Deterministic Parsing of Multiply-Structured Inputs

Georg Dorffner and Stan Kwasny
Computer Science Department

Robert F. Port
Department of Linguistics
Indiana University
Bloomington, IN 47405

Technical Report No. 190

Deterministic Parsing of Multiply-Structured Inputs

by
Georg Dorffner and Stan Kwasny
Computer Science Department
and
Robert F. Port
Department of Linguistics

April, 1986

Deterministic Parsing of Multiply-Structured Inputs*

Georg Dorffner
Computer Science Department

Stan C. Kwasny
Computer Science Department

Robert F. Port
Department of Linguistics

ABSTRACT

Many problems in parsing are caused by improper "conceptual factoring" in the parser and premature commitment to specific categorical labels. This paper shows how both of these problems can be minimized. First, a technique for coordinating language processing across multiple representational perspectives is presented, which has several distinct advantages over other approaches. Next, our method of labeling segmented acoustic speech signals is shown to provide a framework for speech processing which avoids hasty classification of the segments into overly distinct categories. Our ideas are illustrated through examples implemented in a deterministic, left-to-right model of parsing, which develops coordinated structures simultaneously at multiple conceptual levels. Justification for the approach, both in conventional Natural Language settings as well as in speech recognition, is presented.

1. Introduction

Many problems in parsing, we believe, are caused by improper factoring in the parser and premature commitment to specific categorical labels.

Proper factoring produces a parser which is efficient in reducing both the redundant representation of specific facts and the number of distinct hypotheses that must be pursued. Woods (1980) considers this notion extensively in relation to Cascaded ATN grammars. We show how advantages similar to these can be gained within a deterministic framework.

Commitment to a specific classification of an input element, based on partially complete information, often necessitates backtracking. For example, if a parsing algorithm is faced with an ambiguous input, say a choice between

* This work was supported in part by the National Science Foundation under grant No. DCR-8518725.

classifying it as a noun or as a verb, many parsers make a guess, but if the guess is incorrect, the parser must eventually retreat back to that point to try another guess. The parser must also keep track of which classification hypotheses have been tested.

Our method of classification, which only attaches labels to the input having the highest degree of reliability, minimizes the amount of bad guessing. Subsequent parsing may refine the labelling, but in a strictly monotonic, additive way. The notion of monotonic structure building is discussed extensively later.

Our approach in this problem is guided by the *principle of least commitment* (Marr, 1982), which suggests that premature conclusions about the input be avoided when all relevant facts are not known. An over-zealous parser, based on the idea that every input must be assigned an initial distinguishing classification, is incorrectly conceived, we believe. Ways of coping with these problems in a deterministic framework are discussed in this paper.

Determinism eliminates the need for backtracking during parsing. This is accomplished only by insuring that correct decisions about the input are made while structures are being built, left-to-right. Since such decision-making cannot occur accurately within a strictly local context, a buffer of constituent items must be maintained for the purpose of looking ahead while parsing. The "wait-and-see" (WASP) parser of Mitch Marcus (1980) serves as the basis for our implementation of these ideas.

A variant of a WASP parser has been developed by us which preserves all of the important properties of the WASP approach, but adds the capability of simultaneously pursuing multiple levels of description during the parsing process. In this way, a single input stream can be processed simultaneously from the perspective of syntax, semantics, pragmatics, phonetics, prosodics, or whatever other perspectives are useful. This approach views the input as implicitly structured. The task of the parser is simply to extract this structure, which may be multifaceted. The constituent buffer serves as a "window" into the input at a level appropriate for the current processing activities. "Attention-shifting" rules are used to switch among structural levels while still tracking multiple conceptual levels. Individual rules can operate within these multiple perspectives, causing attention-shifting perhaps multiple times. Thus, the factoring principles of Woods are re-cast within a deterministic model.

In speech processing, a sequential grammar, which captures the facets of the problem in different collections of rules, lacks the coordination necessary to properly relate structures. Furthermore, although non-left-to-right methods seem essential in processing speech, deterministic analysis of multiple levels provides another option.

