MAS.632 Conversational Computer Systems
Fall 2008

9

Higher Levels of Linguistic Knowledge

Chapter 1 presented an analysis of human voice communication as a hierarchical stack of linguistic layers. In this schema (depicted in Figure 1.1), each layer of the stack is progressively more removed from the acoustical events associated with speech. The higher layers pertain to the meaning and intention of an utterance rather than to the actual sounds used to convey individual words. As proposed in Chapter 1, analyzing conversation into layers is useful for computational purposes because each layer is likely to necessitate its own representation.

The discussion of speech recognition and synthesis technologies emphasized the lower linguistic layers of communication, i.e., speaking or recognizing words. Syntactic and semantic constraints were introduced as aids to connected speech recognition and influences on intonation and stress. The previous chapter focused on interactive techniques to recover from speech recognition errors with the goal of properly decoding the words in an input utterance. From the user's perspective, speech recognition is not about identifying words; rather, it is a means whereby a computer system understands and acts upon the user's desires. Performance and utility of speech systems are ultimately judged against the highest layers of speech understanding. To this end, this chapter presents representations of syntactic, semantic, pragmatic, and discourse structure that serve the reader as an introduction to topics in natural language processing. The chapter includes several case studies to illustrate the wide range of conversational interactions made possible by models of discourse.

SYNTAX

Syntax is structure in language imposed by the limited number of ways in which words may be combined in various linguistic roles. Words have different characteristics in a language. Nouns refer to persons, places, concepts, and sensations, while verbs describe actions performed by or on nouns. Adjectives modify nouns, qualifying or quantifying them, and adverbs similarly modify verbs by further specifying the action.

Syntactic structure groups words into phrases and phrases into sentences and describes the relationships between components at each level of structure. At a low level, syntax constrains the gender and number of adjectives and pronouns to agree with the noun to which they refer. The pronoun in "Jane ate his candy bar" is ill-formed if the candy bar belongs to Jane. In many languages the gender of nouns is more significant than in English, and adjectives exhibit different forms for each. For example, in Spanish "el rey blanco" (the white king) and "la reina blanca" (the white queen) illustrate the change in form of the adjective ("blanco") as well as the determiner ("el"/"la") depending on whether they modify a masculine or a feminine noun.

Conversational systems may utilize knowledge of syntax while trying to understand a user's request using speech recognition; because syntax limits the possible combinations of words, perplexity can be minimized. Syntax must also be incorporated into language generation or else the system's replies may be incoherent or less intelligible due to grammatical errors. For a conversational system to make computational use of syntax, two components are required: a grammar and a parser.[1] A **grammar** is a formal specification of language structure; it defines how words can be combined into syntactic units and how these units can be used to build sentences. A **parser** analyzes a sentence to determine its underlying structure as defined by the grammar. The following sections describe several grammatical representations and then present some basic parsing techniques.

Syntactic Structure and Grammars

Syntax provides structure in a language by constraining the manner in which words can be combined according to a regular set of rules. In this section we consider representations of syntactic structure and formalisms for expressing the rules that comprise a grammar. We can represent the layers of syntactic structure as a **tree,** starting by representing the complete sentence as the root node; this is similar to the technique of diagramming sentences, which many of us learned in elementary school. For example, the most simple sentences contain only a noun

[1]Language structure can also be represented probabilistically, e.g., using Hidden Markov Models. Although such an approach is adequate for encoding constraints to aid speech recognition, it is less well suited for representing the grouping of words into syntactic components.

(the subject) and a verb and can be represented as in Figure 9.1, e.g., "Cats purr." Trees capture the order and grouping of words and also indicate the unique label (noun, verb, etc.) associated with each word.

To generalize to more complex sentences we must introduce the concepts of **noun phrase** and **verb phrase.** A noun phrase is a group of one or more words that act as a noun in a sentence. A verb phrase similarly allows multiple words to act as the verb. Because they can contain multiple component words, these phrases introduce another level of structure to our tree representation. A sentence such as "Orange cats purr loudly." is represented as in Figure 9.2. The branching of this tree structure indicates that the adjective "orange" modifies the noun "cats," while the adverb "loudly" is likewise related to the verb "purr."

While trees such as these examples convey the structure of particular sentences or classes of sentences, they do not tell us how to derive the structure for a

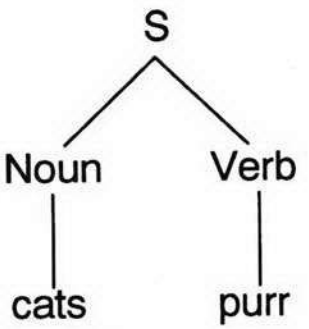

Figure 9.1. The figure shows the root node representing the entire sentence above two leaf nodes: one for the noun and one for the verb.

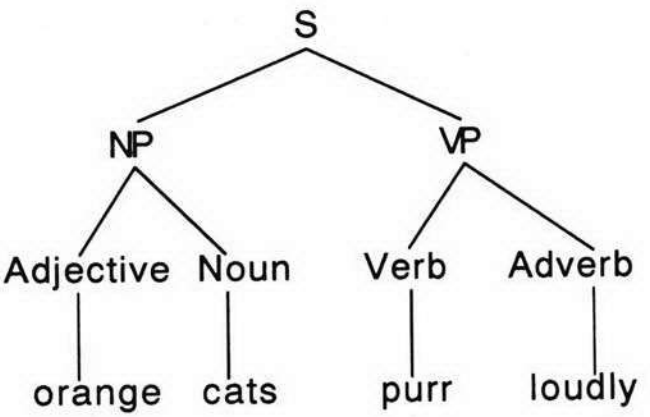

Figure 9.2. The tree representation expanded to account for a noun phrase and a verb phrase.

given sentence. Such analysis requires a set of rules that can describe all sentences of a language. A formal set of rules is a **grammar;** a complete grammar specifies the rules for all syntactic structures in a language. Grammars can be specified in several ways.

A **context-free grammar** describes syntactic structures as a series of **rewrite rules.** Each rule defines how a single symbol (on the left side of the rule) can be decomposed into one or more components (the right side of the rule). The grammar is "context free" because the symbol on the left appears in isolation, i.e., removed from the context of the other symbols. Although this constraint limits the range of sentences that can be expressed by a context-free grammar, it is nonetheless quite powerful and frequently used.

Using a context-free grammar, a sentence is broken down step by step into its constituent components through successive invocation of the rewrite rules. The symbols on the left are **nonterminal,** while those on the right may be either nonterminal or **terminal;** terminal symbols are individual words. Successive application of rules reduces all nonterminal symbols to terminals.

A grammar specifying the two sentences previously shown in tree form might resemble that shown in Figure 9.3. The first of these rules (S ← NP VP) specifies that a sentence "S" consists of a noun phrase and a verb phrase. The second rule defines a noun phrase as a single noun ("cats") or an adjective and a noun ("orange cats"); the vertical bar symbol means "or." Although this grammar describes both "Orange cats purr loudly" and "Cats purr," it excludes many other sentence structures such as "Purr!" (no explicit NP), and "Large orange cats purr loudly" (multiple adjectives). Note that each rule breaks the sentence into the same component parts as appear on the branches of the tree representation.

We can include the first of these exceptions by modifying the first rule to read.

S ← NP VP | VP

The new version of the rule specifies that a sentence can consist of a noun phrase followed by a verb phrase or just a verb phrase in isolation. A minor modification of the second rule allows an arbitrary number of adjectives to precede the noun.

NP ← N | Adj NP

```
S ← NP VP
NP ← N |    Adj N
VP ← V |    Adverb V
```

Figure 9.3. A simple context-free grammar. The symbols S, NP, and VP are nonterminal, whereas the rest are terminal and correspond to individual words.

Context-free grammars can describe many sentences in a natural language such as English; they also specify most programming languages in their entirety. Note that the abbreviated discussion of context-free grammars presented here has avoided some of the issues of gender, number, and case agreement mentioned earlier. For example, the modified verb phrase rule just described allows "Purr!" which is a valid imperative sentence, as well as "Purrs," which is incomplete.

A second powerful representation for grammars is the **transition network** or a slightly more general variation, the **recursive transition network, RTN.** A simple transition network consists of nodes and arcs as shown in Figure 9.4. Each arc is labelled with a word category, and each node is a state in the syntactic analysis of a sentence. Starting at the beginning of the sentence, each word is compared with the label on the arc from the initial state, **S.** If the word and label match, analysis transitions to the next node and word and label comparison continue as before. The last arc is labelled **pop** and always succeeds, indicating completion of the network.

