A Very Brief
INTRODUCTION TO COMPUTATIONAL LINGUISTICS

Ralph Grishman
New York University

1. INTRODUCTION

1.1. Why study natural language processing?

The ability to automatically analyze and understand natural language opens up a wide variety of applications. One of the first was **machine translation**; after many years of a relatively low level of effort, this area has seen a strong resurgence in the 1980's. Another application area is **information retrieval**; since much of the world's store of information is in written texts, systems which could understand these texts and extract information on request would have great value. The general area which has seen the greatest activity is **man-machine interfaces**, and in particular "question-answering systems" (natural language interfaces for data base retrieval). Current systems are still quite primitive, but such interfaces should make computer systems much more accessible to computer-naive users in the future.

In addition, work on the processing of natural language has provided new insights into language itself. It has encouraged the use of explicit procedural models and a wholistic view of the language faculty, including in particular the interaction of language and knowledge.

When used in concert with speech recognition, natural language processing has two roles to play. First, it can provide a rich set of expectations to aid the recognizer in identifying words. Second, for most functions (except dictation) we want a natural language system in order to *do* something in response to our utterance.

1.2. Our objectives

Our objectives in this very brief introduction are twofold. First, we want to describe how a combination of relatively simple mechanisms can provide us with a rudimentary natural language understanding ability. This should give you a good idea of how some of the systems now seeing the commercial light of day operate. Second, we want to point out in what respects these mechanisms only "scratch the surface" of our natural language abilities: how much more research needs to be done to develop a truly "natural" language facility.

1.3. An Outline

Our brief tour of natural language processing will be organized in three parts: **syntax analysis** (determining the structure of a sentence and the relationships between its words); **semantic analysis** (translating a sentence into a formal or readily interpretable language), and **discourse analysis** (identifying the relationships between sentences and the information implicit in a text).

This tutorial is organized roughly along the lines of my book, *Computational Linguistics: An Introduction* (Cambridge University Press, 1986). I have necessarily hit only a few of the highlights, and have been sometimes forced to oversimplify some issues. The tutorial is split into short sections corresponding, for the most part, to individual foils of the presentation.

2. SYNTAX ANALYSIS

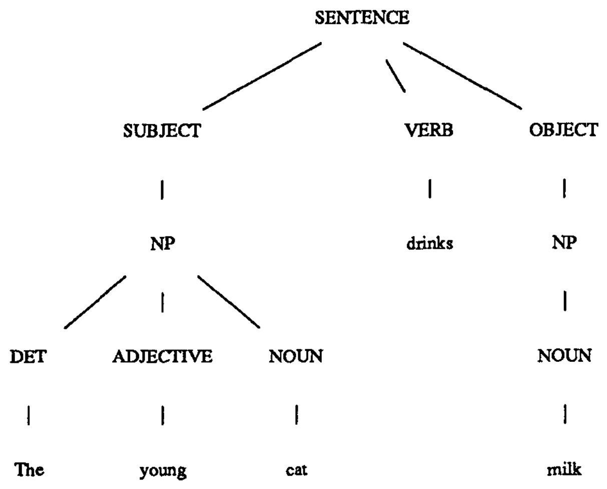
The first step in trying to figure out what a sentence means is trying to analyze the structure of the sentence: what the subject and object of the verb are, what words are modifying other words.

2.1. Phrase Structure

We want to divide the sentence up into phrases, and the phrases up into smaller consituents, until we reach individual words; you may have learned to "diagram" a sentence in this way. We can represent this structure by a *labeled bracketing* of the sentence:

$$[_{\text{SUBJECT}}\ [_{\text{NP}}\ [_{\text{DET}}\ \text{The}]\ [_{\text{ADJECTIVE}}\ \text{young}]\ [_{\text{NOUN}}\ \text{cat}]]]\ [_{\text{VERB}}\ \text{drinks}]\ [_{\text{OBJECT}}\ [_{\text{NP}}\ [_{\text{NOUN}}\ \text{milk}]]]$$

or equivalently, by a tree diagram:



(Some of the terms will be explained below.)

2.1.1. Formal grammar

We want to characterize the language by a set of rules, independent of the procedure we will use for analyzing the sentence. Such a set of rules is called a *formal grammar*. A formal grammar determines a set of grammatical sentences, and assigns a structure to these sentences. Our challenge is to develop a formal grammar which matches the intuitions of grammaticality of speakers of the language, and which assigns structures which are useful in determining the meaning of the sentence.

2.1.2. Phrase structure rules

Most computationally oriented grammars are based on, or are extensions of, context-free phrase structure rules. Each such rule describes one type of sentence constituent, specifying how it is composed from words or other sentence constituents. For example,

sentence → subject verb object

says that a sentence is composed of a subject followed by a verb followed by an object. Similarly,

subject → *noun | *adjective *noun

says that a subject is composed *either* of a noun *or* of an adjective followed by a noun. Alternatives are separated by "|". Symbols designating classes of words are prefixed by "*".

2.1.3. A Simple Grammar

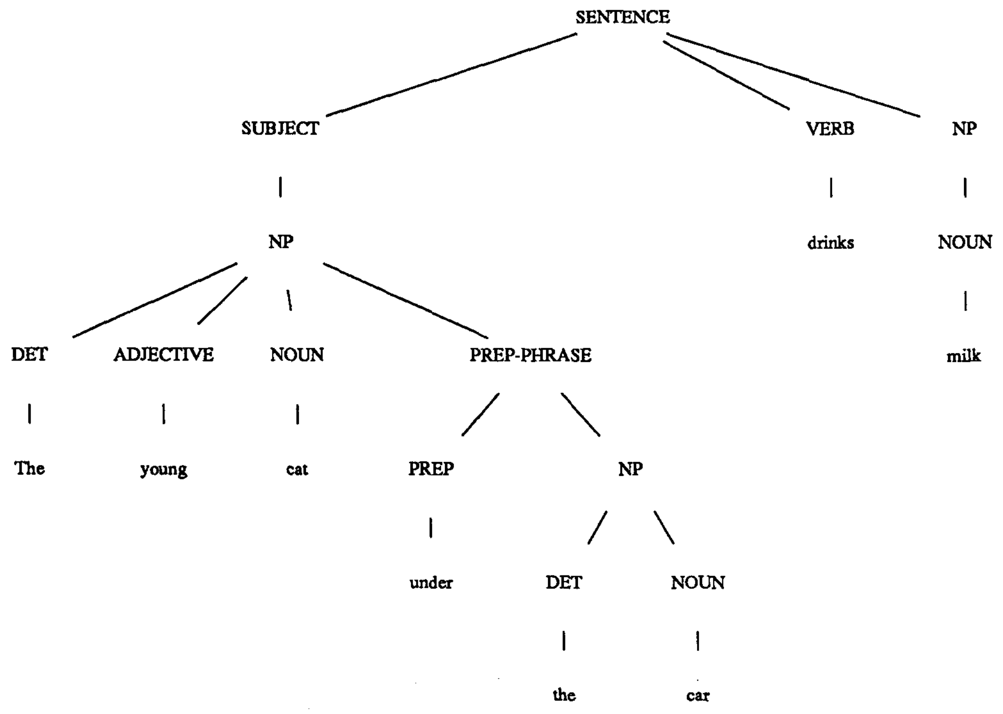
sentence	→ subject *verb object
subject	→ np
np	→ [*det] [*adjective] *noun [prep-phrase]
prep-phrase	→ *preposition np
object	→ np	null

The brackets in these rules indicate optional elements (that an article and adjective are optional before a noun, for example). The symbol "np" stands for "noun phrase", and "det" for determiner (an article, such as "a" or "the", or a quantifier, such as "some" or "every"). This simple grammar can generate a wide variety of sentences, such as

Cats eat fish.
The young cat under the car drinks milk.

2.1.4. A Sample Parse

Given this grammar, the sentence "The young cat under the car drinks milk." would be assigned the structure:



2.2. Parsing

Parsing means analyzing a sentence with respect to a grammar: determining if the sentence is grammatical, and what the structure of the sentence is.

2.2.1. Top-down vs. Bottom-up

Parsers are usually classified as *top-down* or *bottom-up*. These terms refer to the direction in which the parse tree is built.

A top-down parser starts with the *sentence* node. It thinks as follows: I'm trying to decide if these words are a sentence. A sentence is defined (in the grammar) as a subject, a verb, and an object. Let me first look for a subject. A subject is a noun-phrase, so let me look for a noun phrase. A noun-phrase may begin with a determiner -- is there a determiner here as the first word? Yes, let me look next for an adjective and then a noun; if I find all three, I've succeeded in finding a noun-phrase.

A bottom-up parser starts with the words, and builds a tree upwards. It thinks as follows: Here's an article, followed by an adjective, followed by a noun. Are there any constituents made up of these three word classes. Yes -- it's a noun-phrase. This noun-phrase could be a subject; it could also be an object. Etc.

2.2.2. Bottom-up Algorithm

The basic strategy for bottom-up parsing is quite simple. Starting with the sentence words, we look for sequences of words or constituents which we can link together to form a larger constituent. We repeat this process until we cannot build any more constituents. We then look for any constituents named "sentence" which cover all the words of the sentence. These constituents are the parses of the sentence.

2.2.3. An example of Bottom-up Analysis

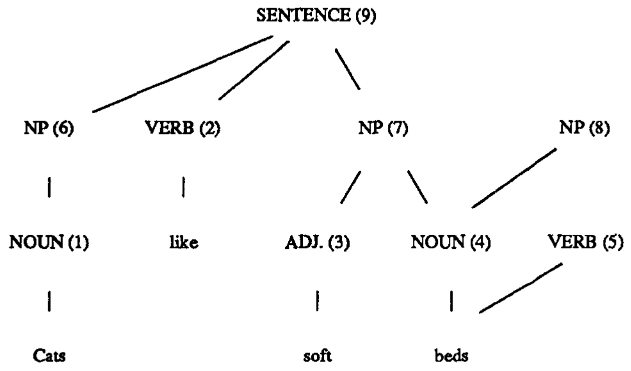
To show how this works, let us use an extremely simple grammar:

sentence	→ np *verb np
np	→ *noun I *adjective *noun

then, with the sentence

Cats like soft beds.

we would build the constituents



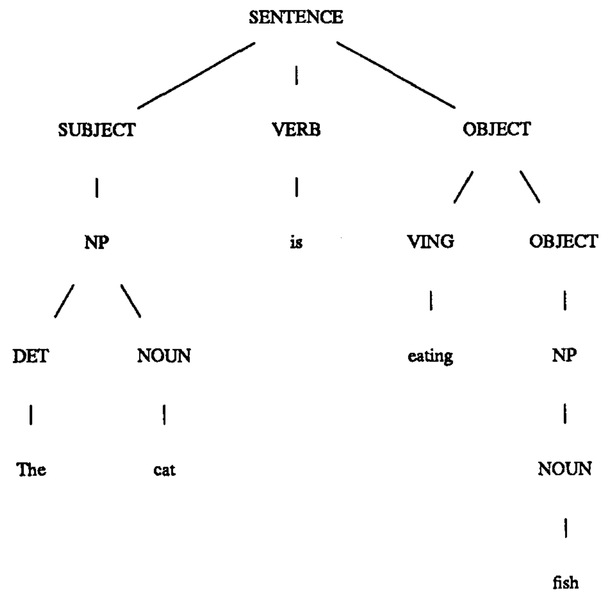
in the order given in parentheses after the node names. In this example there are two nodes which are created but not used in any larger constituent: VERB (5), corresponding to the possible usage of "beds" as a verb (as in, "He beds down for the night."), and NP (8), corresponding to the analysis of the word "beds" as a complete noun phrase (without the modifier "soft"). For a larger grammar, there would be many more such "dead ends".

2.3. Elaborating the grammar

The grammar given above (section 2.1.3) is so restrictive that it avoids a number of basic issues. Therefore, before we proceed further we will indicate how the grammar can be extended a little bit.