This paper discusses first the notion of multilevel parsing, including an example which illustrates parsing at both syntactic and semantic levels; next, it discusses a proposal for delaying commitment to a specific input labelling within the structures being built, with an example of speech processing at multiple levels; finally, it concludes with some brief speculations about where this work is leading.

2. Multilevel Parsing

In this section we introduce a concept we call "multilevel parsing" which we developed to perform advanced analysis of natural language and speech utterances.

Parsing is the process by which the input string of symbols is assigned structure, as defined by a grammar. In doing this, the string is also determined to be well-formed with respect to the grammar. In the theory of formal language, the input string of terminal symbols can be reduced to nonterminals and, if well-formed, it can be reduced to exactly one nonterminal. These terminal symbols have no internal structure or "inner life." That is, they exist without any features that go beyond their mere existence in the grammar. Therefore, by definition, parsing is bound to the act of detecting the one structure that is defined by the grammar rules. By saying *one* structure, we mean one structural framework, knowing that ambiguous sentences have several individual structures within the same frame work.

In spite of striking similarities between formal and natural languages, our goals in handling the latter are somewhat different. The terminal symbols here (i.e., the words or other units) contribute much more to the sentence than just their position. Although the structure defined by a grammar plays an important and justified role, there exists more than one pattern that can be assigned to the sentence. One of these patterns is the syntactic structure and another is the semantic structure. It is formed in a different way than the syntax, although interdependently. We will show later that the syntactic and the semantic pattern need not be the only ones considered during sentence parsing.

Therefore, in contrast to many common techniques, parsing in general should not be considered as the application of one single grammatical framework, but rather should allow the building of multiple levels of structure. From this viewpoint, parsing is a multilevel act, wherein the processes which mutually depend on each other on the single levels are done in parallel. Successful parsing incorporates successful construction of structures on *all* levels. Failure on one level results in failure of the whole parsing act.

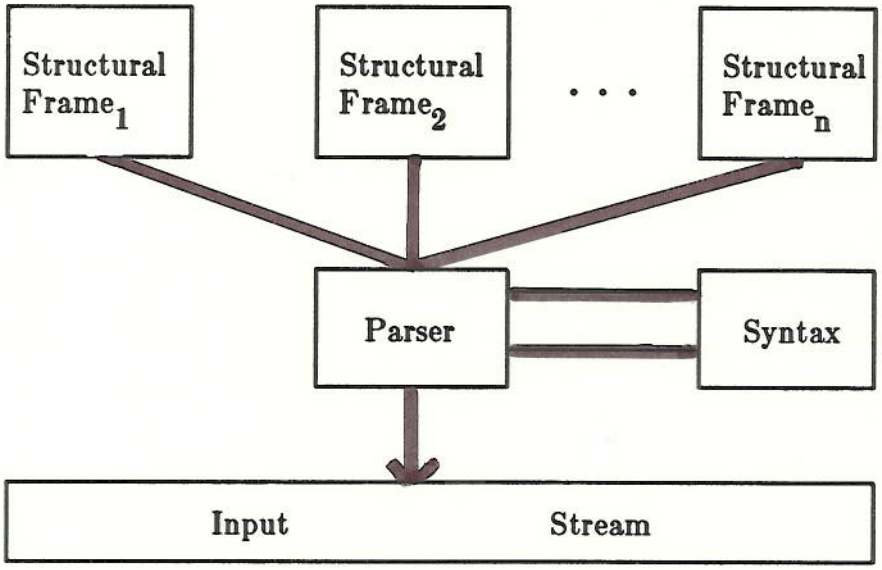

Figure 1

Figure 1 illustrates this. The parser in the center interacts with several different frames, each of which is the implementation of one level of the grammar. We now use the word grammar in talking about the whole of all structure definitions, not for what is usually called a syntactic grammar alone. The input string consists of symbols with different features with respect to all levels. Whether interpreting the grammar or deriving a semantic frame, this allows the parser to build up the pattern of the sentence. If the pattern in each structure is ultimately validated and the input string is empty, then parsing has been successfully completed.