This linear network does not convey the syntactic relationships between the adjective-noun and adverb-verb pairs as well as do the equivalent tree or context-free grammar representations. The network can be enhanced by allowing arcs to be labelled with the names of other networks as well as individual words; with this ability, the network has become recursive. For example, Figure 9.5 illustrates a transition network describing a noun phrase as an arbitrary number of adjectives followed by a single noun. Figure 9.6 shows how the noun phrase net-

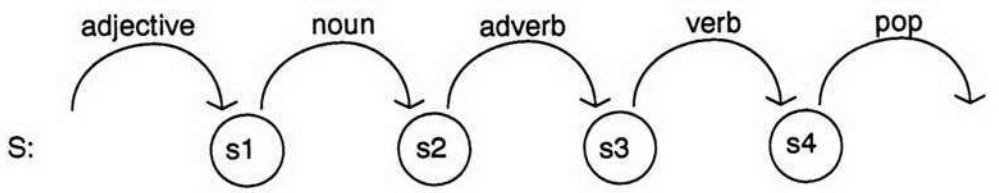

Figure 9.4. A simple transition network for sentences consisting of a single adjective, a noun, a single adverb, and a verb. The states, s1 through s4, are connected by arcs labelled with the classes of words required at each transition.

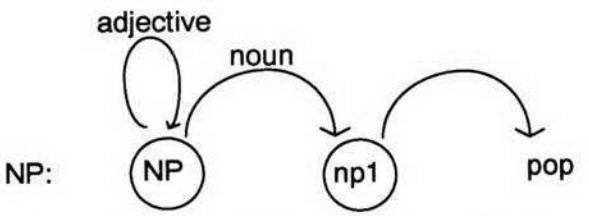

Figure 9.5. A transition network for a noun phrase allowing an arbitrary number of adjectives.

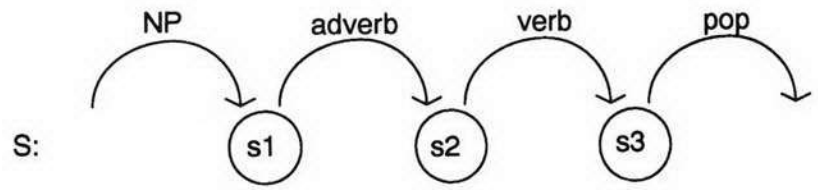

Figure 9.6. The transition network shown in Figure 9.3 modified to use the noun phrase network in Figure 9.4. This network is now recursive.

work can be incorporated as a subnetwork by modifying the simple network shown in Figure 9.4.

Let us return to the issue of agreement of gender, number, case, tense, etc. mentioned as an open issue in the discussion of context-free grammars. RTNs can manage some of the agreement issues but in a rather tedious manner. For example, we can define subnetworks for a singular noun phrase as well as for a plural noun phrase, restricting the number of the constituents of each, and then pair these with verb phrases of the appropriate number as shown in Figure 9.7. But a more succinct method is to return to Figure 9.6 and apply a condition to the arc labelled "verb" from state **s2** to state **s3.** This condition stipulates that the number (singular or plural) of the verb satisfying this arc must equal the number of NP satisfying the arc from **S** to **s1.** Although we cannot use conditional transitions within the constraints of the RTN formalism, we can extend the RTN to an **augmented transition network,** or **ATN.** Two features of ATNs are the conditional transition just mentioned and the ability to save information during transitions.

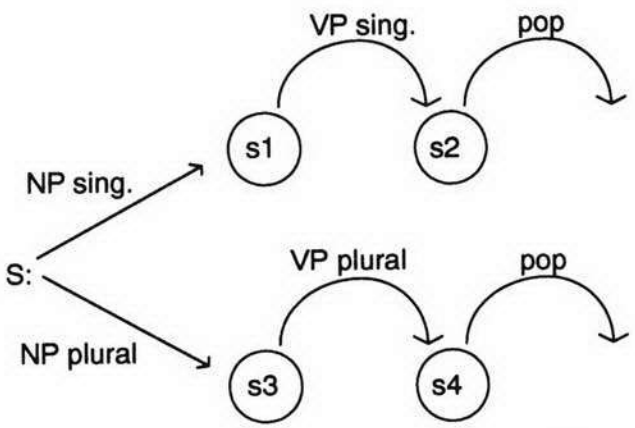

Figure 9.7. One way of representing number agreement between subject and verb using RTNs.

Parsers

We have just described the various means of representing or specifying the syntactic structure of sentences in a language. It is the task of a **parser** to determine the actual structure of a specific sentence. In its most simple form, a parser takes a sequence of words as input and given the constraints of grammar determines whether the sequence is a valid sentence. By itself this tells us little: only whether the sentence is well-formed. A more complete parser also produces a representation of valid input, which can be employed in higher layers of sentence meaning analysis.

A parser can operate in a **top-down** or **bottom-up** manner to relate the set grammatical rules to the actual structure of the given input. A top-down parser that uses rewrite rules starts with the most general rule describing sentences, i.e., that rule for which the whole sentence "S" is the symbol on the left side of the rule. Applying rules breaks the sentence into smaller components, that eventually become terminal symbols, i.e., specific words in the rewrite rules. For example, using a context-free grammar, a top-down parse of the sentence "Orange cats purr loudly" would first identify the sentence as containing a noun phrase and a verb phrase. Then the noun phrase rule, specifying a NP as some number of adjectives followed by a noun, would be selected, and as a result the parser would identify "orange" as an adjective and "cats" as the noun. At this point, the parser would analyze the remainder of the sentence by identifying the verb phrase, i.e., the right side of the tree from Figure 9.2.

The RTN version of this parse is nearly identical because the top-down parser operates from left to right within a level of syntactic structure. The first constituent of the highest RTN from Figure 9.6 is the NP; this causes the parser to invoke the NP RTN (see Figure 9.5), which allows a NP to begin with either an adjective or a noun. After traversing the adjective arc (for "orange") the parser is still in the initial state. The next word, "cats," is a noun leading to state **np1,** which is followed by the pop arc that returns to the RTN describing the main sentence. At this point the VP RTN can be applied.

Top-down parsing starts with the constituent units such as NP and VP and attempts to match them against incrementally smaller units and eventually words. By contrast, bottom-up parsing begins with the individual words of a sentence and tries to match each word against the grammar's smallest constituent units. These units are, in turn, matched against higher units until the structure constitutes an entire sentence. The bottom-up parser consults a lexicon to identify the syntactic sense of each word. To continue with the example sentence, the first word "orange", is taken to be an adjective from the lexicon. The second word is a noun. These can be matched to a NP. Similarly, "purr loudly" can be matched to a VP in the form of a verb followed by an adverb. A NP followed by a VP makes a complete sentence and the parse is finished.

In the course of parsing, **backtracking** may be required due to ambiguous sentence structure. "Orange" could be taken as a noun (a fruit) instead of an adjective in which case the bottom-up parse would hypothesize that the NP consisted of the single word "orange" (the simplest NP consists of a single noun). But then

"cats" would have to be taken as a NP too since "cat" is also a noun. At this point the parse would fail since there is no sentence structure containing two adjacent NPs in this grammar. A bottom-up parser keeps track of each decision in a data structure called a **chart** and uses this to backtrack to a previous state. For the example we have been pursuing, the parse would return to the decision to classify "orange" as a noun, classify it instead as an adjective, and proceed smoothly to complete the sentence.

Both top-down and bottom-up parsers have advantages and disadvantages. Because a top-down parser starts from a syntactic constituent and maps it to individual words, it never considers word categories in grammatically incorrect sentence positions. On the other hand, the top-down parser may process many rules before it ever considers an actual word, and it may repeatedly parse the same segment by backtracking further than necessary. A bottom-up parser avoids repetition; once words are successfully matched to a higher syntactic category they are never reconsidered. But as previously discussed, the bottom-up parser may need to consider multiple senses of each word in the process of matching it to constituent structures. These considerations have led to hybrid parsers, which are top-down in part and bottom-up in part as well.

The parsing process just described either succeeds or fails in its attempt to analyze a sentence. Success indicates that the series of words presented to the parser is a valid English sentence, and failure implies the opposite. But this conclusion alone is of little use to a language understanding system; a practical parser must also produce a computational representation of the syntactic structure and its composition from individual words. For example, if the user says "Delete all files," a language understanding system needs to know which command the user uttered as well as its object; this specific information is more valuable than knowing the NP and VP structure of the sentence. This information is conveyed by recording the actual word that triggers each rule in a context-free grammar or each transition in an ATN. The parse then produces not only the success or failure indication, but more importantly a representation of the sentence structure and how the words of the sentence satisfy this structure.

The discussion so far has assumed that either a sentence was valid in which case the representation is desired, or the input did not form a valid sentence. But for voice interactions additional flexibility is desirable. If the user speaks a 10 word sentence and one of the words is not recognized, it is appropriate to detect this and employ some of the error correction techniques described in Chapter 8. Even if a recognition error does not occur, we often speak utterances that are only partial sentences. A **robust** parser should be able to cope with such fragmentary or ill-formed input, but most natural language processing work to date avoids this issue.