2.3.1. Progressive Tense

The progressive tense uses a form of "be" plus the present participle of the verb: "The cat is sleeping.", "Mary is eating corn.". There are several ways of extending the grammar to handle such forms. One possibility is to consider "is" to be the main verb, and the present participle plus its object to be (together) the object of the sentence, so that "The cat is eating fish." would be analyzed as:



We can include such structures in our grammar by adding the rule

object → *ving object

where "*ving" is our name for present participles (verbs ending in "ing").

2.3.2. Passive

In comparing a passive sentence

The cake was baked by Sam.

with the corresponding active sentence

Sam baked the cake.

we see that three things have happened: the object of the active sentence has moved into subject position, the subject has moved into a "by" phrase, and the verb has changed to a form of "be" + the past participle. The net effect is that we still have a noun phrase in the subject, but we now have a "by" phrase where the object used to be. We can therefore analyze passives by treating "be" as the main verb (as we did for progressives) and adding the production

object → *ven "by" np

to our small grammar (where "*ven" is our name for past participles, which usually end in "en" or "ed").

2.3.3. Relative Clauses

Consider the following relative clauses:

The man *whom I met* comes from Philadelphia.
The man *who opened the door* comes from Detroit.
The man *whom I sold the book to* comes from Miami.

Can we give a unified account of these different structures? In each case the phrase following "who" or "whom" is itself a full sentence from which a single noun phrase has been omitted. We can make this explicit by putting back the omitted words, enclosed in brackets:

The man *whom I met [the man]* comes from Philadelphia.
The man *who [the man] opened the door* comes from Detroit.
The man *whom I sold the book to [the man]* comes from Miami.

In terms of our grammar, we could modify the noun-phrase rule as follows:

np	→ [*det] [*adjective] *noun [noun-modifier]
noun-modifier	→ prep-phrase I "who" sentence I "whom" sentence

Note the recursive structure: the entire sentence may contain a smaller sentence structure within it. To handle the omission within the relative clause, we must allow exactly one np within the relative clause to take the value *null*. This requirement is not readily stated within our phrase-structure rules.

2.4. Syntactic Constraints

The rules given above can be used to generate quite a variety of sentences; unfortunately, they can also generate quite a variety of ungrammatical sentences, such as

The cats eats fish.
The cat sleeps fish.
The cat has sleeping.
The cat which cat eats fish is sleeping.

These sentences violate particular syntactic constraints; we shall consider some of these constraints in this section.

2.4.1. Some Constraints

Here are a few of the constraints not captured by our rules:

number agreement. The subject and verb must agree in number ("Cats sleep." but not "Cats sleeps."). Also, the determiner and noun must agree in number ("A cat ..." but not "A cats ...").

count noun. A singular noun representing a countable entity must be preceded by a determiner ("The cat is eating." but not "Cat is eating.").

subcategorization. Only certain types of objects may appear with certain verbs ("The cat sleeps." but not "The cat sleeps fish.").

omission. Exactly one np should be *null* within a relative clause.

2.4.2. Why enforce the constraints?

One's first reaction, when seeing all these constraints, is "Why bother to enforce them?". After all, we expect our input to be reasonably well-formed sentences, not gibberish like "My cat sleeps fish.", so it seems safe to assume that these constraints will be satisfied.

However, if we try parsing some sentences with a simple grammar which doesn't check these constraints, we will discover that we get quite a few parses, even for rather innocuous sentences. For example,

The bird can fly.

will get (in addition to the correct parse) two other parses: one parse analogous to "The workmen can tomatoes.", the other to "The trash can flies [through the air]". Both of these parses violate number agreement (the first violates the count noun constraint as well), so a grammar which checked these constraints would give us only the correct parse.

In addition, for speech recognition we would like to have as many constraints as possible in order to narrow the range of possible words we should expect.

2.4.3. How to enforce the constraints

Most of the constraints we have mentioned are not easily captured by extending the phrase structure rules. To add subject-verb number agreement to our simple grammar, for example, we would have to double the sentence, subject, and np rules:

```
sentence          → singular-subject *singular-verb object |
                    plural-subject *plural-verb object
singular-subject  → singular-np
plural-subject    → plural-np
singular-np       → [*det] [*adjective] *singular-noun [*prep-phrase]
plural-np         → [*det] [*adjective] *plural-noun [*prep-phrase]
```

Some systems get around this problem by associating features (such as syntactic number) with the nodes of the parse tree, and extending the rule formalism to assign and test these features. Other systems allow the grammar writer to write procedures which enforce the constraints by checking the properties of the words. Such systems are called *augmented context-free grammar* systems. ATNs (augmented transition networks) are a form of augmented context-free grammar in which the phrase structure component is represented by networks rather than by rules as shown above.

2.4.4. Expressing constraints as procedures

An augmented context-free grammar consists of a set of context-free phrase structure rules, such as those we gave above, plus a set of procedures which enforce grammatical constraints. These procedures are associated with particular definitions in the grammar; for example, the procedure for subject-verb number agreement would be associated with the rule for sentence. When this rule is used to build a new node of the parse tree, the agreement procedure is invoked. It checks the syntactic number features of the verb and the head (main noun) of the subject; if they don't match, the procedure fails and the node is discarded.

2.4.5. A grammatical restriction

Different systems use different languages for expressing these restrictions. Most ATNs use LISP, and provide special predicates for testing and recording features. The systems at NYU (the Linguistic String Parser and PROTEUS Parser) use a language called Restriction Language designed for stating these restrictions. For example, in Restriction Language a simple number agreement restriction might be written

```
WAGREE = IN SENTENCE:
 BOTH IF THE VERB IS PLURAL THEN THE CORE OF THE SUBJECT IS PLURAL
  AND IF THE VERB IS SINGULAR THEN THE CORE OF THE SUBJECT IS SINGULAR.
```

(the "CORE" is the main noun of a noun phrase).

2.4.6. Expressing constraints using features

As we noted above, an alternative approach is to associate features with the nodes of the parse tree and extend the rule formalism to assign and test these features. For example, we can introduce the feature *number* with values *singular* and *plural*, and associate it with nouns, verbs, noun phrases, and subject nodes:

```
sentence           → subject<number> *verb<number> object
subject<number>    → np<number>
np<number>         → [*det] [*adjective] *noun<number> [*prep-phrase]
```

If the feature marker *<number>* appears at two places in a single production, the values of the feature at the two places must be equal. Thus the number feature of the np must be the same as that of the noun, the number of the subject the same as that of the np it dominates, and the number of the subject and verb in a sentence must be equal.

2.5. Regularization

2.5.1. Syntactic variety

A basic function of syntactic analysis is to establish the relationships among the constituents in a sentence. In the sentence

Sam baked a cake.

the subject is the thing doing the baking and the object is the thing being baked. If we look at closely related syntactic forms, however, we see that this relation no longer holds. For example, in the passive

A cake was baked by Sam.

the subject is now the thing that was baked, while the thing doing the baking has moved to a "by" phrase. In the progressive form,

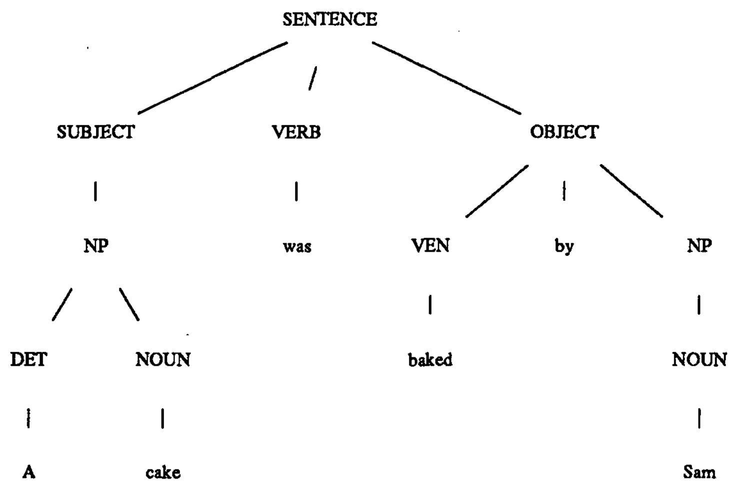
Sam is baking a cake.

"is" has now become the verb constituent, and the "main verb" is within the object constituent.

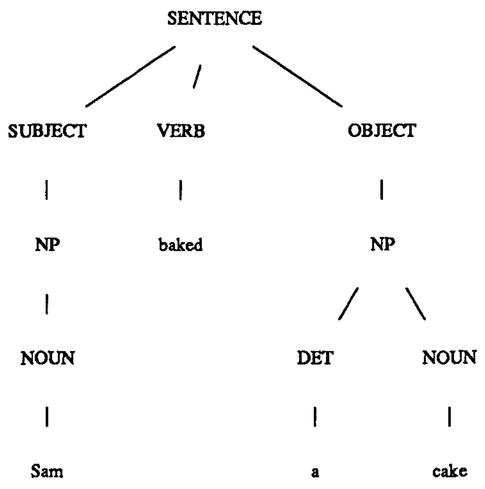
2.5.2. Reducing the variety

This syntactic variety obscures the common functional relationships among the act of baking, Sam (the baker), and the cake (the think baked) in these three sentences. We can clarify these relationships by reducing all of these forms to a standard form, such as the simple active sentence. This process of *syntactic regularization* is a part of most natural language systems. It simplifies the next stage of processing -- semantic analysis -- which relies heavily on the functional relationships.

In some systems, regularization is done by transformations which act on the parse tree, transforming, for example, the passive sentence structure:



into an active sentence structure:



In other systems (in most ATNs, for example), the regularized structure is built up incrementally during parsing.

3. SEMANTIC ANALYSIS

So far we have focussed on determining the structure of natural language sentences. This,however, is rarely our final objective. Rather, we are concerned with understanding what the sentences *mean*, or performing some action in response to a received sentence. We will begin by looking at the question of meaning at a slightly abstract level, and then in a short while will connect this up with an application using natural language input.

3.1. Semantic Representation

What does it mean to understand a sentence? One answer, for declarative sentences at least, is to say that we understand a sentence if we can determine, under any given set of circumstances, whether it is true or not. The usual approach to this is to select a formal language for which the rules of evaluation are simple, and translate the natural language sentences into this language. We shall use predicate logic with restricted quantifiers for this task, and shall call the representation of a sentence in predicate logic its *logical form.*

3.1.1. Predicates

Our world will be described in terms of a set of *objects* and a set of *predicates.* The predicates are functions whose arguments are objects and whose value is *true* or *false.* For example, we could have a "microworld" inhabited by Tom, Dick, Harry, and Jane. We can have predicates like "male" (which takes one argument) and "father-of" (which takes two arguments). The current state of the world can be described by listing, for each predicate, the values of the arguments for which it is true. For example,

male(Tom)
male(Dick)
male(Harry)
father-of(Tom,Dick)
father-of(Tom,Jane)

We would then analyze a sentence such as

Tom is the father of Jane.

by translating it into a predicate with arguments

father-of(Tom,Jane)

and then seeing if the value of the predicate for those arguments is *true.*

3.1.2. Quantifiers

With predicates alone, we are very restricted in the range of sentences we are able to translate. In order to handle sentences such as

Everyone is mortal.

we need to introduce *quantifiers* into our formalism. We will introduce two quantifiers: existential quantifiers and universal quantifiers. A formula with a universal quantifier, such as

$(\forall x) P(x)$

says that P is true for every object in our world. A formula with an existential quantifier, such as

$(\exists x) P(x)$

says that there is some object for which the predicate P is true. Thus in a world in which the only objects are people, the sentence

Everyone is mortal

would be translated to

$(\forall x)$ mortal(x)

3.1.3. Restricted Quantifiers

Of course, we don't live in a world in which the only objects are people, so we will introduce *restricted quantifiers.*

$(\forall\ x : Q(x))\ P(x)$

is true if, for *every* object for which Q is true, P is true too.