One of the levels, the syntactic level, is necessarily different from the others because it forms the center of the parsing act. The structure on this level is built according to the position of the terminals, which controls the construction of structures at the other levels. While other frames are more autonomous, parsing at the syntactic level is much more visible to the grammar interpreter. We will show this in more detail later with an example of sentence parsing.

The organization of our parsing system can be compared with other systems that attempt to coordinate multiple conceptual levels. Two such systems, representing opposite extremes of organization, are now examined.

The LUNAR system (Woods et al., 1972) uses a sequential organization in which essentially all syntactic processing precedes semantic processing. The need for multiple perspectives in parsing has certainly been recognized for a long time and the LUNAR approach typifies early attempts to deal with both syntax and semantics in a single system. The difficulty of separating the two components in this way stems from the inability of syntax to verify its structures semantically. Thus, essentially all possible parses must be provided by syntax so that semantics can work to filter out those that are semantically anomolous.

The architecture of the HEARSAY system (Erman, et al., 1980) uses a blackboard model to coordinate hypotheses from several cooperating knowledge sources (KS). In this way, each individual KS competes with other KSs in attempting to apply its knowledge to the problem. Multiple levels of the problem are addressed by KSs which possess expertise for that particular level, but each level is kept separate from the other.

We find both of these approaches lacking in various ways. The LUNAR approach fails to capitalize on the interactions possible between levels to constrain processing at each level. HEARSAY provides a framework for interaction, but abdicates control to the KS itself, which complicates the task of the KS unnecessarily. It also requires that each level be separately encoded in a KS, when there may be great advantage in combining common components.

3. Wait and See Parsing

This section gives a brief overview of deterministic parsing (Marcus, 1980; Hindle, 1983). Although our aim is to suggest general approaches and mechanisms, this model is used because it has proven to be a powerful tool for our purposes.

A wait-and-see parser is based on two data structures: the BUFFER and the NODESTACK. The buffer, which has limited length, can contain input elements and/or structures. Elements from the input string enter at the right. Elements from the node stack can be inserted, at an arbitrary position, as a result of an action. Actions concerning the buffer are:

[1] Reading the i-th element (if the i-th place is still empty then more elements are read from the input string and inserted into the buffer)

[2] Inserting an element on the i-th place

[3] Deleting the i-th element

The node stack contains nodes from the parse tree. Elements from the buffer are attached to the "current node" or topmost node of the stack. Some nodes may be of a special type, like S and NP for a sentence parser. The node of this type with the highest stack position is called the "current cyclic node" and is also accessible. Actions concerning the node stack are:

[1] Attaching an element from the buffer to the current node

[2] Popping the current node and inserting it into the buffer

[3] Popping and attaching to the next node in the stack

[4] Creating a new node

At any time, properties of all buffer elements, the current node, and the current cyclic node can be read, changed or added.

These actions are tools for the grammar interpreter, which is controlled by rules. Rules are divided into rule packets which can be activated and deactivated. Activation is bound to the current node in the stack. If a new node is

created, the rules that were active before are hidden and appear again when the new node is popped from the stack. Thus, the active rules at a certain moment are those contained in the packets marked as activated for the current node. Every rule also has a priority number, so rules with the highest priority are considered first. Each rule looks like

$$(\text{COND}_1\ \text{COND}_2\ \text{COND}_3\ \ldots \rightarrow \text{ACTION}_1\ \text{ACTION}_2\ \ldots\ )$$

If all conditions are fulfilled, the rule fires and the actions are executed. normally the conditions check for certain features of the buffer elements.

Special rules, called *attention shifting rules*, serve to parse lower level structures before they are used by higher level rules. For example, a condition of a rule could test for an NP at a certain buffer position. This would first require that an NP be parsed, because the buffer currently contained, perhaps an Article, the starting word of an NP. Attention shifting rules create a window over the buffer so that the buffer virtually starts at the position containing the Article. This window exists until an NP is successfully parsed and the original buffer is restored by another action.

This parser is deterministic in the sense that no structure is built on the node stack that is allowed to be thrown away. Therefore, no real backtracking is permitted and all decisions are made by looking at the buffer elements and the current node.