SEMANTICS

Semantics is the realm of **meaning,** in particular the meaning of a sentence considered in isolation from other sentences. A sentence has meaning in terms of

how it relates to the world and objects in it; the sentence stands for some concept or real entity. Meaning is what differentiates nonsense utterances of arbitrary words from rational sentences, and much of the study of semantics is devoted to representations of knowledge about the world.

Semantics begins with meaning at the lexical level; we cannot understand a sentence unless we know the meaning of each word in the sentence. Further, most words have multiple meanings or **senses,** and we must choose among these senses to understand the sentence as a whole. For example, "green" when used as an adjective usually denotes color ("green leaves") but also has another sense in which it means novice or naive ("a green first baseman"). These two senses may be differentiated by considering the noun that "green" modifies. Experience is a concept applied to sentient beings not plants so the sense of "novice" is appropriate only when green modifies a noun that corresponds to a person. One word in a sentence may constrain the interpretation of other words in the sentence. Note the interplay between syntax and semantics: syntax reveals that "green" is used as an adjective and indicates which noun "green" modifies, from which we can then determine the sense in which "green" is being used.

The intended senses of words in a phrase or sentence are mutually constrained; this is called **selectional restriction.** Our knowledge of the attributes of objects in the world helps us identify the sense in which a word is being used to describe an object as we did in the example in the previous paragraph. In addition to objects, i.e., noun phrases, we also apply knowledge about how objects interact, i.e., verb phrases, while selecting word sense.

Case grammar attempts to enumerate the cases, or roles, a noun phrase can take with respect to a verb. One case is AGENT; the agent is the instigator of an action and is usually the subject of a sentence. The object acted upon, often identified as the object of the sentence, is a case called THEME. In the sentence "John smashed up the car," John is the agent and the car is the theme. Cases such as AT, TO, and FROM indicate location. An animate object for which an action is performed is the BENEFICIARY, while an animate object in a mental state is the EXPERIENCER. The number of cases is extensive because noun phrases can be used in many different relationships to verbs. Just as real-world knowledge helps establish the meaning of noun phrases, case grammar provides a representation of meaning derived from the roles of noun phrases in an action or state of being.

From this very brief discussion it should be apparent that we obtain cues for the cases of noun phrases from their position in a sentence. For example, subjects, or AGENTS, precede objects, or THEMES, in an active voice sentence, but this order is reversed for passive voice. Prepositions are particularly strong cues for the location cases (AT, TO, and FROM). Selection of cases is controlled by verbs in other ways as well. The verb "move," for example, has a strong requirement for a TO location case to complete its usual meaning as in "we moved to Wyoming." For other sentences a different case is appropriate; an EXPERIENCER completes the sentence "Her singing moved the audience." The concept of case conveys some aspects of the meaning of noun phrases in a sentence, but in the case of a noun phrase it also depends on the meaning of the verb with which it is associated. The

presence or absence of a particular case is, in turn, a cue towards selecting the sense in which the verb is being used.

Earlier in this section we discussed how the noun selects between various senses for the adjective "green." The "green first baseman" refers to the lack of experience by a person. How can we represent the knowledge that a "first baseman" is a kind of person to provide the selectional restriction on "green"?

Semantic networks are a powerful representation for such relationships. A semantic network is a directed graph where nodes represent an individual word or a class of words, and the links represent relationships between child nodes and their parents. A very simple form of semantic network is the **type hierarchy** in which every arc represents the "is-a-kind-of" relationship. Figure 9.8 shows a partial type hierarchy indicating that a first baseman is a baseball player, baseball players are athletes, athletes are people, people are mammals, and mammals are animate objects. The type hierarchy also indicates that reptiles are animate objects and that rattlesnakes are a particular kind of snake.

The hierarchical nature of this representation indicates that all members of a lower node possess all the characteristics of their parent node, or superclass; this concept is called **inheritance.** Inheritance is useful to express generalizations concisely. With any taxonomy of this sort it is important to select the levels of classification with care; ideally, each step down the hierarchy represents a consistent increase in specificity across the breadth of the network.

A significant limitation of this form of type hierarchy is that any node can be a child of exactly one parent node, representing a subclassing of the parent along a

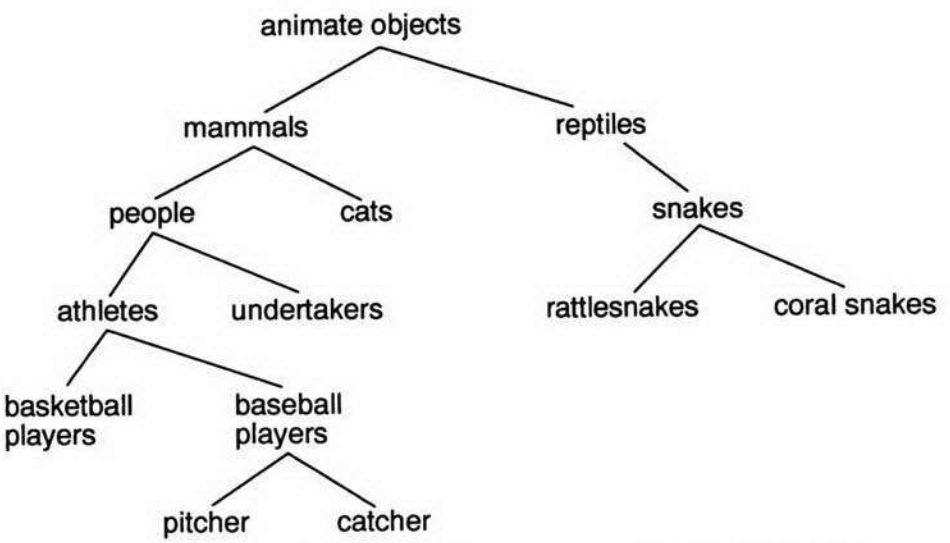

Figure 9.8. A type hierarchy. Each arc represents the is-a-kind-of relationship.

particular set of criteria. However, many conflicting subclassifications may be possible. For example, Figure 9.8 subclassed people on the basis of their profession. Figure 9.9 shows other ways of subclassing people based on sex or age. All these subclassification schemes are valid and express a set of differences among people. But these simple two-dimensional type hierarchies break down when attempting to depict multidimensional classification parameters, e.g., differentiating male basketball players from female first basemen.

Semantic networks can be used to convey concepts more general than type hierarchies and are used to represent more complex knowledge relationships. Figure 9.10 shows a sentence displayed on a semantic network in which both the type hierarchy relationships (ovals) as well as case grammar roles (squares) are drawn. Networks can be partitioned into subnetworks to keep close, specific relationships distinct from more general background knowledge or situational context, much of which may be irrelevant to the meaning of a particular sentence. Identifying the meaning of a sentence with semantic networks involves matching the sentence to a particular path or grouping within the network, so that the specific words in the sentence are mapped to their own specific meanings and the case relationship between the words becomes manifest.

PRAGMATICS

Pragmatics refers to the meaning or purpose of an utterance in a context broader than the semantic meaning just discussed. Pragmatics is about how language is used both in terms of the forms in which speech is invoked as well as in terms of its effects on the conversants. This chapter includes in the scope of pragmatics the role of the utterance in the world and among conversants (To which specific objects in the world does the utterance refer? What does the talker seek to accomplish in speaking it?); to this end it covers knowledge representation as well.

This section introduces three topics in pragmatics, beginning with knowledge representation and plan recognition, which are particularly computational views. The second topic is speech act theory, which views utterances as actions in and of themselves. Finally, a set of guidelines that regulates how much or how little we say and when we say it is introduced; this is formulated from the perspective that an utterance often contains much more meaning than is obvious from the words themselves (the semantic meaning).

Figure 9.9. Alternate type hierarchies subclassing people.

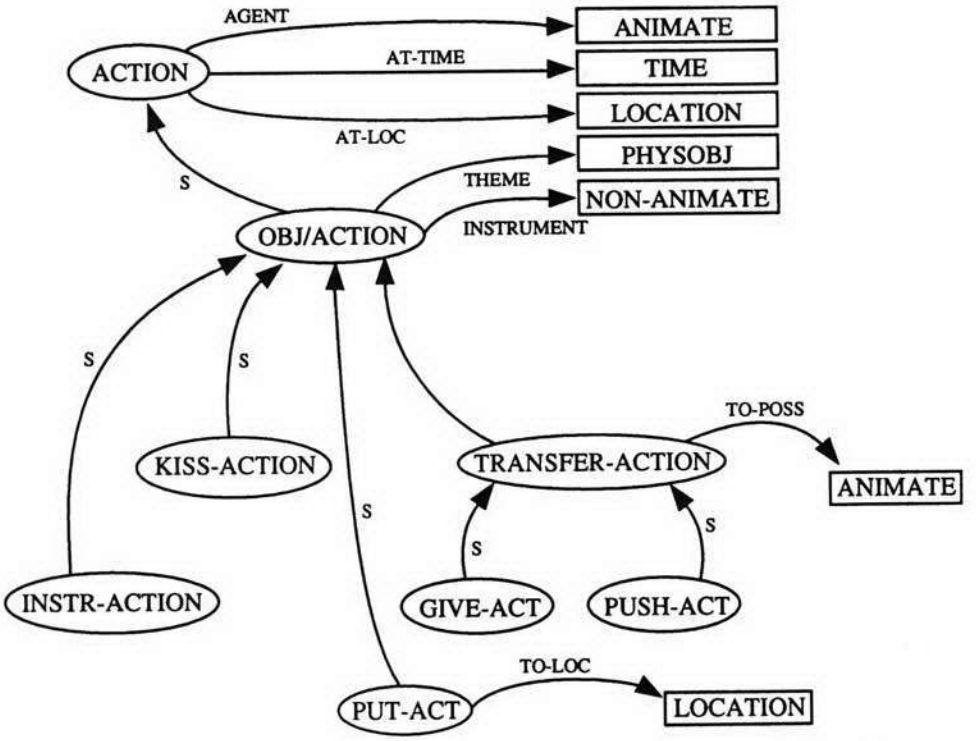

Figure 9.10. A semantic network including both a type hierarchy (ovals) and case grammar (squares) categories. After [Allen], p. 209.