$(\exists\ x : Q(x))\ P(x)$

is true if, for *some* object for which Q is true, P is true too. Then, in a world with people and other things, we would translate

Everyone is mortal

to

$(\forall\ x : \text{person}(x))\ \text{mortal}(x)$

3.2. Quantifier Analysis

Having defined a semantic representation, we should now sketch a procedure for mapping our (regularized) syntactic structures into this representation.

3.2.1. Simple sentences

The simplest sentences are those involving only constants (names). For example,

Mary loves Tom

could be translated into

loves(Mary,Tom)

Thus, the verb is translated into a predicate and the subject and object are translated into arguments.

3.2.2. Quantifiers

When the subject or object involves an English quantifier, we have to translate the noun phrase into a quantifier governing one of the arguments of the predicate. For example,

Every student loves Mary.

could be translated to

$(\forall\ x : \text{student}(x))\ \text{loves}(x,\text{Mary})$

Note that the head of the noun phrase translates into a restriction on the quantifier. If both subject and object have quantifiers, we will end up with two in the logical form:

Every student has a terminal.

becomes

$(\forall\ x : \text{student}(x))\ (\exists\ y : \text{terminal}(y))\ \text{has}(x,y)$

3.2.3. Noun phrase modifiers

If the noun phrase has modifiers, these will translate into further restrictions on the quantifier:

Every student has a red notebook.

becomes

$(\forall\ x : \text{student}(x))\ (\exists\ y : \text{notebook}(y)\ \&\ \text{red}\ (y))\ \text{has}(x,y)$

3.2.4. Relative clauses

If the noun phrase modifier is a relative clause, the approach is the same -- it translates into a restriction on the quantifier -- but the formulas become more complicated. For example,

Every student who has a terminal has a modem.

would translate into

$(\forall x : student(x)\ \&$
$\quad (\exists t : terminal(t))\ has(x,t))$
$(\exists m : modem(m))\ has(x,m))$

To do this translation, our procedure must operate *recursively*: we first translate the relative clause "who has a terminal", in much the same way as we would a simple sentence. This produces the second line of the logical form above. We then use this as a quantifier restriction in creating the translation of the entire sentence. This recursive translation procedure parallels the recursive syntactic structure we introduced for relative clauses.

3.3. Data Base Retrieval

The semantic representation and quantifier analysis procedures may be quite elegant, but they may also seem quite useless. After all, no one is likely to pay us a lot of money for a program which prints out logical forms; they want a natural language system to *do* something. The "best-sellers" among natural language programs these days are "question-answering systems": natural language interfaces for data base retrieval. We will therefore consider how our semantic analyzer can be readily transformed into a simple question-answering system.

3.3.1. Predicates and Relations

Let's suppose we have a relational data base. The relations in the data base can be viewed as predicates: the predicate P(a,b,c) is true if the relation P has a row with values <a,b,c>. Thus if we have a query such as

Is Frank employed by NYU?

we would generate its logical form,

employ(NYU,Frank)

and then, treating this as a data base query on relation "employ", see whether it is true or false and then respond "yes" or "no".

3.3.2. Interpreting Quantifiers

Quantifiers have a very direct procedural interpretation: to evaluate

$(\forall x : R(x))\ P(x)$

we iterate over all the objects in our world, and for those for which R is true, check that P is true. Similarly, for

$(\exists x : R(x))\ P(x)$

we iterate over all the objects in our world, and look for one for which both R and P are true. For a large data base, of course, this will be inefficient, but -- depending on the data base query language -- there will typically be more efficient approaches. The existential quantifier shown, for example, may be realized as a join of P and R.

3.3.3. Wh Questions

A wh-question (one beginning with "who", "what", or "which") can be interpreted as a request to determine the set of values for which a formula is true. For example,

Which students own a typewriter?

can be interpreted as

(find the set of x : student(x))
$\quad (\exists y : typewriter(y))$
$\quad\quad own(x,y)$

In data base terms, this means returning the set of values of one attribute of the relation, rather than just a "yes" or "no".

3.3.4. Verbs

In doing our semantic analysis, we have assumed that each verb corresponds to a predicate (or relation) of the same name. However, one of the benefits of a natural language interface lies in the ability to refer to the same relationship in several ways, and the ability to refer succinctly to complex relationships (which may not be directly recorded in the data base). For example, using an employment data base, we would want the system to accept either

How many people work for XYZ?

or

How many people are employed by XYZ?

This indicates that, in general, several verbs may be translated into a single predicate. If the system has historical data on individual hirings and firings, we would probably want to allow

How many people were rehired last year?

and have "rehire" translate into a complicated condition involving a firing and a subsequent hiring.

3.3.5. The structure of a question-answering system

Putting all the pieces together, even a simple question-answering system will have the following stages:

Parsing
Syntactic Regularization
Translation to Logical Form
Translation to Data Base Query
Retrieval

If the system does anything fancier, such as analyzing pronouns (to be discussed below), this will normally be done in terms of the logical form, before the translation to a data base query.

3.3.6. What's missing

The last few sections might suggest that we know all there is to about building good question-answering systems. In fact, current systems are still very rudimentary -- not at all a full natural language interface. Because such systems rarely incorporate any deep model of what is in the data base, or what the user's interest might be in querying the data base, they are very limited in their responses. Few systems allow questions about what type of information is in the data base ("What do you know about company XYZ?"). Many systems interpret questions literally, even if that is clearly not their intent ("Do you have any record of Joe's salary?" -- "Yes."). And no system provides helpful feedback for questions which fall outside the semantic model.

3.4. Semantic constraints

In our discussion of syntactic analysis we pointed out that not all the sentences generated by our phrase structure rules were grammatical. To account for this we introduced various syntactic constraints into our grammar.

In semantic analysis, our aim is to decide whether a sentence is true. However, some of the sentences which are grammatical are so nonsensical that we might be reluctant to identify them as true or false; for instance,

The closet likes scrambled eggs.

or

The road is wearing a brown hat.

If we want to build a practical natural language system, why would we be interested in identifying such nonsensical sentences (rather than, say, just considering them to be false)? Surely we don't expect them to appear as input. Our answer is much the same as it was for grammatical constraints: constraints which in one case separate sensible from nonsensical sentences may in other cases separate correct and incorrect readings of a sensible sentence. Consider for example

I passed a man on the road wearing a brown hat.

Syntactically, this sentence is ambiguous: is the man or the road wearing the hat? If we have a constraint that roads don't wear hats, we could block the incorrect syntactic analysis.

3.4.1. Predicate domains

How should we characterize and organize these facts about sensible and nonsensical sentences?

First, we can observe that these facts are best stated as constraints on semantic structure, not on syntactic form. If

The road is wearing a brown hat.

doesn't make any sense, then neither does

A brown hat was worn by the road.

or any other form of the sentence. So, if we translate the verb "wear" into a predicate "wear", we can state this as a constraint on the predicate: that

wear(road,hat)

is a nonsensical combination of predicate and arguments. More generally, we will want to assert that this predicate is meaningful only for certain values of the arguments; we call this the *domain* of the predicate.

For any realistic subject area, enumerating separately the domain of each predicate would be an overwhelming task. However, many predicates share domains; these domains correspond in many cases to generally recognized "semantic classes". For example, we might say that the domain for the first argument of wear is the set of animals (including people):

The man wore a hat.
The horse is wearing a saddle and horseshoes.
The dog is wearing a sweater.

This set is also the predicate domain for many predicates associated with animal functions: seeing, breathing, sleeping, eating, chewing, etc.

Thus, we would specify these semantic constraints by defining a set of semantic classes (typically as a hierarchy of broader and finer classes) and then specifying the predicate domain of each predicate in terms of these classes.

3.4.2. Limitations of predicate domains

The use of predicate domains is particularly successful in dealing with texts in clearly restricted subject areas, such as technical and scientific reports. It is less successful in dealing with texts which range over a broad area, such as fiction or newspaper stories. Such texts involve many different types of objects, making a classification difficult, and may include metaphorical and imaginary usages, which blur the lines of predicate domains.

It is also important to recognize that these semantic classes provide only a relatively general constraints. In some cases we will need much more detailed information about what is possible or impossible in order to understand a sentence correctly. Consider for example the two sentences

I left the toaster in the kitchen on the floor.
I left the toaster in the kitchen on the first floor.

"on the floor" indicates where in the kitchen you left the toaster; "on the first floor" indicates where the kitchen is.

3.5. Anaphora

In formal and programming languages, when we want to refer to something more than once we generally have to give the object a name and refer to it by name. Natural language provides a much more flexible means for referring to entities previously mentioned in a text. Such references are called *anaphoric references*. The most familiar form of anaphoric reference is the pronoun:

I bought an ice cream cone.
It was delicious.

The previous noun phrase to which the pronoun refers is called the *antecedent*. Noun phrases with "the" are also frequently used anaphorically:

I bought an ice cream cone and a hot dog.
The cone was delicious.

3.5.1. Using predicate domains

The simplest cases of anaphora can be accounted for quite straightforwardly using the notions of predicate domains and semantic classes which we just introduced. We begin by translating the sentence with a pronoun into logical form, treating the pronoun just as we would other noun phrases. We then determine the predicate domain for the argument position occupied by the pronoun, look for the most recently mentioned noun phrase belonging to that semantic class, and consider that the antecedent of the pronoun. A translation of the antecedent then replaces the pronoun in the logical form.

For example, if we read

> Sam bought a hat from the store on Wednesday.
> Ted wore it on Thursday.

we might translate the second sentence to

> wear(Ted,it,Thursday)

The predicate domain for the second argument of "wear" is "clothing", so we would look for the most recently mentioned noun phrase in that class. We find "hat", identify it as the antecedent, and then, roughly speaking, replace "it" by "the hat" in the logical form. (Strictly speaking, we will replace "it" by the logical representation of "the hat which Sam bought from the store on Wednesday".)

3.5.2. When domains aren't enough

This approach works quite well, but -- as in the case of syntactic ambiguity -- there are cases where more detailed information is needed. Winograd created an oft-repeated pair of sentences to illustrate this:

> The city council refused to grant the women a parade permit because they advocated violence.
>
> The city council refused to grant the women a parade permit because they feared violence.

In one case "they" refers to the council, in the other case to the women; making the proper choice requires rather detailed reasoning about the concerns of the city council.

3.5.3. Contextual reference

In many cases a definite noun phrase ("the ...") refers not to something explicitly mentioned previously, but rather to something *related* to a previously mentioned object or activity. This is termed *contextual reference.* Thus in

> I bought an apartment with a small kitchen. The stove is in the middle, the dishwasher underneath, and the refrigerator on the ceiling.

we understand "stove", "dishwasher", and "refrigerator" because we expect them as part of a kitchen. We would be perplexed if the second sentence said "The bulldozer is in the middle, the parakeet underneath, and the bed on the ceiling.". Such references, therefore, rely on a quite detailed knowledge of the structure of objects and actions.

4. DISCOURSE ANALYSIS AND KNOWLEDGE

4.1. The importance of world knowledge

4.1.1. Syntactic ambiguity and contextual reference

In the last section we saw examples where substantial world knowledge was needed to correctly understand some natural language input. We noted that constraints based on predicate domains were not sufficient for resolving many syntactic ambiguities. For example, if you told a robot to

> Broil the steak on the top shelf of the refrigerator.

it might start by building a small fire in your refrigerator. Similarly, when cooking was finished, if you told it to

> Put the chicken on the table and cut off the legs.

you might get the response "What legs?" or, even worse, a shorter table. To determine that "legs" *might* refer to "chicken", our robot must know about the structure of things (in this case, chickens); to determine that "legs" *does* refer to "chicken", it must know about appropriate actions (in this case, in serving food).