4. A Parsing Example Based on Syntax and Semantics

A detailed example for multi-level parsing is now presented in which both syntactic and semantic structures are built. The interaction of syntax and semantics of natural language has long been a point for discussion. Systems for parsing natural language sentences can be ranked on a scale from purely syntactic to almost purely semantic. We claim that syntactic and semantic analysis must be done concurrently, if one component is to influence or help the other in building up a structure of the sentence.

This is easily demonstrated with sentences like:

(a) I saw the man with the red hat

(b) I saw the man through my glasses

The PP "with the red hat" in (a) clearly must be attached to the NP "the man", while in (b) the PP "through my glasses" is a constituent on the sentence level. Needless to say, this cannot be decided using syntax alone.

Therefore, we augmented the grammar rules by adding a level of structure which contains semantic information. The semantic structure of a sentence is here based on case frames (Fillmore, 1968; Dorffner, 1984) which are defined generally, in the same manner that a context free grammar defines the syntactic structure. The case frame gets filled in during parsing.

The center of the main clause is the main verb. Each verb has a lexical feature that describes its possible arguments, or thematic roles, selected from a closed list (e.g. AGENT, OBJECT, INSTRUMENT, etc.). Each has certain restrictions associated with it in order to be a candidate for a given role slot. These restrictions can be of a syntactic or semantic nature.

For example, the verb SEE could have the following arguments:

Role	Restrictions	
	Syntactic	Semantic
AGENT	(NP, NOMINATIVE)	(ANIMATE)
OBJECT	(NP, OBJ-CASE)	(PHYS-OBJ)
INSTRUMENT	(PP, with)	(OPTICAL-DEVICE)

For the sake of simplicity the semantic properties of a role and also of words in the dictionary are lists of features.

The semantic interpretation of sentence (b) could therefore be:

	Case Frame	
Head/Verb:	see	
	AGENT:	"I"
	OBJECT:	"man"
	INSTRUMENT:	"my glasses"

This structure, however, does not capture the full spectrum of the meaning of a sentence. How could, for example, the fact in sentence (a) that the hat belongs to the man be represented? For this reason, we also introduced a concept, similar to the case frames, for nouns, which we will call a "noun-frame." Just as the case frame is the semantic correlate of a sentence, the noun frame corresponds to NPs and PPs. Like case frames, noun frames have a head, which in this case is a noun, plus slots to fill. For these slots we introduced categories like ATTRIBUTE (corresponds to adjectives), POSSESSIVE (like "with the head"), LOCUS (like "on the hill"), etc. Again, the nature of the noun which gave rise to this instance of a frame determines the actual number of slots and their restrictions. Restrictions also exist, as described above and can require syntactic categories (e.g., ADJ, PP) as well as semantic features. An adjective that fits with the noun "man", for example, could be called DES-PERSON (for "describe"). From this, "strong" would be DES-PERSON while "liquid" would not, since we do not consider metaphors.

There is no reason to make a distinction between NPs and PPs on a semantic level, because both syntactic structures express the same phenomena on different surface levels, (morphological case versus prepositionally marked case). In fact, what is or is not expressed by morphological case varies from language to language.

The order in which the roles appear in the frame determines which role is assigned if a constituent matches several slots. The verb "see," for instance, lists its arguments in the order (AGENT OBJECT INSTRUMENT). In the sentence

"I see you," "I" is assigned to AGENT and "you" to OBJECT. Note that it is only the inversion of order due to syntactic transformations, (for example, in the passive "You are seen by me"), that is handled by the WASP rules. Important here is the fact that semantic roles are no longer bound to syntactic constituents. For example, the verb "give" will have the role list (AGENT RECIPIENT OBJECT), while "receive" has (RECIPIENT OBJECT AGENT) which binds the subject to RECIPIENT.

In both the case frame and the noun frame, some constituents occur before the head (e.g. the subject of a sentence occurs before the verb). To handle this, an additional variable was introduced to store this constituent until the head is parsed. If the latter is done, this constituent is treated as all the others and stored in the frame once again.

As we already mentioned, syntactic and semantic parsing is done at the same time, making use of the inter-dependencies between syntax and semantics. Syntax helps to find the right constituents, while the semantics disambiguates many sentences with more than one syntactic structure. It can also help to reject "meaningless" sentences early in the parsing process.