Knowledge Representation

To understand what a person intends by an utterance we need to employ knowledge about the world and the objects in it being referred to by the sentence. For example, to interpret "It's a bit too breezy in here" as a request to close the window of a car, we need to know that a breeze is motion of air, that solid objects block the flow of air, and that windows are solid objects that can be closed by some mechanism such as a crank or motor. A **knowledge representation** provides a means to store and computationally access information about the world, which may be general facts (although windows are clear, they can close an opening) or details of a specific current situation (the car window is open, and the crank will close it).

Knowledge representation systems contain two components, a knowledge base and the inference engine. A **knowledge base** is a database containing facts about the world or information about a situation, and the **inference engine** operates on the knowledge base according to a set of rules. Inference may be primarily **declarative** or **procedural,** depending on whether it stresses the knowledge database or the inference operations to draw conclu-

sions. This section touches upon several aspects of applying world knowledge to language understanding; this is meant to be just an introduction to a complex topic.

Frames are a very general representation used by many procedural language understanding systems. A frame is a loose collection of facts about a situation, some of which may be relevant to understanding a specific utterance [Minsky 1975]. Inferences are made by comparing relationships described by the frame with the semantic representation of the utterance. The important objects contained by a frame are called **roles.** For example, a frame for an airplane would likely have roles such as wings, tail, and engines; for a particular aircraft, the role of the engines might be associated with details such as the number of engines, their configuration, and manufacturer. Frames are often implemented in concert with a type hierarchy and likewise may incorporate inheritance.

Planning systems focus on the roles of actions and their grouping as a sequence to effect a change of state, or **goal.** Actions are described in terms of preconditions, the set of situations that must be true for the action to occur, and effects, or changes in the world situation as a result of execution of the action. For example, preconditions to sending a letter are writing materials and an addressee, while the effect is for the addressee to receive a written message. **Linear** plan reasoning analyzes goals in terms of sequential subgoals; the satisfaction of each subgoal yields the preconditions for the next. **Nonlinear** planning merges all subgoals into a global set of actions worrying about order of the actions only when necessary. Planning systems seek to understand utterances by recognizing the talker's plan, i.e., a sequence of actions to achieve a goal state. As a plan is likely to evolve over multiple utterances, planning systems are equally salient to discourse analysis as to semantics.

Another representation for commonly performed actions is **scripts,** which specify all the actions ordinarily required as steps toward completion of a "typical" activity [Schank and Abelson 1977]. For example, commercial air travel usually involves consulting schedules, choosing flights, making reservations, purchasing tickets, arriving at the airport, checking baggage, flying, retrieving baggage, and leaving the airport. These actions can be represented as part of a general script for traveling by public transportation (traveling by private car, bicycle, or foot must be described by a different script). Note that some steps, such as checking baggage, may be optional. Plan reasoning can be applied to scripts as well by considering each step in a script to be a precondition for the next step.

Scripts can be used to recognize the real-world activity to which an utterance refers. "I bought my ticket" could refer to any of an extensive class of activities for which admission is required (flights, plays, movies, baseball games). "I bought the ticket for my flight" can invoke the travel script, including the related concepts of departure time and destination city. Once the travel script has been recognized, other components of the script provide a framework for reasoning about the talker's activity. Once we know the ticket is for air travel, for example, we might ask "Where are you going?" or "When will you arrive?"

Speech Acts

Speech act theory considers utterances as actions in and of themselves. Much of modern speech act theory began with the work of Austin [Austin 1962]. Austin started by considering **performative** utterances, for which the act of speaking itself constitutes the action, such as:

I now pronounce you man and wife.
I hereby declare war on Germany.
I apologize for your inconvenience.

As the first two utterances indicate, these performative utterances are often of a legal or religious nature in which the talker has been granted a higher authority to speak for a society rather than merely speaking as an individual. Austin extended the notion of performative utterances with the concept that many utterances manifest a **force,** or ability to change the state of the world by their very existence, due to their impacts on the beliefs or intentions of the conversants. He identified three senses in which action is performed through an utterance.

1. The **locutionary act** is the actual utterance of a meaningful sentence.
2. The **illocutionary act** is the sense in which the utterance itself contains force to effect a change in state among conversants, e.g., the utterance may act as a promise, offer, decree, statement of commitment, or admission of guilt.
3. The **perlocutionary act** is the manner in which the utterance changes the audience by changing their beliefs or causing them to perform some action.

There have been some attempts to associate force with sentence form; some obvious examples include commanding, using the imperative form, and questioning via the interrogative form. But the mapping from force to form becomes problematic for **indirect speech acts** in which the force depends on speaking conventions. "Can you pass the salt?" is usually meant as a request not a question about the other's ability to pick up the salt shaker. In a similar vein, "It's pretty noisy in here" might be uttered as a request to turn down the radio volume.

[Searle 1976] offered a more detailed taxonomy of five classes of illocutionary action conveyed by an utterance.

1. **Representatives** are sentences by which the speaker asserts or commits to the truth of the utterance.
2. **Directives** are requests to the listener to do something. Questions are directives in that they attempt to evoke a response.
3. **Commissives,** or promises, commit the talker to future actions. Threats are also examples of commissives.
4. **Expressives** create a psychological state rather than causing a physical action. They include welcoming, apologizing, thanking, etc.
5. **Declarations** are the performative statements mentioned earlier such as declaring war or firing an employee.

The notion of speech acts is a very insightful explanation of the purpose of utterances, and much of our speech can be effectively analyzed according to Searle's taxonomy. But speech act concepts have practical limits. Speech is very social in nature, and many utterances are oriented less towards causing an action than towards establishing or maintaining a social relationship. The Coordinator, a software product that utilizes speech act theory for tracking work team assignments by requiring users to assign speech act classifications to electronic mail messages, has met with mixed acceptance in actual use [Winograd 1988, Medina-Mora *et al.* 1992]. Although utterances may be successfully classified by speech act theorists, in the actual interplay of office communication correspondents may not wish to be so direct with each other.

Conversational Implicature and Speech Acts

Conversational implicature is the principle that an utterance often contains much more meaning than the words themselves indicate directly. Imagine that two people meet on the street and the following conversation ensues.

A: Did you do it?
B: Not again.

This conversation is largely meaningless to us; the conversants make extensive use of shared knowledge and history, and we cannot know to what action they refer. We can, however, make certain inferences about the dialogue: A and B have discussed "it" before, "it" is dominant enough in their relationship not to be ambiguous, and A assumes "it" is unlikely to be ambiguous. Another inference from the conversation is that B has done this "it" in the past and likewise understands that A knows about this prior occurrence (or most recent of several prior occurrences of "it").

To make these inferences with any degree of confidence, we resort to some implicit assumptions about the use of language. We assume that the utterances have a purpose, e.g., A and B are not total strangers but in fact have a common history. We make assumptions about shared belief, e.g., that A does expect B to know the referent for "it" and is not simply teasing B to see whether he or she can guess what is being talked about. In short, we assume an order or regularity to these utterances.

Grice's theory of conversational implicature [Grice 1975] is based on the concept that there is a set of guiding assumptions molding how a conversation is organized. Such guidelines allow us to make the inferences described above, and any deviation from the guidelines must itself have implications beyond the actual words used. Grice's **maxims of conversation** can be summarized as follows.

- **The cooperative principle:** Speak to support the accepted purpose or direction of the conversation as it is at the moment of the utterance. This is the underlying theme of all the maxims.

- **The maxim of quality:** Speak the truth; do not speak what you know to be false nor that for which you do not have adequate evidence of truth.
- **The maxim of quantity:** Be direct. Say as much as is required at the current point of the interchange, but do not say more than is required.
- **The maxim of relevance:** Make your utterances relevant and to the point.
- **The maxim of manner:** Be brief and orderly; avoid ambiguity and obscurity.