4.1.2. Analyzing texts

The need for world knowledge is particularly acute in analyzing multi-sentence texts. We have the problems mentioned above, such as syntactic ambiguity and contextual reference. In addition, we have the task of deciding how the sentences are related. For example, we wouldn't say that we had understood the following passage

> John threw a cream pie. Mary ducked, and Tom got hit smack in the face.

unless we recognized the relation between the events involved. In any text there will be one or more relations -- cause, time, sequence, elaboration, etc. -- tying the sentences together. These relations will rarely be explicit, yet we must identify them in order to properly understand the text. We must therefore rely on extensive background knowledge to infer these relations from the facts explicitly presented.

4.1.3. Organizing world knowledge

It's all well and good to say that we need to incorporate a lot of world knowledge in our system, but this doesn't tell us what to *do* in constructing a language processing system. How should we collect this knowledge, how should we organize it, and how should we use it in the language analysis? The answers to these questions are as yet poorly understood. They have been addressed only for certain types of knowledge, relevant to the understanding of certain limited classes of texts. We examine in this section two types of knowledge which have been applied to language analysis.

4.2. Analyzing narrative

The first type of text we will consider are narratives about stereotyped sequences of events. We are all familiar with such sequences. As we are growing up we learn, in some detail, what to do in certain social situations: eating in a restaurant, buying food in a supermarket, visiting a doctor's office, taking a plane trip.

4.2.1. Hitting the highlights

Because we share knowledge of these sequences, we don't have to provide all the details when describing such an event; we just have to present the highlights or unexpected events. The listener should be able to fill in the gaps, and tie together the explicitly mentioned events, using the shared knowledge. For example, if you hear

> I went to Clancy's restaurant yesterday. The soup was cold and the steak was tough, so I left
> the waiter a small tip and vowed never to go back.

you can understand phrases like "the soup", "the waiter", and "a tip", and the reason for the small tip, from your knowledge of restaurants.

4.2.2. The script

How should we record our knowledge of such stylized sequences? Schank suggested a structure called the *script*, which is a kind of flowchart. This flowchart involves *actors* (basically, people) and *props* (objects); it consists of a series of primitive actions performed by the actors. For example, a restaurant script (the original example) would include such actors as the customer and waiter, and such props as the food, the check, and the tip. It might include steps such as

> customer enters restaurant
> customer goes to table
> waiter comes to table
> customer gives order to waiter
> waiter brings food to customer
> customer eats food
> waiter brings check to customer
> customer gives money to waiter
> waiter brings change to customer
> customer leaves tip
> customer leaves restaurant

(in an actual script, these would be more detailed and in a formal representation). It could also include various conditional information, such as the relation between the food served and the tip. An actual story, such as the one above, can then be matched against items within the script. Once this matching is done, the script provides the desired connections to tie together the sentences and resolve the contextual references.

4.3. Analyzing text about equipment

The second type of text we shall consider is reports about pieces of equipment -- your car, your radio, or your personal computer. Here too we see the effect of shared knowledge -- knowledge in this case about the structure and function of *objects* rather than *actions*. Suppose we hear

> The car started to overheat. We opened the hood and saw that the fan belt was broken.

Then -- if we know something about cars -- we recognize that the broken belt probably caused the overheating, and that opening the hood let us see the broken belt but probably didn't cause the belt to break.

As with other texts, making these connections is an essential part of understanding the texts. In this case the relevant background knowledge is a simple model of the car: what the components of the car are, what the function of each component is, and how these components interact in the operation of the car. In addition, we require a mapping which relates the predicates of our logical form ("break", "overheat") to states of the model. Then, once the individual sentences of the report have been analyzed into logical form, discourse analysis can use the model to identify the implicit causal relations.

5. CONCLUSION

To organize our quick trip through computational linguistics, we have divided the problems and techniques into three areas: syntactic analysis, semantic analysis, and integration with "world knowledge". Some such subdivision of the problem is essential if we are to successfully address the myriad problems of natural language. This division also corresponds roughly to the stages of processing in many current natural language systems. This division in processing is more problematic: it reflects in part our current difficulty in developing an integrated framework and analysis procedure which will allow us to apply all these constraints -- syntax, semantics, and world knowledge -- in a uniform fashion.

SPEECH TUTORIAL

Edward P. Neuburg

Introduction

During the course of this tutorial I hope to do two things. The first is to present some of our current knowledge about the production, transmission, and perception of speech; and second is to give you some idea of how, on the basis of this knowledge, speech researchers try to extract information from speech automatically. While doing these things I will be inflicting a good deal of jargon on you. Speech researchers are unable to communicate with each other without this jargon, and you are going to have to learn some of it so they can also communicate with you.

Mechanics of speech production

The signal the speech scientist has to deal with is an acoustic waveform in the air, a sequence of sounds we produce in the region known as the vocal tract. An interesting fact about speech production is that all the organs used for speech evolved for other purposes. The lungs, which produce the air stream that powers the making of sound, are actually provided to keep you alive by bringing oxygen to your blood. Here is a picture of the lungs making sound.

(Slide 1 -- lungs)

The airstream passes up through the larynx, which is a valve you close when you eat, to keep food from dropping into the lungs. It is in the middle of your neck, and is larger in men than in women -- enough larger that it causes a protrusion in a man's neck called the Adam's apple. The larynx contains two horizontal, opposed curtains, or folds of flesh supported by cartilage. Here is a drawing of the larynx looking down from the top.

(Slide 2 -- larynx)

Muscles can open and close the opening between the folds; this opening is called the glottis. When the folds are closed, or pulled together, you can still force air up through the larynx. When you do, the folds are forced apart; then they are sucked together again by the air stream; they slap together; the air stream forces them apart again; they close again; etc. The action is that of what is called a relaxation oscillator.

This procedure is often described as vibration of the vocal cords. The implication is that there is a harp-like structure in your throat. I have done considerable research looking for the origin of this misconception, and I have never found it. The earliest speech literature speaks of cords, although it was known anatomically that there were none.

(Slide 3 -- glottal pulses)

Here is a typical time waveform of airflow through the glottis when the larynx is "closed". It is not at all sinusoidal, as would be expected from a vibrating string. It is made up of puffs or bursts of air, relatively short compared to to the cycle time. In males the mass and tensions are such that the puffs occur between 50 and 200 times a second, depending on muscular effort. In women the larynx is smaller and less muscular, and the rate is likely to be between 150 and 350 times a second. If you could listen to this signal it would sound like a buzz, and it is often called just that.

The slide also shows the fourier spectrum of this sequence of glottal pulses. The spectrum of a signal shows how much energy is present at each frequency. The abscissa here is frequency, and the ordinate is amplitude, expressed in decibels, or db. Decibels are a logarithmic scale, used when a variable has such a large range that it is best to express its behavior by showing the ratios, rather than the differences, between its large and small values. The number of db between two values is 10 times the logarithm of the ratio between those values. Numbers that are the same size are 0 db apart; if their ratio is 2, they are about 3 db apart; 30 db means a ratio of 1000; 50 db means a ratio of 100000. The rapid falloff with increasing frequency is an important feature of the buzz, and will be referred to often.

When speech sounds are produced like this, with the vocal folds held together so that a buzz comes out, they are call voiced, or vocalized. The pulses are called pitch pulses, or glottal pulses, and the frequency of the buzz is called pitch, or fundamental frequency, or the fundamental. (Purists reserve the word "pitch" to describe a psychological phenomenon, a feature of perception, not production.) The time from one of these pulses to the next is called the pitch period. The "pitch" pattern of a sound (or of a word or a sentence) is called the intonation.

The term "pitch period" is, unfortunately, used in two different senses. It can denote an amount of time -- the number of milliseconds between one pulse and the next -- or it can mean the event that begins at the leading edge of a certain pitch pulse and ends at the leading edge of the next pitch pulse. I try to use "pitch period" for a length of time, and "pitch epoch" for the event the starts at the onset of one glottal pulse and last for one pitch period.

There is another way of producing sound in speech. The stream of air passing up through the vocal tract can be confined by a constriction so that it breaks into turbulent flow, producing hiss, or frication. Speech sounds produced in this way are called fricatives. Here is a waveform of a fricative sound.

(Slide 4 -- Fricative)

Frication can occur in various places in the vocal tract, all the way from the larynx to the lips. Note that frication (hiss) and voicing (buzz) are not mutually exclusive.

The stream of air can also be stopped, again at any point in the vocal tract. Speech sounds that involve a stoppage of airflow are called, unimaginatively, stops.

Whatever the source of the sound, it is called the excitation, as it is thought of as exciting the vocal tract. The many organs that make up the vocal tract then modulate this excitation.

(Slide 5 -- x-section of vocal tract)

Here is a side view of the vocal tract. The parts that can be adjusted to modulate the sound range from the velum (a flap at the back that determines whether the nasal passage will be coupled into the vocal tract) to the teeth and lips.

The most important modulating organ is the tongue, originally provided for moving food back into the esophagus. It can be humped high, creating a narrow passage, or laid low, making a large passage, and the hump can be at the back of the mouth or at the front. Also the tip can be curled back, or retroflexed. The amount of jaw opening also has an effect on the speech sound. Here is a waveform of a typical voiced sound. Note that it is much more complicated than the glottal waveform. This is the effect of modulation by the vocal tract.

(Slide 6 -- voiced speech)

And here is speech that is both voiced and fricated.

(Slide 7 -- voiced fricative)

Here is a graphic representation of this source-modulation process.

(Slide 8 -- buzz-output)

The upper line shows the sequence of pulses, and what happens when they have passed through the vocal tract. The lower line is the frequency-domain version of this process. A pulse train has a spectrum consisting of lines at multiples of the fundamental, as shown at the left. The vocal tract passes different frequencies with different amounts of attenuation; the function that describes this process is called the transfer function, shown in the middle. The output signal has a spectrum that is the product of the first two.

Articulatory Phonetics and the Sounds of Speech

Phoneticians are concerned with describing and making taxonomies for speech sounds. The study of how the sounds of speech are produced is called articulatory phonetics. Sounds can be divided into crude categories, such as voiced or unvoiced, and into fine categories such as exactly where in the mouth frication takes place. Here is a fairly coarse categorization of sounds according to how they are produced.

(Slide 8a -- sound categories)

These categories can be lumped together in certain ways to produce certain important cruder categories. For example, if a sound is not stopped, it is called a continuant. A continuant with no frication is called a sonorant. If it has turbulence, whether or not it is voiced, it is called a fricative.

A vowel is a sonorant in which there is no obstruction in the vocal tract (unlike /L/, for example, in which air is forced to flow around the tongue). Vowels occupy a major fraction of the total time in speech, and an even larger fraction of the total energy; and every word has at least one vowel in it. The tongue is the major determiner of vowel quality.

(Slide 9 -- vowel trapezoid)

This shows tongue position for the vowels of English. Left is the front of the mouth, and up means high. Vowels at the left are produced by pushing the tongue hump forward, and vowels at the right by pulling the tongue back. Vowels at the top have a high tongue hump, and vowels at the bottom have a fairly flat tongue.

So far we have looked at sounds only in terms of how they are produced. Most of the work in phonetics has to do with dividing up sounds according to how they are perceived, and how they are used in the spoken language. There is an infinity of produceable sounds, but we seem to perceive them as a small number of classes. Phoneticians classify perceived sounds in a number of ways.

One is the phoneme. A phoneme can be thought of as a linguistic sound class. Two sounds A and B "belong" to different phonemes if there is a pair of words W1 and W2, identical except in one place, such that putting sound A in that place and putting sound B in that place make W1 and W2 have different meanings. For example, the words "hit" and "heat" differ only in the sound between the /H/ and the /T/, and they have different meanings. Therefore the sound you hear as /IH/ and the sound you hear as /EE/ must belong to different phonemic classes.

Phoneticians divide sounds up in this way for all languages. There is not universal agreement as to how to do the dividing. But by any method, most languages have about 45 phonemes. Here is a list of one version of the phonemes of English.