To build this into the Wait and See Parser we augmented its rules as follows:

$$(\mathrm{COND}_1\ \mathrm{COND}_2 \ldots \mathrm{COND}_k\ /\ \mathrm{COND}_{(k+1)} \ldots \rightarrow \mathrm{ACTION}_1\ \mathrm{ACTION}_2 \ldots)$$

The conditions COND_1 through COND_k each correspond to an element in the buffer (e.g. checking for a feature of the buffer position). The conditions after the "/" are additional tests. This distinction is necessary because the rule application procedure tries to apply an attention shifting rule corresponding to one of the buffer positions while scanning through the conditions 1 to k.

Both checking and building a semantic structure is now implemented as part of the rule conditions. Each rule that attaches an NP or a PP gets an additional clause in its conditional part (after the "/"). This clause actually is an action, a function that sends the corresponding noun frame to the current case frame. If it is accepted, then the condition succeeds. If it is not, then the condition fails and the rule does not fire. Success of all conditions allows the actions of the rule to be applied. Rules on the NP-level are modified in a similar way. This prevents, for example, the PP "through my glasses" from becoming attached to "the man" while parsing sentence (b) above.

Figure 2 shows an example for an augmented rule. It attaches a PP like "with the red hat" to an NP, if the noun frame (e.g., that for "man") accepts it.

```
(set-rule  attach-PP-in-NP
    (feature 'PP 1)                ; test: is first buffer element a PP?
    /
    (send-frame (bufread 1) nil)   ; send PP to case frame, accept or reject
    →
    (attach-from-buffer 1)         ; attach PP
)
```

Figure 2

Figure 3 illustrates the output for sentence (a):

```
((np: (nbar: (pronoun: I)))
 (aux:)
 (vp: (verb: see)
     (nps:                    ;SYNTACTIC STRUCTURE
      ((det: the)
       (nbar: (noun: man)
            (pps:
             ((prep: with)
              (np: (det: the) (nbar: (adjs: (red)) (noun: hat)))))))))))

(verb: see                    ; SEMANTIC STRUCTURE
      ((agent: (noun: I))
       (object: (noun: man ((poss: (noun: hat ((attributes: red)))))))))
```

Figure 3

Since the parser builds a semantic structure of the sentence directed by the syntax one could argue why we do not discard the syntactic structure altogether. It is useful, however, to maintain the syntax tree since it contains some important information: for example, HOW something was said (passive vs. active, tense, etc.) while the frame structure tells WHAT was said.

This parser serves only as an example of how multilevel parsing could be implemented in the deterministic framework of a wait and see parser. The parallel building of structure on different levels is simulated by a modification to the left hand side of the rules. Elements of the additional structural levels (here, semantic representations of words and phrases) are sent to autonomous frames representing these structures. These elements are either accepted as appropriate or rejected as inappropriate by the frame. If accepted, the rule is fired and parsing continues. If not, this rule does not lead to a successful parse.

As we described earlier, one of the structural levels, the syntax level, is extracted and fully visible to the parser. All the syntactic rules are formulated explicitly and control the whole parsing process. Most rules in the other levels, for example the semantic grammar, are formulated within the autonomous frame. For this reason the structure in these levels are not directly dependent on the

syntactic rules. The only thing the grammar interpreter has to know is when to send elements to the frames. It would be possible, for example, for a rule to send an element to one of the frames without contributing to the syntactic structure.

In the above example we were talking about a parser with two levels. Of course, we do not have to restrict ourselves to merely two. If more than two levels are employed, then on the left hand side of the rules, elements are sent to frames only to test their acceptability. If all frames accept, the rule can fire. On the right hand side, the elements are actually attached. This has to be done because we are not allowed to detach any element from a structure since this would violate the determinism of the parser design. Some possible additional structural levels for sentences could be:

[1] a stress or intonation structure in spoken language;

[2] a reference structure for definite noun phrases and pronouns — to solve references like: "Peter knew that he was ill" vs. "He knew that Peter was ill."