The cooperative principal defines conversation as an organized, purposeful, and efficient series of spoken exchanges, and the remaining maxims are specific techniques to support this principle. In this framework, conversants work together in a conversation, their utterances are mutually relevant, and they do not speak nonsense or seek to confuse each other.

In reality, language often is not nearly as orderly as these underlying principles seem to imply. However, in many cases when Grice's principles appear to be violated, they unify otherwise disjoint or seemingly purposeless utterances. Consider the following exchanges.

A: Did you feed the cat?
B: There's a pile of feathers at the doorstep.

B appears to violate the maxim of relevance; A wishes to know whether B has fed the cat, yet B talks about a pile of feathers. By assuming that B's reply must be relevant, we can infer that B suspects that the cat has caught a bird, and perhaps that this should substitute for the cat's dinner or that the cat should not be fed as punishment.

DISCOURSE

Discourse refers to multiple utterances over time often by different talkers. Discourse is a broad term covering several distinct areas of language understanding. The utterances in a discourse are not disjointed but rather related or connected. Discourse deals with understanding the purpose of these multiple utterances, including issues such as plans that extend across several utterances, references in one utterance to objects or concepts specified in a prior utterance, and the use of multiple utterances to comprise a single speech act. Discourse issues in conversations include the use of language to regulate the conversation's flow, the temporal distribution of contributions among its participants, and the coordination of the effective exchange of information among the participants such that all arrive at the same understanding of what was said.

Discourse issues have formed a theme of conversational interaction and feedback throughout this book. This section emphasizes how conversation is broken

into turns. The flow of turn-taking provides a collaborative environment for each talker's contributions to the conversation. Conversants maintain a common focus across turns; without this, pronouns could not refer to ideas and objects mentioned in an earlier sentence. Feedback techniques to ensure mutual understanding are discussed in the subsequent section.

Regulation of Conversation

We all know from personal experience that in a conversation the various talkers take **turns** speaking. After each turn, remarkably little time transpires before the next turn begins. Occasionally turns overlap, as one participant begins before the previous has completely finished, but it is remarkable that conversations can be as dynamic and fast paced as they are without more "stepping on each other's toes." Equally remarkable are the conversational processes for selecting a mutually agreeable topic, moving on to new topics, and returning to a previous topic.

Conversation is rich in social conventions that invite discourse and most utterances occur in a discourse context [Goffman 1981]. Turns in conversations are regulated and ordered to allow a chain of utterances to refer to a single topic; often subsequent utterances can be understood only in the context of the earlier portions of the discourse. The most simple example of this dependency is the **adjacency pair** [Sacks *et al.* 1974], in which something is presented or proposed in the first utterance and responded to, accepted, or rejected in the rejoinder. For example:

A: I brought in the mail.
B: Thank you.

A: How much does this cost?
B: Two dollars.

Note that the second utterance in the pair, which brings the pair to some form of closure, has little clarity of its own outside of the adjacency pair.

Where applicable, adjacency pairing simplifies the question of how the listener knows when the talker's turn is over as the listener must clearly wait for the proposition to be presented in the first member of the pair. Although it may be suggested that all conversations can be reduced to sets of adjacency pairs possibly with inserted sequences of other adjacency pairs between the first and second member of a pair, most conversations are more complex and resist this analysis.

In the absence of simple pairs, it is harder to specify when one turn has ended and another talker may begin a new turn. What constitutes a turn? How does the talker signal this to the listener? Turns are often composed of one or more syntactically or semantically meaningful units. These units may correspond to sentences, but they are equally likely to be smaller phrase-like units; fluent conversation often contains incomplete sentences. One appropriate unit is the **breath group,** or the string of words between catching one's breath, which usually expresses one or more coherent thoughts.

For detecting turn boundaries, the problem lies with the "one or more" of the preceding paragraph. If one talker spoke and the second always picked up when the first stopped for breath, turn taking would be more predictable. But the talker may continue to "hold the floor" for multiple utterances or the listener may interrupt before the talker has even finished or otherwise signal so that the talker modifies the utterance even as it is being produced.

The time between turns is too short (often shorter than pauses within a turn) to believe that the listener simply waits to hear if the talker has more to say. [Duncan 1974, Duncan 1972] analyzed a number of conversations and suggested the following indicators in addition to the completion of a syntactic unit of subject and predicate by which the talker can signal the end of a turn.

- **Intonation:** A level or falling pitch at the end of a sentence indicates termination of the thought being expressed.
- **Syllable lengthening:** The final syllable, or more correctly the final stressed syllable at the end of a turn, is longer than it would be otherwise. Duncan refers to this as "drawl."
- **Gesture:** Termination of a hand gesture while speaking acts as a cue that the accompanying turn is drawing to a close.
- **Key phrases:** Certain phrases such as "you know . . ." at the end of a syntactic unit are often spoken at turn termination.
- **Visual attention:** Talkers often avert their gaze from the listener during an utterance so looking back to the listener could cue the end of a turn.

The completion of each syntactic unit or phrase is a possible end of the turn. If the talker indicates termination by cues such as those just listed, this invites the listener to take a turn. If the current talker desires to continue the turn, i.e., to present a subsequent phrase, the end-of-turn cues can be avoided or more strongly, the opposite behavior can be invoked such as using a rising intonation or beginning a hand gesture.

Conversation does not always break into turns cleanly. Sometimes, either deliberately or accidently, the listener may **interrupt,** i.e., begin speaking before the other has finished. Interruption is usually dealt with effectively in conversation: Often one party will back off and quickly cease speaking, in which case whichever party is then the talker tends to repeat or summarize what was said during the period of overlap to insure that it was heard correctly. During overlap, the conversants have not yet resolved who should have a turn; one party may attempt to assert control by emphasis such as speaking more loudly or with increased pitch range or lengthened syllables.

Speech may be used by the listener in a manner that initially seems to be a short turn or an interruption but does not really take a turn away from the talker. **Back channels** refer to a number of behaviors whereby the listeners give feedback to the talker [Yngve 1970]. They include paraverbal utterances ("Hmmm," "Uh-huh"), completing the other's sentence or offering a paraphrase of it, short interjections ("Of course," "You don't say?"), head nods, and various facial expressions.

Back channels are a cooperative mechanism; listener feedback indicates what is known or accepted so that the talker can continue the exposition with confidence. Back channels make for more productive conversation. For example, in an experiment by [Kraut *et al.* 1982, Kraut and Lewis 1984], subjects described scenes from a film to a listener who attempted to identify the film. If the listener who was out of sight could not speak back, it took longer for the talker to adequately describe the scene. Even an eavesdropper who could never be heard benefited from the listener's back channel utterances but not as much as the listener did. This suggests that some aspects of back channel cooperation produce generally "better" utterances from the talker, while other aspects of performance improvement are specific to the participation of the back channel provider.

Discourse Focus

Back channels are just one aspect of collaborative behavior in conversation. In the course of speaking conversants change or agree upon the topic of conversation, refer back to previous topics, and reaffirm their basis of mutual belief upon which they can build successful references to world knowledge, either generic or specific and situational. The discussion of turn taking emphasized pairs or short sequences of talk. We now turn our attention to longer conversations with perhaps quite a few turns.

At any moment in coherent discourse the conversants usually agree on what is being discussed. From time to time, the topic of conversation changes. The group of sequential utterances that refers to the same topic is a **discourse segment.** Transitions between discourse segments are often indicated by **cue phrases** such as "By the way . . . ," "Yes, but . . . ," and "Well. . . ." Throughout a discourse segment, all utterances refer to the same topic or noun phrase; this is the **focus** or **center** of the discourse segment.

Identification of the focus of a discourse segment is required to resolve **reference,** which arises from several sources. **Deixis** is the reference of certain pronouns, such as "this" and "those" that point at something either physically or conceptually. **Anaphora** is the reference implied by pronouns such as "he" or "their." The entity referred to by deixis or anaphora is the **referent.** The referent corresponds to the focus of the discourse segment; changing the referent introduces a new discourse segment.

A discourse segment can be interrupted by the introduction of a new discourse segment, and the original discourse segment can be returned to. For example, consider the discourse fragment.

A: California gets so green in the winter, I love it!
B: Seattle gets a lot of rain in the winter too, but not much sun.
A: It's a nice city, but you should check out Hawaii if you want wonderful winter weather.
B: Last winter we went hiking there.
A: Sometimes it gets a spell of rain in February.
B: But it's not as bad as back there! It's so dreary all winter.

In the first sentence, speaker A references "California." Speaker B then introduces a new reference "Seattle." Speaker A refers back to Seattle at the beginning of the next utterance but then introduces a third focus "Hawaii" using the cue phrase "but." The next two utterances then refer to Hawaii as well. In the last utterance, speaker B jumps back to the focus of "Seattle" without needing to further specify the pronoun. How is this accomplished without further negotiation?