(Slide 10 -- phonemes)

The symbols in the left column are the classic ones, standard among phoneticians, but cannot all be produced on a typewriter. The second and third columns are symbols that can be typed, and were developed during the large ARPA-sponsored speech research project of the early 70's. The two-character set has stuck, and is becoming the phonetic symbol set of the computer age.

Note that phonemes are language-dependent. In English we don't care whether a vowel is nasalized or not -- you can say "ah" or you can say "anh", with the velum up or down, and the meaning is not changed. In French, the situation is different. The words "a" and "in" in French differ only in nasalization, and have different meanings; nasalization is a phonemic feature of the language. In some languages, for example Chinese, intonation -- the pitch pattern of a sound -- is phonemic.

Note also that sounds need not be produced similarly in order to be in the same phonemic class. There are two very different ways of producing the phoneme /L/. One uses the tip of the tongue, and the other, a rarer version, uses the back of the tongue. Both are recognized as /L/ in a word like "lift". Thus a phonemic class can be thought of as a linguistic behavior class.

Sounds this different within a phonemic class are rare. But within any phonemic class, there are many different sounds. Infinitely many. But they appear to fall into subclasses in a definable way. In English, the sound /P/ in "pin" has aspiration, a puff of air, after the stop, while the /P/ in "spin" does not have aspiration. Both are assigned to the phonemic class /P/. They are called allophones of the phoneme /P/. The many -- but not infinite -- classes that are formed by dividing phonemes up into allophones are called phones. Phones are the fundamental units of speech, as produced and as perceived, the fine structure. The study of what phones occur in a language, and how they are produced, is called phonology.

All the phones of known languages can be represented with a very large set of symbols known as the International Phonetic Alphabet, using diacritical marks, subscripts, etc. for fine shades of difference.

A curious fact about phonemes: some of them represent speech acts that are steady-state, for example /AA/, the vowel in "ma". Some represent sounds that change slowly, like the glide /Y/ in "you". And some are sounds that change suddenly, like the /P/ in "pin". The units that have proved to be useful linguistically are not at all homogeneous from the standpoint of production.

Acoustic Phonetics

An automatic speech recognizer must deal with the acoustic waveform generated (in the air) by the talker; this is the only signal available. It is obviously sufficient, since it is all the listener has, and he can understand the word. The study of what acoustic features are present in the speech waveform, and how they are used by listeners to identify phones or phonemes, is called Acoustic Phonetics. Much of the work in machine recognition is based on what acoustic phoneticians have discovered about the so-called acoustic cues, the features that humans use to identify speech sounds.

Acoustic phoneticians are very interested in how the ear analyzes sound, as it must hâve a bearing on how we perceive speech. Much theorizing in speech perception is done in terms of current models of what the ear is telling the brain. Various models of the ear have been proposed as a result of surgical observation and psycho-physical experiments; some characteristics are common to all models and accepted by all speech researchers.

1) The ear does a fine frequency analysis of incoming signals and also resolves signals in time very accurately. (Unlike analyzers built by acousticians and electrical engineers, which cannot do both simultaneously.)

2) Low frequencies are "resolved" better than high frequencies; in fact, resolution depends on the difference of frequencies up to about 1000 hz, and on the ratio above 1000 hz. Thus, the relationship of the "natural" scale to the frequency in herz is linear up to 1000 hz, and logarithmic thereafter. This natural scale is called the mel scale by some and the bark scale by others. Here is a graph of the mel scale plotted against the frequency in herz.

(Slide 10a -- mel scale)

One of the most important concepts in acoustic phonetics is the formant. Formants were discovered by a 19th century phonetician who had time on his hands (he was ill) and very acute hearing. He noticed, while listening to his own voiced sounds, that besides hearing the buzz of the glottal waveform he could hear several high-pitched notes. The pitch of these notes was consistent for a given vowel, and differed from vowel to vowel. The high-pitched signals came to be known as formants -- they seemed to characterize the vowel. It is as if the vowel were really a sum of high-frequency periodic signals.

(Slide 11 -- narrow-band spectrum of /AA/)

If we make a fourier spectrum, a frequency analysis, of a vowel, we can see these formants. Here is a spectrum of the vowel /AA/. There are lines in the spectrum at multiples of the pitch frequency. Riding on top of these lines you can see large lumps in the spectrum. These are the result of modulation of the glottal signal by the vocal tract, or ringing of the so-called resonances of the vocal tract. The vocal tract is allowing more energy to get through at the frequencies near these lumps that at other frequencies. It is these lumps, or formants that you can hear as notes if you have very good hearing.

(Slide 12 -- wide band spectrum of /AA/)

The lumps are more apparent if we make a wide-band spectrum. That is, we do a frequency analysis using a filter that is so wide that it cannot resolve the lines, but can only follow the general shape of the spectrum. Here is the wide-band spectrum of the vowel /AA/. You can see that it follows roughly the tops of the lines of the narrow-band spectrum

(Slide 13 -- spectrum of /IY/)

If we compare the spectrum of another vowel, the vowel /IY/, with that of /AA/, we see that they are different. The lowest lump in /IY/ is lower than that for /AA/, and the second lump is much higher. It turns out that vowels can be characterized by the positions of these lumps.

The lowest-frequency formant is traditionally called the first formant, abbreviated F1. Voiced sounds usually have from 3 to 5 formants within the first 5000hz (telephone bandwidth). F1 generally falls in the range 250-1100 hz, F2 in 1000-2200 hz, and F3 in 2000-3500 hz. The fundamental frequency, or pitch, is often abbreviated F0, even though it has nothing to do with formants.

Phoneticians have learned a great deal by studying displays such as these. But by far the most-used display is the so-called sonogram. (This is actually a proprietary name -- the generic name is sound spectrogram.) This is the picture you get if you do a lot of spectra closely spaced in time, and plot the amplitudes as a function of two variables, frequency and time. The common way to display this function is with time along the x-axis, frequency along the y-axis, and amplitude as blackness -- the higher the amplitude at a given frequency and time, the blacker the mark on the paper.

(Slide 15 -- narrow-band spectrogram)

Here is a sound spectrogram of speech analyzed with a narrow filter; this is traditionally called a narrow-band spectrogram. The lines that run roughly horizontally are the harmonics of the fundamental; they are called pitch bars. Where there is no voicing there are, of course, no pitch bars.

(Slide 16 -- wide-band spectrogram)

If we use a broad analyzing filter, the picture looks like this. The vertical striations correspond to individual pitch epochs; the spectrum late in the epoch is different from the spectrum early in the epoch, which makes for a stripy look. The pitch bars are no longer resolved, because the filter is broad, but the formants are now more obvious. For this reason, phoneticians traditionally use this representation to display and study speech. It is still easy to tell voiced speech from unvoiced -- the unvoiced portions have no striations.

The Importance of Formants

(Slide 17 -- Peterson-Barney)

The connection between formant position and perceived sound is central to acoustic phonetics. Most of the information is found in the first two formants; F3 and higher seem to be fairly constant over all sounds for a given talker.

A famous study of F1 and F2 as indicators of vowel identity was done by Peterson and Barney; it is illustrated in this picture. From spectrograms they measured formant positions for ten vowels spoken by a large number of people. The picture shows F1 and F2 for every one of these tokens. F1 is plotted in the x-direction, and F2 in the y-direction. The identity of the vowel is shown by the symbol plotted. The symbols cluster, and the regions where they fall are indicated by a surrounding balloon and a phonetic symbol to show what vowel (mainly) falls in that region. There is no doubt that vowel identity and formant frequency are related.

(Slide 18 -- vowel vs. F1-F2)

The relation between vowel identity and frequency of F1 and F2 is summed up in this diagram, which also shows what the vocal tract is doing. In this regard it is instructive to compare two earlier pictures. If we superimpose the vowel trapezoid, which shows tongue position, on the Peterson-Barney plot, which shows F1-F2, we see that they are related. F2 is correlated with frontness, and F1 is anti-correlated with tongue height.

(Slide 19 -- stop and glide formants)

You shouldn't think that only vowels are characterized by their formants. For example, these highly stylized spectrograms show what formants look like for stops and glides. All sounds pictured end with the phoneme /EH/. The upper left is the nonsense word /BEH/. The lower left is /GEH/. The /EH/ part is the same, but the beginning has a so-called formant transition that is characteristic of the stop that precedes the vowel. The second formant in /BEH/ starts low, and the second formant in /GEH/ starts high. The 4th from the left in the top row is /WEH/. It differs from /BEH/ in that the formant transitions are slower. Thus stops and glides can both be characterized by the behavior of the formants of the nearby vowels.

Spectral Characterization of Speech Sounds

Since the 1930s speech researchers have been trying to characterize and categorize speech sounds automatically, and reliably, using a small number of parameters. There have been two main motivations for this work: one is the need to transmit speech economically, which has led to the development of vocoders, speech compression devices in wide use today; and the other is the desire to recognize words automatically. These two have developed together, and many researchers have made contributions to both.

The fourier spectrum, the version with broad peaks and no pitch bars, has played the principal part in both speech recognition and speech compression. In English and other non-tonal languages the pitch bars carry no linguistic information, so the shape of the broad spectrum, especially the positions of the main peaks, is a good clue to the identity of a continuant speech sound. And the spectrum changes fairly slowly, because the articulators in the vocal tract are sluggish; you don't have to derive the spectrum very often to capture changes in the speech sound -- 100 times a second is plenty.

(You might think that voicing would not be captured by the broad spectrum. The narrow-band spectrum does capture this information, since it has lines, or pitch bars, for voiced speech and none for unvoiced speech. But the broad-band spectrum also contains voicing information, in its general shape. In voiced speech, the spectrum of the excitation, the glottal pulses, falls off rapidly with frequency; therefore the output spectrum, which is the product of the glottal spectrum and the transfer function of the vocal tract, also falls off rapidly. Unvoiced speech has for its excitation something like white noise, which has a flat spectrum. Hence the broad spectrum of an unvoiced sound is much flatter than the spectrum of an unvoiced sound.)

We have already seen, in the Peterson-Barney diagram, that the formant positions for a given sound can vary considerably from person to person; they are not even consistent for a single individual (although much more constrained than for the whole population). This unfortunate variability will be dealt with in more detail later. Focusing attention for the moment on a single individual, whose formants will be tightly localized, it would seem a simple matter to find the formants and use their position to characterize the sound. Many a researcher has blunted his spear trying to do just that; the lumps in the spectrum have proved remarkably hard to find automatically. And if, for example, you miss the first formant entirely, and identify the second as the first, you will almost surely mis-identify the sound. An advantage of using the whole spectrum to characterize the sound, rather than just the formant positions, is that it fails gracefully.

Practically every speech recognition system, then, transforms incoming speech into a sequence of spectra, and discards all other information that may have been present in the speech waveform. The vocal tract is regarded as a machine that churns out spectra, one spectrum every 1/100 of a second. On the question of how actually to represent that spectrum as a sequence of numbers, there is no such unanimity. To begin with, some systems are based on a linear spectrum: they divide up the spectrum into k equal intervals, find the amount of energy in each interval, and represent the spectrum as a sequence of those k numbers. Others are based on the mel scale mentioned above (in describing the action of the ear); the spectrum is divided up linearly in the range 0-1000 hz, and logarithmically from there on.

(A parenthetical note: while most recognition systems are based on the spectrum, since it seems to characterize speech sounds well, some systems are based on more complicated functions of the incoming signal, functions that transfom the acoustic signal as we think the ear does before sending it on to the brain. The idea is that the ear-brain combination is a splendid recognizer, so if the ear does it, it must be a good thing to do. There is evidence that this is a good direction for recognition research to go in, but opinion is still divided. To keep it simple, I will talk about systems as if they were all spectrum-based. In fact, except that speech is represented differently, the ear-model systems are just like the spectrum-based systems.)