5. Monotonic Structure Building

In this section we address additional issues for deterministic parsing that go beyond the work of Marcus. He introduced the wait-and-see parser as a tool for parsing natural language sentences, with lexical words as the smallest units.

We are applying this parsing framework to other aspects of language processing as well. In particular, we want to be able to parse a low level description of acoustic speech input involving some kind of phonetic labels as the smallest elements. Since we are still processing natural language, Marcus' justification for the deterministic model is still valid. That is, it seems plausible that human understanding of speech works to a great extent without backtracking and, therefore, is deterministic. We tentatively choose a variable length buffer until further experiments provide justification for a definite size. But this is not the only detail that has to be reconsidered.

Low level procedures which analyze the speech signal and which we call label finders, are not reliable enough to make an absolute decision about a label. For example, in analyzing the utterance "feel" we might not be able to tell confidently that the fricative at the beginning of the word is an [f], because in some utterances it might resemble an [s]. However, the decision that it is a fricative, and not a stop like [t], should be confidently determined. This means that, unlike the task of parsing written language, the elements of the input string may not be known in full detail when they appear in the buffer for parsing. Moreover, the decision about these details often must be deferred to a later point.

In order to make a deterministic parser to handle this situation, we have two choices. One is to extend the length of the buffer (i.e. the length of the lookahead) and formulate rules that look at enough elements of the input string to make a definite decision. But this has a great disadvantage. The rules get extremely complex and involve a lot of different processes, (such as, lexical access

and others) to make a decision about the fricative in the above example. The second choice, and the one we want to suggest, is to build a structure on the node stack with those elements that the parser feels confident in identifying. If a more detailed decision can be made, this information is added to the node already on the node stack. We refer to the procedure of adding information to the node stack as "monotonic" structure building. We start at a low level of information and always *add* information but never *throw away* anything already parsed. This would be counter to the deterministic assumption.

Implementation of monotonic structure building implies a hierarchy of input elements, as in Figure 4.

Figure 4

As we add information we go down this hierarchy toward more detail. In the above example, the label fric (fricative) would be one successor of a less detailed label C(consonant) and itself would have several more detailed labels, like [s] and [f]. This suggests that we could involve this hierarchy in the (syntactic) structure tree. There, we have the mechanism of attention shifting rules which are perfectly designed to handle different levels of input elements. For example, we could have a rule which asks for a "C" in the first buffer position. If our label finders were good enough to detect a fricative and this fricative now appears in the buffer, there could be an attention shifting rule which fires and parses the fricative as a consonant. In other cases, the lower level structure could consist of more than one element. For example, a stop could consist of a closure followed by a burst. After finishing the parse, the consonant would be popped back into the buffer and the rule under consideration could fire. If, however, the label finder could not make a decision about the nature of the consonant, "C" would be the label that appears directly in the buffer, which again allows the rule to fire.

Unlike previous sections where we talked about "levels" referring to different structures of an input string, i.e. the breadth of structures, we now mean 'levels' going into depth of the syntactic structure tree.

This task is comparable to parsing sentences which might contain an NP or PP as input elements. The important difference in the style of parsing is that if we can get a more detailed description of an imprecise label later on, we can add this information to the structure in the node stack. Note, that this is not simply attaching buffer elements at nodes other than the current node, because in the final parse tree elements might appear which have never been part of the input string.

6. Phonetic Label Parsing —an Example

We already have been talking about parsing of speech input represented by some kind of phonetic elements. We now look at this task in more detail to show the advantages of monotonic structure building.

The purpose of parsing acoustic phonetic strings is to help a speech recognition system recognize words in a continuous utterance. As we already mentioned, a program on the level of the acoustic signal would be applied that detects certain acoustic segments in the speech input and assigns labels to them. These labels could look like the ones in Figure 5, where fric stands for fricative, *t*cl stands for *t*-closure, and *p*cl stands for *p*-closure.