[Grosz and Sidner 1986] proposed a discourse model differentiating the **attentional structure** that specifies the target of reference from the **intentional structure** that is roughly the pragmatic purpose of the discourse segment. They suggested a **stack** model for the attentional structure. A stack is a data representation in which items are put on ("pushed") and removed ("popped") from the top so that the most recently pushed item is always the one that gets popped. Figure 9.11 shows the stack progressing during the course of the example discourse. In the last snapshot, the top focus "Hawaii" has been popped, leaving "Seattle" exposed as the prime candidate for reference.

This model suggests that once popped, an object cannot be referred to again by a pronoun without being specifically introduced as a new focus so that it appears on the top of the stack again. But this is not entirely true, revealing that the model although powerful is incomplete. Speaker B might say, after a pause and somewhat longingly, "It was so nice there, what a great place to walk" referring back to Hawaii. Somehow the conversants would shift back to this focus, aided by the tense shift in B's utterances.

How do we know when a new discourse segment is introduced? In addition to the cue phrases mentioned above, the way reference is used signals a new discourse segment. Grosz, Joshi, and Weinstein use the term **backward-looking center** to refer to the entity in the current utterance that refers to the previous utterance [Grosz *et al.* 1983]. They suggested that as long as the center of the current utterance is the same as that of the preceding utterance, a pronoun should be used. If a pronoun is not used, this might suggest that a new discourse segment is being introduced. As the focus of conversation changes, the new topic may be introduced explicitly as the theme of a sentence, or it may be selected by reference from a series of things that have already been talked about (i.e., past backward-looking centers). Which of the possible backward-looking centers is selected depends on their ordering, which is dominated by recency.

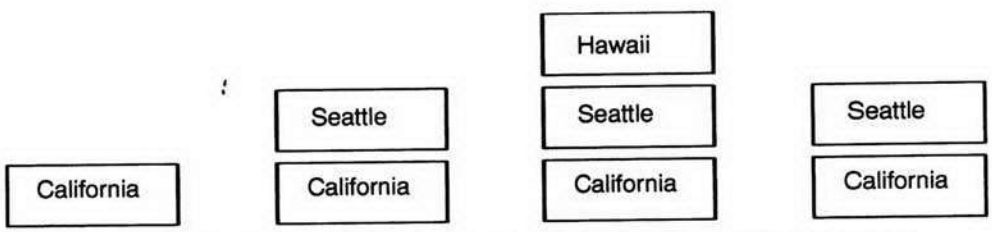

Figure 9.11. A series of snapshots of a stack model of the shift in focus of the weather discourse. Two new topics are introduced. The last utterance pops the top topic Hawaii to refer back to Seattle.

Intonation is a strong cue to shifts in focus. New information, i.e., potential new centers, are usually stressed more than old information. Intonational cues can also be used to override the default ordering of centers.

1: John hit Bill and then he called the police.
2: John hit Bill and then HE called the police.

In sentence one John calls the police; in sentence two Bill does. Ordinarily we would expect that "he" refers back to John, but if this word is emphasized as in sentence two it indicates to choose an alternate center, i.e., Bill.

Focus management requires the agreement of all participants in a conversation—if one party introduces a new topic without the appropriate cues, then other conversants do not receive the intended message. Conversation is a cooperative process. Another aspect of collaborative conversational behavior relates to synchronizing mutual beliefs. Clark *et al.* identify a number of conversational moves to make sure that both conversants agree to the identity of the current focus [Clark and Marshall 1981, Clark and Schaefer 1989, Clark and Wilkes-Gibbs 1986]. Clark and Brennan [Clark and Brennan 1990] introduce the concept of **grounding** to describe the process whereby we ensure that the listener understands our utterances as we intend them to be understood and that we agree that they stand for the same entities in the world.

Although much of conversation is purposeful and can be categorized as an attempt at communicating a concept or performing an action, conversation also serves a social nature. Part of the feedback provided by back channels, for example, informs the talker "I am paying attention to you. I am listening. I value your contribution." Sometimes talk exists as much to fill a communication space between people who feel awkward in silence as it does to change the other conversant's opinion or affect changes in the physical world [Goffman 1981].

CASE STUDIES

This section presents case studies of several projects that attempted to maintain interactive conversations utilizing the aspects of higher-level linguistic knowledge described in this chapter. Although only fragmentary use was made of the formalisms just described, these case studies offer some evidence of the potential of the topics in this chapter for enabling more sophisticated conversational interaction. Syntactic and semantic knowledge can be used to detect speech recognition errors and to determine the meaning of an utterance. Pragmatics relates an utterance to the larger world situation, and discourse structure helps cue appropriate responses.

Grunt

Grunt was an experiment that explored the utility of a discourse model in isolation from other linguistic knowledge [Schmandt 1988]. Grunt attempted to main-

tain a conversation by listening to the user, but without the use of word recognition. Without word recognition, the ability to negotiate conversational roles or topics was limited, so Grunt was engineered around the predefined, specific task of giving driving directions between two known points. The goal of the system was to employ acoustic feedback (or lack of it) from the user to improve his or her ability to understand and transcribe the directions. Grunt was complementary to its contemporary, Direction Assistance (described in Chapter 6), which allowed its user to choose a source and destination but did not offer any conversational flow control while reciting driving directions.

Grunt's discourse model was developed after recording and analyzing conversations between two people who could not observe one another, in which one was assigned the role of writing down the directions spoken by the other. Grunt's task was to present segments of the route as a series of turns, and the user was encouraged to engage in natural discourse behavior for conversational flow control. Subjects spoke to Grunt over the telephone without having been given any hints about its capabilities before the interaction; they had been told that they would hear driving directions to a bakery on the other side of town and that they were requested to write down the route.

Initially Grunt employed a very simple discourse model which was then elaborated over the duration of the project. Grunt would speak a sentence and then listen for a reply, relying on back channel utterances from the listener to speed up the conversation. Since there is great variability in how fast people transcribe directions, if Grunt waited after speaking sufficient time for the slowest writers to finish most others would become frustrated. Hence the need for flow control. A user's spoken response was detected by the change in audio level; at the end of the reply, Grunt would proceed to the next step of the bakery directions. If the user said nothing, Grunt would wait a rather long time and then proceed with the directions. To encourage back channel behavior by the user, Grunt would occasionally query a mute listener "Are you there?" (subjects always responded to the question) and offer hints such as "I'm listening to you too."[2] Such exchanges, called **channel checking** in [Hayes and Reddy 1983], are not uncommon during telephone conversations in which one conversant speaks much more than the other.

This model might have worked if the listener consistently and completely understood the directions when they were first spoken but this was not the case. People sometimes misunderstand each other, and successful communication was further hampered because the listener had to cope with Grunt's text-to-speech synthesis. When we cannot comprehend each other's directions, we ask for clarification; how could Grunt differentiate these requests from back-channel utterances? Based on the observation that back channel acknowledgment responses tended to be very short ("uh-huh," "OK," "hmm . . ."), Grunt measured the dura-

[2]As further encouragement of listener participation, subjects were asked "What is your name?" and "Are you ready?" by Grunt before any directions were given to help them realize that their speech could be heard by the computer.

tion of each utterance, and if the response was longer than 800 milliseconds, it assumed that the listener was confused and had asked either a question or for repetition. In response to this "clarification" reply, Grunt said "I'll repeat" and repeated the directions verbatim but at a slightly slower rate. Because Grunt's assumption that the listener was confused was based on skimpy evidence, it was important to explain "I'll repeat" so as to not risk misleading the user. For example, in the following dialog the user might insert an extra mile-long leg followed by a left turn into the directions.

Grunt:	Go about one mile and take a left at the first stop sign.
Listener:	I think I know where that is.
Grunt:	Go about one mile and take a left at the first stop sign.

Although extremely simple, the discourse model described so far (see Figure 9.12) was surprisingly effective and robust. In the constrained context in which subjects were exposed to Grunt, it behaved flawlessly for about one out of five users. Observations of failure modes in which the conversation digressed from Grunt's discourse model and its responses were therefore inappropriate resulted in several improvements.

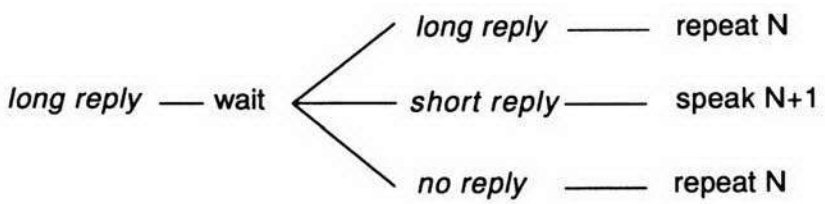

Figure 9.12. The states in Grunt's discourse model.