The time needed to calculate the spectrum is of concern to system builders. Fortunately, just when computer speech recognition research began in earnest, a new method of computing the fourier spectrum was invented, known as the Fast Fourier Transform, or FFT. The FFT is most easily and rapidly computed if its size is a power of 2. It produces a spectrum that covers the frequency range from 0 to half the rate at which the speech was sampled -- 5000 hz, if sampling is done 10000 times a second. Thus many recognition procedures are based on a spectrum that divides the frequency range 0-5000 into 64, 128 or 256 intervals.

Linear Predictive Coding

A development of the early 70's was the application to speech of Linear Predictive Coding, or LPC. This process, in use at that time by seismologists, makes use of the notion that if we make certain acoustic assumptions about the machine that is producing the speech sounds, the spectrum is very easy to calculate -- or more properly, to approximate. LPC is based on the observation that the vocal tract is a tube, closed (most of the time) at the glottal end and open at the lips end. Such a tube can be approximated by a set of coaxial cylinders, all the same length, but of varying cross-sectional area, like this.

(Slide 20 -- Acoustic tube)

Nobody thinks the vocal tract actually looks like this, but acousticians assure us that the sound coming out of a vocal tract can be duplicated by introducing a buzz or hiss, as appropriate, into such a tube. For reasons not detailed here, there should be 10 sections, each 1.7 cm long; specifying the area of each of these 10 sections completely specifies the quality of the output sound -- that is, every sound is characterized by just 10 numbers.

The LPC process can then be thought of as follows: Form the fourier spectrum of the current centisecond of speech. Now generate a sound with the acoustic tube, and form its fourier spectrum, and compare that spectrum with the spectrum of the speech sound. Do this again and again, varying the areas of the tube sections, until the spectrum of the otput of the tube is maximally like the spectrum of the speech sound. Characterize the speech sound by the 10 cross-sectional areas of this "best" tube. (The LPC algorithm is a clever method for carrying out in a very short time what seems like a wander through an infinite 10-dimensional space.)

The spectrum of the output of this "best" tube is called the LPC spectrum. It is in many cases remarkably like the spectrum of the speech sound. Here is a comparison of the LPC spectrum of the sound /AA/ with the fourier spectrum of that sound. The LPC spectrum has captured the essential shape, especially the formants; and remember that this spectrum is specified by just 10 numbers, the areas of the 10 sections of the tube.

(Slide 21 -- LPC spectrum of /AA/)

Here are the fourier spectrum and LPC spectrum of the vowel /EE/.

(Slide 22 -- LPC spectrum of /EE/)

Some systems based on LPC use the 10 areas to characterize the current speech sound, others use the LPC spectrum, still others use certain functions of the 10 areas such as the ratios between successive areas, logarithms of those ratios, or another function called the reflection coefficients.

The Cepstrum

The spectrum is used because it seems to retain the linguistic information and discard the non-linguistic information. In many systems the recognition algorithm operates not on the original spectrum (whether mel or linear), but on some function of the spectrum that is thought to do an even better job of emphasizing information useful in recognition and discarding the "noise".

One such function is called the cepstrum (a coined word). The cepstrum is the spectrum of the spectrum (actually the spectrum of the logarithm of the spectrum). The reasoning behind the use of the cepstrum is the following. A spectrum is a frequency analysis -- it is a function that tells you what frequencies are present in a signal. The spectrum of speech is itself a signal, a signal whose shape corresponds to the sound, or phone, being produced. A frequency analysis of that signal, of the spectrum, might capture the salient features of that shape, and thus characterize the phone. The more "coefficients" one uses in forming the cepstrum, the finer the frequency analysis (of the speech spectrum) that will result. Cepstrum-based recognition systems generally use about 15 cepstral coefficients to represent the speech spectrum. The cepstrum is very much in vogue at present.

Recognition Algorithms

The stage is now set. Speech has been transformed into a sequence of vectors, one each centisecond, vectors which are thought to be similar for similar sounds, and different for different sounds. For concreteness, let us assume that the vectors are 15 long. A word half a second long will be represented by about 50 of these 15-long vectors. How do we now decide what word it is? We need a recognition algorithm.

There are two schools of thought about how a recognition algorithm should be created. One school says, the way to proceed from here is to study the characteristics of these sequences of vectors, to see what common behavior they have -- what is invariant -- over all tokens of a particular sound. On the basis of such study rules can then be developed, involving features (such as formant behavior) of the vectors in the sequence, rules that hold whenever that sound is spoken. (This is very much what acoustic phoneticians have been trying to do for many years, largely through the study of sound spectrograms -- the spectrogram is, of course, a visual display of a sequence of spectral vectors.) The recognition system can then test an incoming utterance, to see if it is some particular word, by determining (according to the rules) what the incoming sounds are, and then looking at the pronunciation of the word to see if the sounds are appropriate to that word. Systems that operate in this way are called acoustic-phonetic, or feature-based systems.

(Slide 23 -- schools of thought)

The other school says no, we are not clever enough to develop rules that will tell us what sounds are being produced; we must let the speech speak for itself, by "training" the recognizer automatically. There are three current methods (not entirely mutually exclusive) for doing this:
Template matching;
Statistical modeling;
Neural nets (which I will not discuss).

For a long time the speech community has been struggling to produce a successful feature-based system, and some day they will. At present, however, the other approaches, particularly, the statistical, are way out in front. We are just too ignorant to extract and codify the important clues to sound or word identity. In the rest of this tutorial, then, I will not talk about feature-based systems.

The template-matching system

The simplest sytem is one in which there is a vocabulary of some number of words (50 is a common number for this kind of system) and the system is "trained" by having a talker, the person who will be using the system, speak each of the 50 words. The system stores each word as a sequence of vectors. The stored sequence is called a template. Recognition is done by converting an incoming word to a sequence of vectors, and comparing that sequence to each of the templates in turn. Whoever matches best against the new sequence is the winner.

Implicit in this description is the ability to find the beginnings and endings of incoming words. In running speech, this is a very hard thing to do. Various systems get around this difficulty in various ways -- one way is to have the user push a button before and after each word. Another is to have the user pause between words, and use the silences to demarcate the words. Both these allow the system to be what is called an isolated word recognizer -- the problem of breaking up a sentence into words does not have to be faced. Later I will discuss what template-based systems do when the input is so-called connected speech.

The system must have a quantitative way of comparing two sequences, often called a scoring algorithm. Let us assume that the sequences are exactly the same length. The algorithm then has two parts: first, we must say how to score an element (vector) in one sequence against its opposite number in the other sequence. This is commonly done by taking the cross-product (or equivalently, the sum of squared differences). Second, given scores for individual elements of the sequence, we must compute a score for the entire sequence; a common way is just to add the cross-products.

If the sequences are not the same length, we cannot compare them element by element. Fortunately there is an algorithm, called Dynamic Time Warping (DTW) that allows us to compare sequences of different lengths. It does this by stretching and/or compressing the sequences so that they are the same length, and furthermore the places where they are the most similar are lined up with each other. DTW has been astonishingly successful in eliminating the timing problem in template matching word recognition.

A more sophisticated version of DTW, called a level-building algorithm, also allows template matching to be used even on connected speech, where the beginnings and ends of words are impossible to find. Effectively, DTW matches every sequence of words from the vocabulary with the incoming utterance, trying every possible time-warp of each word, and chooses the sequence of words that scores the best.

This is a good place to point out that working speech recognition systems do more than just analyze the acoustic signal to determine what words have been spoken. Every system is designed to work in some task domain, and every such domain has restrictions that limit, or at least change the probability of, the words that can appear at any particular place in a sentence. These restrictions include grammar, semantics, subject matter, and what is called pragmatics -- the particular situation or mode the talker is in (such as the what he said in his previous sentence). All these sources of knowledge are built into the system to the greatest possible degree, and have a tremendous effect on the ability of the system to recognize words.

The system as so far described is still suitable for only one talker -- it was trained by only one talker, and one talker's templates may not be suitable for recognizing another talker's words: as the Peterson-Barney diagram shows, the two talkers may have their formants in different places even when they are making exactly the same sound. Template-based systems attack this problem in two ways: one is to gather several (4 to 6) templates of each word from each of several talkers, and use them all during recognition. Another is to gather such templates, and then make an average template, or a few average templates; perhaps one for men and one for women. These are the current state-of-the-art methods of coping with the spectral variability problem in template-based speech recognition.

Phone and Phoneme Templates

A variant, or refinement, of the template-matching system is a system in which phones or phonemes, rather than words, are the units that are templated. An obvious attraction of this idea is that no matter what the vocabulary, the phonemes are fixed -- there is only a small finite number of them to "learn". If we had a template for every phoneme, and could recognize phonemes as they occurred, we would be just where the acoustic phoneticians wanted to be -- we could search the vocabulary to see what word (when pronounced) best matches the incoming string of phonemes. With a reasonable size of vocabulary, we could even tolerate a fair amount of garbling, or at least uncertainty of identification. But even for a single speaker, this approach has not been fruitful. Phones and phonemes undergo radical changes in pronunciation due to context (coarticulation effects), speed, loudness, and other effects. Many of the rules governing these changes ("phonological rules") are known, but even so, units this small are currently too hard to identify reliably.

It has been suggested (often) that a template-matching system based on the syllable would be a proper compromise between words and phones. Again, no matter what the vocabulary, the number of syllables is finite -- but very large (several thousand) in English. Both the difficulty of distinguishing among so many items, and the fact that a lot of memory is needed to store so many templates, have discouraged Western researchers from pursuing this approach. It is more suitable to Japanese, however, which has about 100 syllables -- and in fact is being tried in Japan.

Statistical Modeling -- Today's Leader

The philosophy behind the statistical model system is the following. We are too ignorant to specify rules for determining, from a spectrum or sequence of spectra, what sound is generated. Word templates are impractical because the variability in pronunciation is so great -- it would take too many of them to cover all the possibilities. Phoneme templates have not worked. But somehow the brain learns to identify a sound after exposure to many examples of that sound (and other sounds). Therefore the sequence of spectra representing that sound has statistical properties that serve to distinguish it from other sounds. We will try to get at the statistical properties of the sounds by imagining that they were produced by a very simple machine; we will assume a form for that machine, and then estimate its parameters by statistical estimation from a large amount of actual speech.

The currently most popular (and successful) statistical model is the Hidden Markov Model, or HMM. By way of introduction to statistical modeling, and before describing a HMM, I will describe a simpler statistical model. To fix ideas, let us assume that English has 99 phones, labeled P1, P2, ... , P99. And assume that we know how to segment speech into individual phones.

Imagine that there is a large urn full of phones, spoken by lots of people. Different phones are present in different numbers -- lots of some, not so many of others. Our model of speech production will be: Speech is generated by drawing phones (independently) out of the urn and concatenating them.

We want to build a recognizer based on this model. We must first find out what each of the phones looks like in a statistical sense. To do this, we collect a lot of speech under controlled circumstances; since we know exactly what was said, we can divide it up into known phones. We then bring together all examples of a given phone (reduced to sequences of 15-long vectors); if the average phone is 10 vectors long, and there are 200 samples of a given phone, there are 2000 vectors all belonging to that phone. We compute the mean and variance of those 2000 vectors -- that mean and variance will be our statistical model for that phone. If we do this for all 99 phones, we will have 99 means and variances -- a statistical description of the imaginary machine that produces speech. We have trained the model.

Now we can use this model to identify an incoming word. We segment the word into phones -- but we don't yet know what phones they are. A typical phone is perhaps 10 vectors long. There is a standard statistical technique for then calculating the probability that those 10 vectors all belong to P1, to P2, etc. That is, we can attach 99 probabilities to the incoming phone, one for each phone in the model. Hopefully, one of them is very large and the rest are very small. At any rate, our choice for the incoming phone is the model phone with the largest probability.

When the word is all in, the recognizer has produced a sequence of phone labels That sequence can then be compared with the words in the vocabulary to see what word the recognizer thinks is most likely.

