Serafini, P. Brezillon, M. Benerecetti, and F. Castellani. 352–374. LNAI 1688. Springer.

van Benthem, Johan F. A. K. 1983. *Modal Logic and Classical Logic*. Bibliopolis.

van Deemter, Kees, and Rodger Kibble. 2000. On Coreferring: Coreference in MUC and related annotation standards. *Computational Linguistics* 26(4):629–637.

van den Berg, M. H. 1993. Full dynamic plural logic. In *Proceedings of the Fourth Symposium on Logic and Language*, ed. K. Bimbó and A. Máté. Budapest.

van den Berg, M. H. 1996. Generalized dynamic quantifiers. In *Quantifiers, Logic and Language*, ed. J. van der Does and J. van Eijk. CSLI.

van Kuppevelt, Jan. 1995. Discourse structure, topicality and questioning. *Journal of Linguistics* 31(1):109–147.

van Kuppevelt, Jan. 1996. Inferring from topics—Scalar implicatures as topic-dependent inferences. *Linguistics and Philosophy* 19(4):393–443.

Vendler, Zeno. 1967. *Linguistics in Philosophy*. Cornell.

Verkuyl, Henk, and Jaap van der Does. 1994. The semantics of plural noun phrases. In *Generalized Quantifier Theory and Applications*, ed. Jaap van der Does and Jan van Eijck. 403–439. NLLI.

Vieira, Renata, and Massimo Poesio. 2000. An empirically based system for processing definite descriptions. *Computational Linguistics* 26(4):539–594.

Vygotsky, Lev Semenovich. 1962. *Thought and language*. MIT.

Walker, Marilyn A. 2000. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research* 12:387–416.

Walker, Marilyn A., Owen Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech and Language* 16(3–4):409–433.

Wallen, Lincoln A. 1990. *Automated Proof Search in Non-Classical Logics: Efficient Matrix Proof Methods for Modal and Intuitionistic Logics*. MIT.

Webber, Bonnie. 1998. Instructing Animated Agents: Viewing Language in Behavioral Terms. In *Multimodal Human-Computer Communication: Systems, Techniques and Experiments*, ed. Harry Bunt, Robert-Jan Beun, and Tijn Borghuis. LNAI, Vol. 1374, 89–100. Springer.

Webber, Bonnie, Norm Badler, Barbara Di Eugenio, Chris Geib, Libby Levison, and Michael Moore. 1995. Instructions, intentions and expectations. *Artificial Intelligence* 73:253–259.

Webber, Bonnie Lynn. 1983. So what can we talk about now? In *Computational Models of Discourse*, ed. Michael Brady and Robert C. Berwick. 331–371. MIT.

Whorf, Benjamin Lee. 1956. *Language, Thought and Reality*. MIT.

Witten, Ian H., and Eibe Frank. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

Yates, Alexander, Oren Etzioni, and Daniel Weld. 2003. A reliable natural language interface to household appliances. In *Proceedings of Intelligent User Interfaces (IUI)*.

Index