The first improvement extended the discourse model to cope with synchronization responses by the listener. If the listener was having difficulty keeping pace while writing, he or she might utter "Just a minute, please" or "Hold on a second." In normal human interactions, utterances such as these suspend the conversation, giving the requestor the floor until explicitly yielded.[3] This final return of the floor is often accomplished by exactly the same short back channel style utterances that Grunt already used as a cue, e.g., "OK". This timing cue was utilized as shown in Figure 9.13. When a long utterance was heard, Grunt no longer immediately repeated the current direction segment. Instead, it waited in anticipation of further speech to differentiate synchronization from clarification requests. If the next utterance was short, Grunt assumed that a synchronization

[3]Resumption of conversation after the granting of the synchronization request need not be explicitly triggered if it becomes obvious to the conversants that talk may continue. For example, if the note-taker were visible to the person speaking directions, the direction-giver could watch for the note-taker to finish writing or to look up.

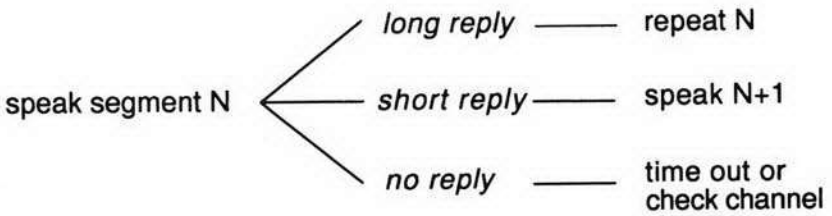

Figure 9.13. Modified Grunt behavior to detect synchronization requests.

cycle had just been completed and moved on. If the subsequent utterance was long, Grunt repeated the current segment assuming that the listener had previously asked for clarification, grown tired of waiting, and was now reiterating the request. This model thus provided for a correct response to a synchronization request and a delayed, but still correct, response to a clarification request.

The second improvement to Grunt's discourse model involved recognizing intonation to detect questions among the short utterances. Initially, Grunt erroneously treated short questions like "What?", "Where?", and "Left?" as acknowledgments. When this occurred, listeners became both annoyed and more confused since Grunt would apparently ignore their question and proceed with the next segment. With the improved model short utterances were analyzed for pitch, and those with clearly rising intonation were treated as clarification requests instead of acknowledgments (see Figure 9.14).

Toward the end of the project Grunt's discourse model was extended to include interruptions by the listener. When interrupted, Grunt would cease speaking, thus yielding the floor. The problem then was how to resume the dialog. Grunt could not know why the listener interrupted so it attempted to respond properly to any back channel acknowledgments. If an interruption was short and not a question, Grunt would repeat a few words and continue the utterance. If an interruption was longer, Grunt would wait for further input after it ceased speaking. After the listener spoke again, the current direction sequence would be repeated from the beginning as Grunt could not ascertain how much the listener had heard the first time.

Figure 9.15 depicts Grunt's final discourse model minus the interruption branches. Each of the advancements over Grunt's original, simple discourse model improved the probability that Grunt would do what the human listener expected;

Figure 9.14. Grunt analyzed short utterances to detect whether they might be questions and reacted accordingly.

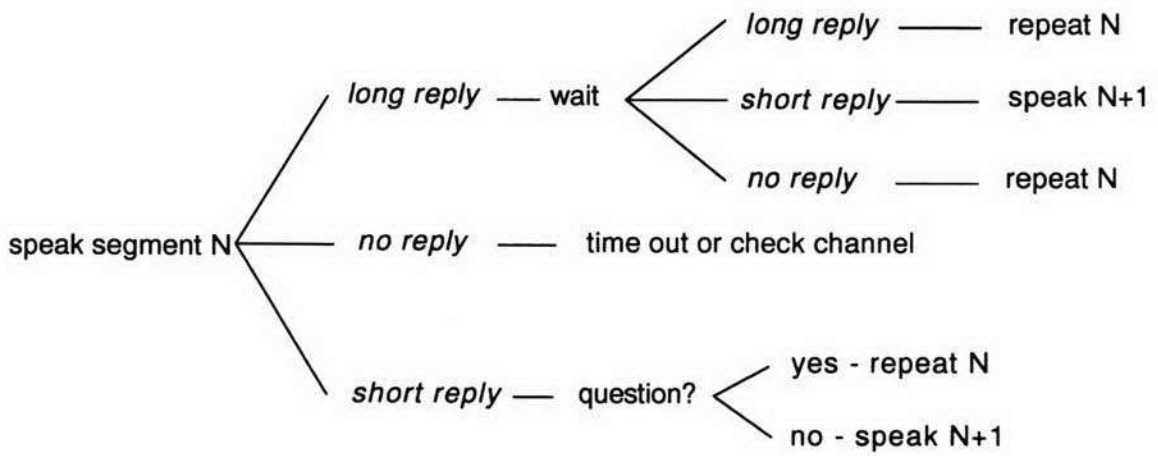

Figure 9.15. Discourse states for the final modified version of Grunt. Interruption is not shown.

this was based on observing breakdowns during its use. However, at its best, Grunt had considerable limitations and exhibited several distinct failure modes.

Summarizing or completing the talker's sentence is a back channel behavior that ordinarily moves a conversation along, and while transcribing directions many listeners invoked such very explicit flow control, but Grunt misinterpreted these as clarification requests. A related behavior is to recite word by word what one is writing as timing feedback to the direction-giver. Grunt interpreted these as a series of short acknowledgement utterances and became confused. Grunt tried to minimize the confusion resulting from false conversational moves by stating very explicitly that it was repeating or paraphrasing, which unfortunately sometimes led to wordy exchanges.

Other failure modes were caused by listeners changing their conversational behavior due to misconceptions of the discourse capabilities of computers. Almost one quarter of the participants said nothing while listening to the directions except when asked directly "Are you there?" Even when the period of time that Grunt waited during the silence between utterances was increased to the extent that interaction became exceedingly slow in the absence of acknowledgments, a significant number of listeners were simply mute, either not realizing that the computer could listen or feeling uncomfortable speaking to it. Some knowledgeable subjects believed that the computer was listening via speech recognition; based on the essentially correct understanding of recognition technology, they spoke in single word utterances, e.g., "Repeat," in order to be understood. But Grunt was not using word recognition, and these misguided attempts to cooperate broke the discourse model.

As a research project, it was intriguing to see how well Grunt could manage a discourse model in the absence of speech recognition. In a way, it was refreshing simply because so much work with recognition gets bogged down in syntactic and

semantic analysis and requires corrective discourse which, in turn, detracts from the naturalness of smooth conversation. The extent to which Grunt succeeded greatly surprised its designers, even though this success was heavily dependent upon having chosen a very limited and focused task with well-defined roles for the conversants. Nonetheless Grunt did demonstrate the power of even a minimal discourse model.

Conversational Desktop

Conversational Desktop, another Media Lab project (1984), provided an integrated voice interface to an integrated office telecommunications and message-handling environment [Schmandt *et al.* 1985, Schmandt and Arons 1986]. Conversational Desktop could place telephone calls, take voice messages, schedule meetings, and answer queries about airline itineraries; its functionality and telecommunications aspects are detailed in Chapter 12. The case study here discusses its language understanding and dialog-generation components, which utilized connected speech recognition for input and text-to-speech synthesis for output.

Conversational Desktop's conversation model provided a context in which to interpret spoken commands and also engage the user in a dialogue to correct recognition errors. Conversational Desktop's language understanding approach illustrates how syntactic, semantic, pragmatic, and discourse knowledge all contribute to conversation management. Although the contributions of this project at each layer were small and informal, their combination shows the richness of a conversational user interface.

The discourse system of the Conversational Desktop project can be thought of as a simple frame-based system. An empty frame was a data structure with fields for the different semantic classes of words in a small vocabulary. Some of these fields were filled in with specific words detected while parsing the user's input. Various procedures were then invoked to determine if sufficient information had been gathered to perform the requested action; these procedures could look to other sources of knowledge (databases) to support inferences.

In Conversational Desktop, the syntactic and semantic portions of language understanding were combined into a context-free grammar-based parser. Semantic knowledge was incorporated into the parsing process by basing the grammar on word categories more specific than the usual syntactic classes of noun phrase, verb phrase, preposition, etc. For example, Figure 9.16 illustrates a fragment of the grammar describing sentences such as "Schedule a meeting with Walter tomorrow at 10:00" or "Phone Chris and Barry." During execution, the parser recorded two categories of information when each grammar rule was applied. First, when any terminal symbols were encountered, both the word and its semantic role were stored in a frame-like structure describing the utterance. The frame had positions (roles) for the various "cases"[4] that might be found in a sen-

[4]The cases used were ad hoc and specific to the task as opposed to the more general cases described earlier in this chapter.

```
sentence := CMD | CMD_N NAME | CMD_NT N_AND_T

CMD_N := phone | where is

CMD_NT := schedule a meeting

N_AND_T := PERSON_NAME TIME  | TIME PERSON_NAME

NAME := PERSON_NAME | PERSON_NAME and PERSON_NAME

PERSON_NAME := Chris | Barry | Walter

TIME := DAY HOUR | HOUR DAY

Day  := today | tomorrow | Monday | Tuesday ...
```

Figure 9.16. Conversational Desktop's parser used a context-free grammar based on semantically meaningful word categories.

tence; as the parser recognized a word fitting into a case, that word was saved in the frame as an instance for its case.