Models like this have been used with some success. Such a model is very like the model where there is a template, or several templates, for each phone, but here the "template" is statistical. A real problem with such a model is that we must be able to segment the training set into phones, in order to collect together all examples of each phone; and then we must be able to segment the unknown, incoming word into phones. The building of the training set is painful, and the segmentation of unknown words is very hard, and is a major source of error in such a recognition system.

The Hidden Markov Model, as used in speech recognition, overcomes these difficulties. We imagine quite a different machine as the speech generator.

The machine has two parts. The first part determines what state the machine is in. We are free to imagine how many "states" the machine has -- to fix ideas, let us say there are 5 of them, labeled S1 to S5. This part of the machine has a centisecond clock and a probabilistic change rule; the machine starts in state S1, and every centisecond it consults the rule to see if it is time for a change. If so, it changes to state S2. It continues this process until it has gone through S3, S4, and S5. When it leaves S5, it stops.

A common variation of this machine allows the machine to skip a state: when leaving S3, for example it may go either to S4 (with a certain transition probability P34) or to S5 (with probability P35). P34 and P35 must of course add to 1. Such a machine can be diagrammed like this:

(Slide 24 -- finite state machine)

The circles are the states. The arrows show what change of state can occur at each tick of the clock. The re-entrant arrows show what happens when the change rule does not call for a change -- the machine stays in the same state. The other arrows show what happens when the state changes -- the machine can advance either one or two states, and the labels on the advance arrows are the probabilities of the two advances.

So far no "speech" has been generated. That happens in the second part of the machine, which works as follows. There are 5 urns of sounds, labeled U1 to U5, one urn for each state of the first part of the machine. (You can think of the contents of the urns as phones, although they don't have to be phones.) Since we represent sounds as 15-dimensional vectors, what is actually in urn U3, for example, is a collection of vectors with a certain mean M3 and variance V3. To make the model mathematically tractable, we assume that the vectors are Normally distributed.

The second part of the machine operates off the same centisecond clock as the first part. At each tick, it selects at random a sound (vector) from the urn corresponding to the state of the first part of the machine. If the first part is in state S2, the second part makes a random drawing from urn U2. The vector that is drawn is defined to be the "speech" that is put out by the machine at the current clock tick. A "word" then is a succession of vectors, first some from U1, then some from U2, U3, U4, and U5 (except that an urn may be skipped).

The "Hidden" in this model refers to the fact that the state of the first part of the machine, and therefore the identity of the urn that is drawn from by the second part, is hidden from us; we see only the vectors that are drawn from whatever urn it was. The "Markov" is a mathematical term having to do with how successor states are (probabilistically) chosen by the first part of the machine.

Now what is the point of such a model? It is this. Suppose we assign some probabilites to the 5 transitions of part one (including the probability of no transition at all), and to the 5 means and variances of part two. And suppose we collect a large number of tokens of some word, which now do not need to be segmented as they were in the phone model described earlier. In the jargon of HMM, this collection of tokens is called the Observations. Then there is a statistical technique for calculating the probability, given the Observations, that the parameters we have assigned to the machine are correct. Further, there is an algorithm that allows us to do a wonderful thing. Based on the Observations, and on the current parameters of the machine, one application of the algorithm will produce a new set of parameters that is guaranteed to be more likely than the set we started with. If we apply this algorithm repeatedly we will "climb" to a set of parameters for our machine that is maximally likely to be correct, given the Observations. This set of parameters -- transition probabilities and 5 means and variances -- is our statistical model for the word we collected.

Note that no segmentation, and no identification of phones, was necessary. This process is repeated for each word in the vocabulary -- all we need is many tokens of each word to properly train the model. The set of all word-models, one for each word in the vocabulary, is our statistical model of speech.

Recognition proceeds just as it did for the previous statistical model. For each word-model we calculate the probability that the incoming word was produced by that model; if there are 50 words in the vocabulary, we get 50 probability estimates, one for each word-model. Hopefully the one with the highest probability is the word that was actually spoken.

An advantage of the HMM, as has been noted, is that we need not do any segmentation of the training collection, or of the incoming word. There are disadvantages, too. For one thing, we must assume that every word, of whatever length, has the same number of states. For another, we can never know what the states really mean; often, if you look at the implicit segmentation (change of state) you will recognize linguistic classes, but not always. This is not a mathematical disadvantage, but it leaves the user somewhat unsatisfied.

The Form of a Speech Recognition System

You will be hearing about many recognition systems, all different but all having the same general form. You could probably draw this diagram yourself by now, but here is the generic Speech Recognition System.

(Slide 25 -- Speech Recognition System)

Sources of Variability

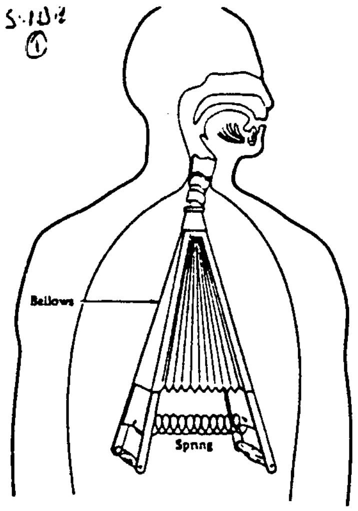
You cannot be allowed to proceed from this tutorial to the rest of this meeting thinking that these systems have an easy time of it. If every phone, or other sound unit, gave rise to just one spectrum, speech recognition would be simple and successful. Unfortunately there is tremendous variability in how spectra can look for a given sound, and in how a sequence of spectra can look for a given word. Here are some of the sources of variability that plague builders of speech recognition systems:

(Slide 26 -- Sources of variability)

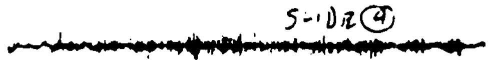
Size, sex, and age of the talker -- men, women, children and the aged have very different spectra for a given vowel, and even within one of these groups there is considerable variation;
Dialect -- can have a gross effect on certain sounds;
Loudness, emotion, vocal effort -- all affect formant size and position;
Coarticulation -- phoneme pronunciation depends on what its neighbors are;
Speech rate, loudness, health -- affect pronunciation and sound quality;
Channel -- the transmission path between talker and listener (or recognition device) -- often changes the gross shape or tilt of the spectrum (but usually doesn't affect formant positions greatly);
Noise -- masks the small ripples in the spectrum, and even some big ones.

Human listeners do just fine in spite of all this variability, but we are a long way from understanding how they do it, and a long way from building speech recognizers that can really cope with it.

SLIDE ①



SLIDE ④



SLIDE ⑤

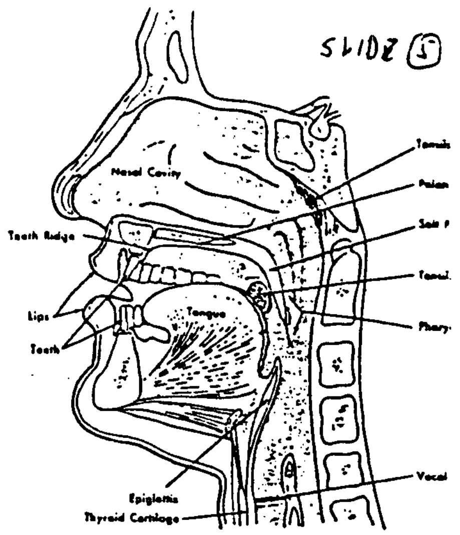
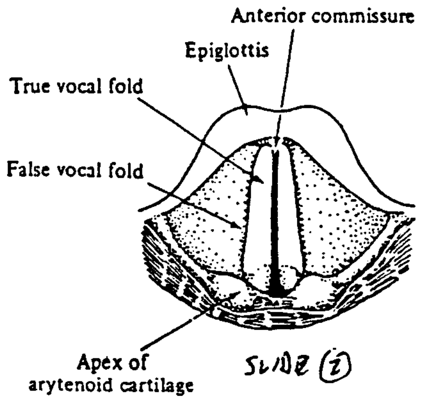
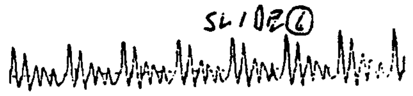


Fig. 4.7 Cross-sectional view of the vocal tract.



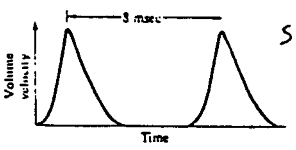
SLIDE ②

SLIDE ⑥



SLIDE ⑦

tral char-
re. [*After*
use. "An
oductions
J. Speech
302-20.]



SLIDE ③

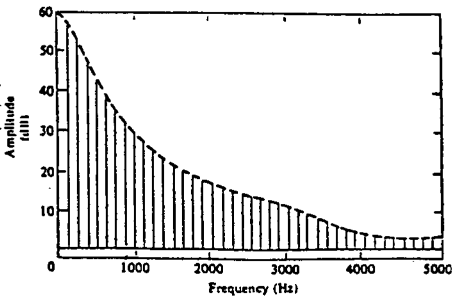
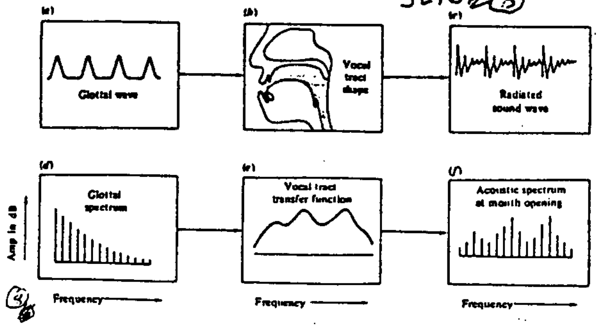
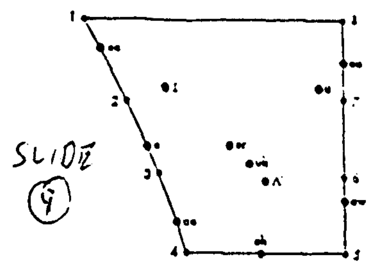


Figure 7-8 Shown is a schematic drawing of the physiological and acoustical characteristics of speech sound production. The triangular volume velocity pulses (*a*) are passed through the vocal tract (*b*) and exit the mouth opening as a complex sound wave (*c*). If this process is viewed from a spectral perspective, we can see that the glottal waveform consists of a harmonic spectrum of the type shown in (*d*). The vocal tract resonating characteristics (*e*) modify the glottal spectrum so that the radiated acoustic wave is dependent upon both the glottal spectrum and the vocal tract transfer function (*f*). [*After G. Fant,* Acoustic Theory of Speech Production (*The Hague: Mouton & Co., Publishers, 1970*)*, p. 19. Reprinted by permission of the author and the publisher.*]

SLIDE ⑤



④



THE PURE VOWELS				THE DIPHTHONGS	
ee	beet	ah	father	ou	tone
I	bit	aw	call	ei	baie
e	bead	U	put	ai	might
ae	bad	oo	cool	au	shout
uh	the	A	ton	oi	toil
9 ɝ	er	bird			

Tongue positions for English vowels (the tongue positions in

SLIDE 10 COMPUTER PHONETIC REPRESENTATIONS

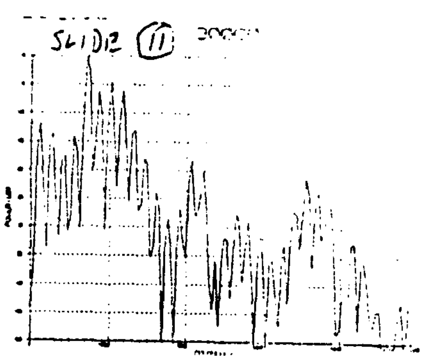
Phoneme	Computer Representation		Example	Phoneme	Computer Representation		Example
	1-Character	2-Characters			1-Character	2-Characters	
i	i	IY	beat	p	p	P	pet
I	I	IH	bit	t	t	T	ten
e	e	EY	bait	k	k	K	kit
E	E	EH	bet	b	b	B	bet
æ	@	AE	bat	d	d	D	debt
a	a	AA	bomb	g	g	G	get
ʌ	A	AH	but	h	h	HH	hat
ɔ	c	AO	bought	f	f	F	fat
o	o	OW	boat	θ	T	TH	thing
U	U	UH	book	s	s	S	sat
u	u	UW	boot	š ʃ	S	SH	shut
ə	x	AX	about	v	v	V	vat
ɝ	R	ER	bird	ð	D	DH	that
au	W	AW	down	z	z	Z	zoo
aI	Y	AY	buy	ž → ʒ	Z	ZH	azure
ɔI	O	OY	boy	č → tʃ	C	CH	church
y	y	Y	you	ǰ → dʒ	J	JH	judge
w	w	W	wit		H	WH	which
r	r	R	rent	syl l	L	EL	battle
l	l	L	let	syl m	M	EM	bottom
m	m	M	met	syl n	N	EN	button
n	n	N	net	flapped t	F	DX	batter
ŋ	G	NX	sing	glottal stop	Q	Q	

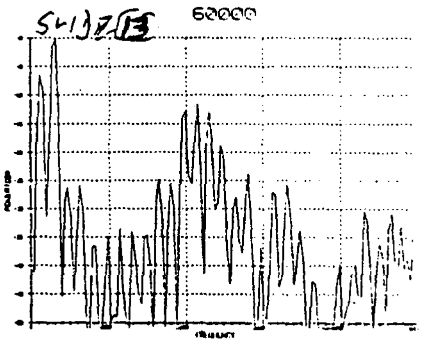
AUXILIARY SYMBOLS (1- AND 2-CHARACTER CODES ARE IDENTICAL)

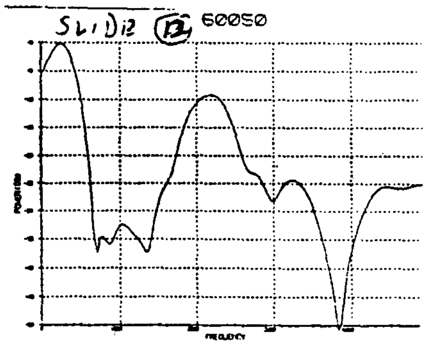
Symbol	Meaning	Symbol	Meaning
-	Silence	?	Interrogative terminal juncture
+	Morpheme boundary	,	Non-terminal juncture (comma)
/	Word boundary	( )	Non-speech escape
.	Declarative terminal juncture	" "	Phonetic or allophonic escape

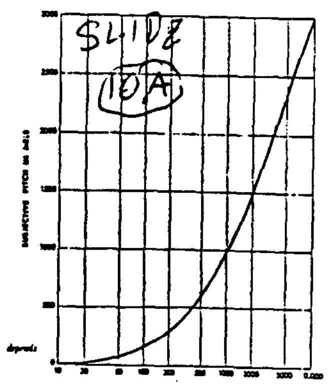
STRESS REPRESENTATIONS (IMMEDIATELY FOLLOW THE VOWEL, IF DESIRED)

Value	Stress Assignment	Value	Stress Assignment
0	No stress	3	Tertiary stress
1	Primary stress	:	(Etc.)
2	Secondary Stress		









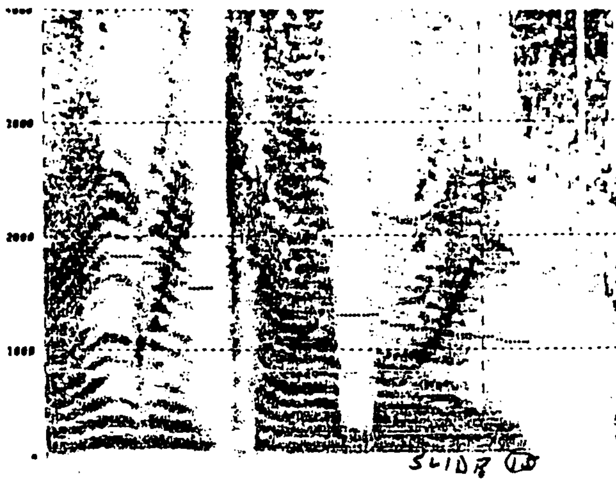
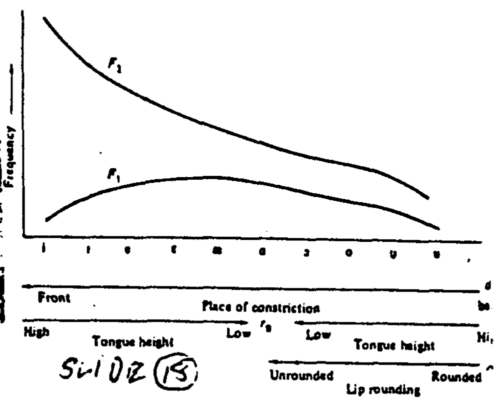
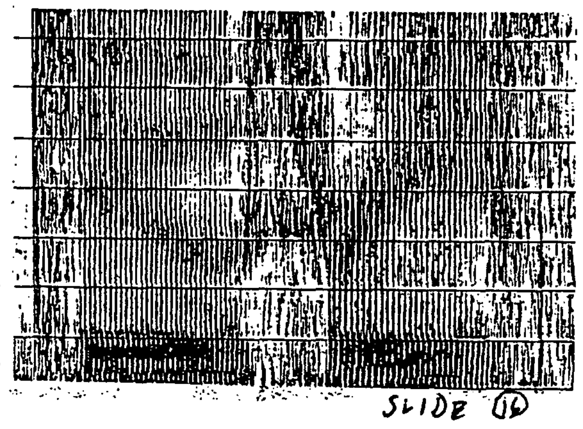
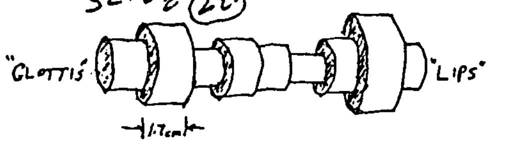
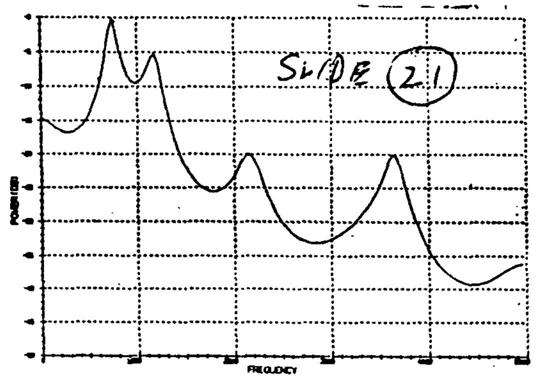


Figure 7-18 Formant 1 and formant 2 resonances for selected vowel sounds. The arrows at the bottom of the figure depict the influences place of constriction, degree of constriction, and the degree of lip rounding on vowel resonance patterns.









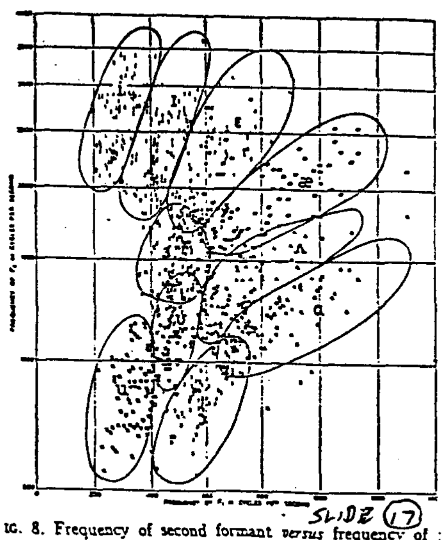


FIG. 8. Frequency of second formant *versus* frequency of formant for ten vowels by 76 speakers.

17

130 LIBERMAN, DELATTRE, GERSTMAN, AND COOPER

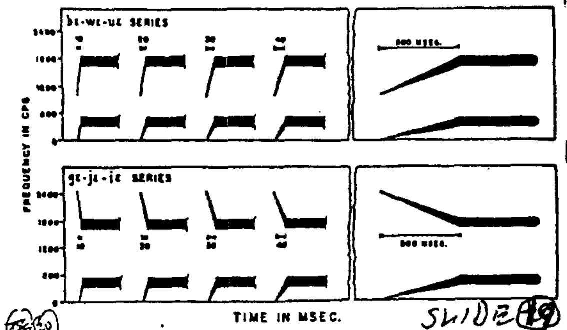
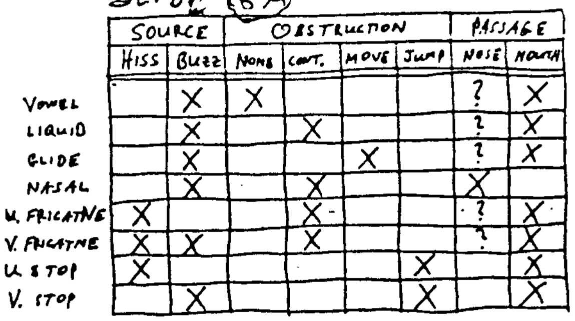


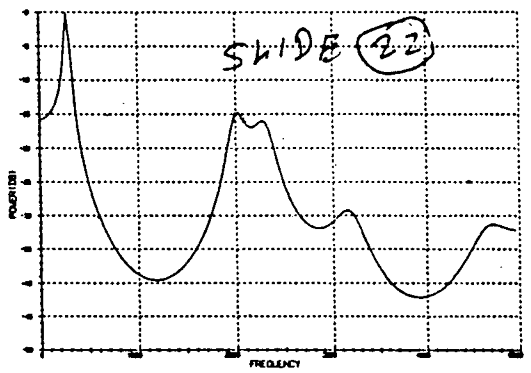
FIG. 2. Illustrations of the spectrographic patterns used to produce the stimuli of Exp. I. The

SLIDE 8A



	SOURCE		OBSTRUCTION				PASSAGE	
	HISS	BUZZ	NONE	CONT.	MOVE	JUMP	NOSE	MOUTH
VOWEL		X	X				?	X
LIQUID		X		X			?	X
GLIDE		X			X		?	X
NASAL		X		X			X	
U. FRICATIVE	X			X			?	X
V. FRICATIVE	X	X		X			?	X
U. STOP	X					X		X
V. STOP		X				X		X

SLIDE 22



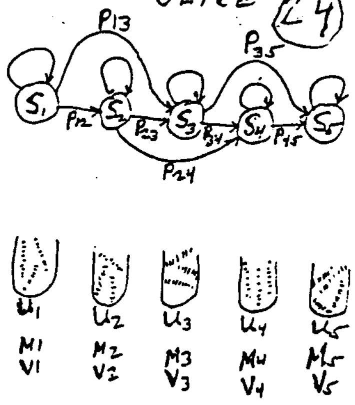
SLIDE 23

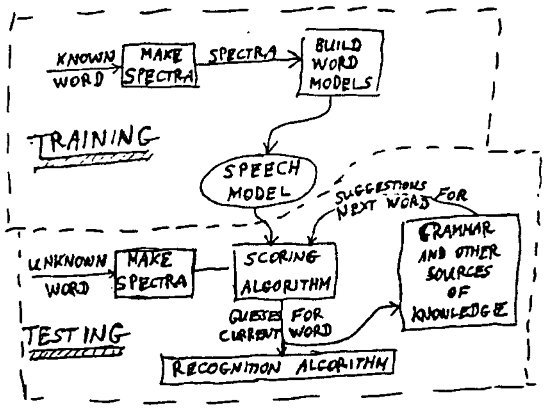
PHILOSOPHIES

FEATURE EXTRACTION

TEMPLATE MATCHING
STATISTICAL MODELING
NEURAL NETS

SLIDE 24





SLIDE 25

SOURCES OF VARIABILITY

SIZE, SEX, AGE
DIALECT
LOUDNESS, SPEECH RATE
HEALTH, EMOTION
COARTICULATION
CHANNEL
NOISE

SLIDE 26