The second result of the parser's analysis was a list of additional information that would be required to complete the sentence. For example, when the parser noticed the SCHEDULE-MEETING action, it would note that a PERSON, a DAY, and a TIME were required and add these word classes to the list. Similarly, when encountering AT used in the temporal sense, Conversational Desktop would note that a DAY and a TIME were required to complete the sentence. As each of these missing classes was located later in the parsing process it was removed from the list. If this list was not empty after further pragmatic and discourse analysis, Conversational Desktop had an incomplete understanding of the user's request and engaged in dialogue to identify each item in the list.

A unique attribute of Conversational Desktop's parser was the underlying assumption that the input from connected speech recognition would be error-prone. With the use of connected recognition, not only can any individual word be misrecognized, but the possibility of insertion and rejection errors also implies that even the number of words sent to the parser can be incorrect. The goal of the parsing process was to extract whatever useful information was contained in the recognized words. In other words, it was to identify the most likely utterance spoken given the words reported by the recognizer, the syntactic and semantic constraints of the context-free grammar, and knowledge of error probabilities.

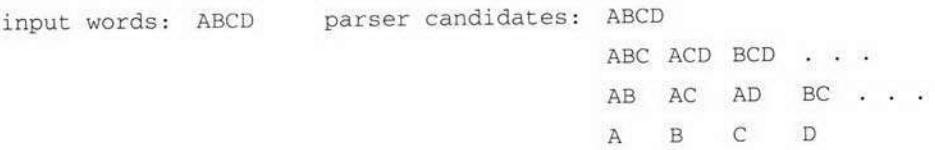

Figure 9.17 The Conversational Desktop attempted to parse all substrings of the input utterance. Each substring was evaluated, and those which could be parsed were candidates for selection.

Because any word reported by a speech recognizer might be a spurious insertion error, all substrings of the input words were parsed as shown in Figure 9.17. If *any* words were recognized correctly, one of the substrings would contain exactly all these words without the insertions. All of the substrings were submitted to the parser, which had to reject those strings containing grammatically correct insertions, but accept the substring that represented the original sentence minus the insertion.

Rejection errors caused incomplete sentences and resulted in fragments of syntactically correct tokens. To cope with such errors, the grammar was modified to describe sentence fragments as well as complete sentences, e.g., the word sequence "Barry Friday at two o'clock" is a well-formed fragment from a sentence about scheduling a meeting. This parsing strategy also handled those substitution errors resulting in semantically incongruous sentences; if one of the substrings contained exactly the correct words, then the resulting sentence fragment would be accepted by the parser's rules for fragments.

These parsing strategies had to be combined because any type of error could occur in each sentence. Because of the fragment rules, multiple substrings from each input sequence would usually be accepted by the parser; the next stage of analysis selected the best of these. Evaluation of "best" was based on empirically derived weightings of the following.

- **Completeness:** Assuming that the user spoke well-formed sentences, a successfully parsed complete sentence was more likely spoken than was a sentence fragment.
- **Number of words:** A greater number of words in an input string was preferred because recognition results were more likely to be correct than erroneous. This judgment was also based on the assumption that the user spoke sentences based only on words in the recognizer's vocabulary. If both "ABC" and "AB" could be parsed, "ABC" would receive preference from this metric.
- **Sequences of words:** Because of the difficulty of word endpoint detection, connected recognizers tend to produce runs of correct or erroneous results. If the endpoint of one word is incorrectly identified, not only is that word likely to be misrecognized but also the following word. The scoring metric favored substrings that included

consecutive tokens matching the input. Substring "AB" was favored over "AC" from input "ABC."

Once all candidate substrings had been evaluated and the best guess selected, further processing could be invoked if the user's request was not adequately specified by the recognized utterance. Pragmatic analysis consisted of resolving situational references in the context of the user's current or planned activity. For example, if the user said "Schedule both of us a meeting . . ." while on the telephone, Conversational Desktop assumed that the person at the other end of the connection should be included. Similarly, "When's my flight?" was evaluated with reference to the situational context; if asked while a call was in progress, flights to the caller's location were checked before the default, i.e., next scheduled flight. Pragmatic processing associated descriptions of events ("my flight") with known entities in databases (Boston to San Francisco flight on August 12th at 5:37 PM) so that action could be taken on behalf of the user.

At the final stage of language understanding, discourse analysis was invoked to resolve remaining ambiguities. This was a weak model of focus based on tracking the current task frame; there was no stack or other history mechanism in the model, so previous topics could not be recalled. The model of the current task allowed limited resolution of anaphora, thus supporting transactions spanning multiple sentences. For example, the user might inquire about someone's availability for a meeting: Conversational Desktop would consult schedules and suggest a time, which the user could accept by saying "Confirm it." Or having requested information about a scheduled activity, the user might command "Delete it."

Conversational Desktop also initiated dialogue. Despite the various stages of language analysis just described, errors in speech recognition often prevented complete identification of the user's request. When this happened, the list of missing roles for the sentence would not be empty, but instead it would specify missing information required to complete the request. Conversational Desktop phrased a question around the needed information but also echoed much of the request as understood so far: "At what time tomorrow do you wish to meet with Barry?" Echoing provided feedback and reassurance that the transaction was proceeding, and it could alert the user quickly when the recognizer had been severely mistaken. More importantly, revealing to the user what had been recognized allowed human detection of semantically correct recognition errors that Conversational Desktop could never detect. If, for example, the user had requested a meeting with "Harry," but this was recognized as "Barry," user intervention ("Change that to 'Harry'.") was the only recourse. Without echoing, this semantically correct error would go undetected.

Much like the research project Put That There (see Chapter 8), Conversational Desktop awaited further input after speaking a query. Except for a few special commands ("Change that" and "Cancel"), this additional input was merged with what had previously been recognized and the parsing cycle was repeated. This process continued until an action could be taken. As recognition accuracy decreased, it took longer for the user to complete a transaction, but the interaction rarely broke down completely.

SUMMARY

This chapter considered computational aspects of the higher-layer aspects of speech communication whereby words are combined into sentences and sentences into conversation and how these structures imply intentions and actions in the real world. Syntax and semantics embody the constraints whereby words are combined in certain sequences to form rational utterances. Pragmatics allows these utterances to carry force, while discourse provides a framework that structures a sequence of utterances into a coherent whole. To employ such concepts in interactive computer systems, underlying representations of each layer must be specified, and then procedures must be developed to operate on these representations.

The syntactic structure of a sentence may be represented as a tree, but a grammar is better defined by a set of context-free rules or an augmented transition network. Parsers analyze sentences in a top-down or bottom-up order as they apply these grammar rules. Semantic analysis requires representation of word meaning, and in particular it must differentiate the various senses in which a word may be used in a sentence. Case grammars represent the roles or functions a word plays in the action of a sentence, while semantic networks and type hierarchies describe the meaning of a word in terms of the other entities of which it is an instance.

Pragmatics relates the sentence both to the talker and to the situation in which the sentence is spoken; it links words to objects in the real world. This aspect of meaning, either general or specific to the current situation, requires a knowledge representation of which scripts and frames are examples. Pragmatics also considers the implicature of speech, i.e., the meaning beyond the words in a sentence such as "Do you have the time?" Speech act theory considers the utterances themselves to be actions in the world.

A number of aspects of conversation come together at the discourse layer. Discourse describes the methods and possible rules for determining who holds the floor, how talk is divided into turns, and the duration of a turn. Discourse also encompasses maintaining a focus across multiple utterances so that all conversants speak about the same topic and mutually agree on the referents for anaphora. Lastly, discourse provides a context in which conversants collaborate to discover or construct a common ground of experience and understanding so that the pragmatic implications of an utterance can be shared by all.

A case study description of Grunt illustrated that discourse models may be surprisingly useful in very constrained contexts even without word recognition. By building on accepted expectations of the flow of discourse, this simple mechanism added surprising vigor to the interface. A description of Conversational Desktop highlighted the interplay between language understanding components operating at all the communication layers described in this chapter. In this project, information gathered during parsing and further analysis was used to direct dialogue with the intent of resolving ambiguities which arose during conversational interaction.

FURTHER READING

Allen [1987] is an excellent textbook covering all aspects of the material discussed in this chapter. Winograd [1983] focuses on syntax, covering this topic in great detail. Levinson provides an excellent textbook on material included as both pragmatics and discourse in this chapter. Grosz *et al.* 1986 is a collection of readings on computational linguistics and includes many of the classic papers on plans and discourse models. Cohen *et al.* is another excellent collection of readings relevant to this chapter.