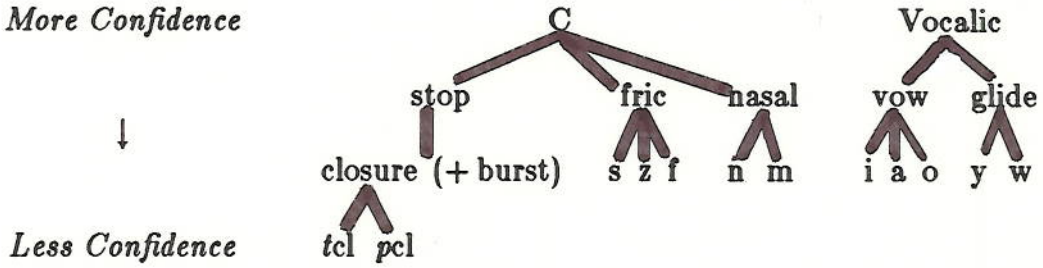

Figure 5

Figure 5 depicts a possible hierarchical structure, corresponding to what we described in the last section. Parsing should be possible using labels on any level of this hierarchy. The more specific the classification, the less confidence generally in assigning that label. One task, for example, is to decide on possible syllable boundaries. A consonant between two vowels could be the end of a syllable having the first vowel as its nucleus or it could belong to the syllable having the second vowel as its nucleus. That is, it could be the coda of the first syllable or the onset of the latter.

One possible tool to aid in developing such decision-making rules is a technique called *discriminant analysis*. We employ it to find the most powerful linear combination of segmental duration measurements for the purpose of discriminating words or syllable structures despite variations in speaker and tempo. We have successfully applied this method to durational measurements of speech segments to distinguish small word sets using temporal measures only (Reilly & Port, 1985). There appear to be different relationships between the durations of segments, (for example, the two vowels and a stop closure) depending on the type of syllable structure. This kind of statistical analysis, if trained on a corpus of tokens, can be used to reliably detect such relationships. The important fact is, that discriminant analysis can be successfully applied even if only the label "C" or "fric," rather than the more specific "f" or "s" is known. This means that the parser does not have to wait until all details of the structure tree are known to apply a useful method to decide about these very details. Not having all the input elements in as much detail as will be required at the end of the parse does not prevent us from building a less-specific structure on the node

stack.

Therefore the act of parsing can employ bootstrapping: The input string contains those labels that can be clearly recognized by just looking at the speech signal. With the help of these labels some methods (like discriminant analysis) can be applied, which results in more refined labels, and so on. This is what distinguishes this system from other, previously suggested ones, for example the work of Church (1983).

Of course, there are several other sources of information that can help in making the decisions necessary to refine the structure on the node stack. If, for example, the lexicon of the system contains strings on all label levels (e.g., *pan* as both [*p*cl-*p*b-asp-*ae*-*n*] and [stop-vow-nas]), it could be one of these sources.

Consider another example. Let our input string be

vow fric stop ih *p*cl *p*b/

where "ih" stands for a long [i], *p*b for *p*-burst, and vow is an arbitrary vowel. Discriminant analysis applied on the durations of all (or some of) the segments could decide that the syllable or word boundary lies between "vow" and "fric":

vow / fric stop ih *p*cl *p*b/

Now "fric" and "stop" can be parsed as the onset of the next syllable by building a node on the nodestack and attaching these two elements. But, the only fric-stop sequence in English phonology that is possible as a syllable onset is one with "s" as the fricative. This information can now be added.

As we proceed, we see that the next three labels ("ih", "*p*cl", "*p*b") are detailed ones and we parse them as the nucleus of the syllable and its coda, respectively. We are now ready to try lexical access as we reached the end of the utterance and can be sure that the word is over. The only word found is "steep" which has a "t" on the stop-position. Therefore, we assume that it is this word and add the information "t" to the onset node that contains "stop".

7. Future Perspective

It should be clear from the example above that our approach is aimed at interpreting the speech signal in a rather bottom-up fashion. The least commitment principle has lead us to maintain a label hierarchy, which permits us to attach a label to a segment of speech according to our best information at the time it is needed.

Currently, we are developing our model in the context of speech recognition with particular attention being paid to prosody. We claim that this approach, with its multiple levels, determinism, and buffered window into the signal, can adequately support our approach to speech recognition. We are in the process of substantiating that claim.

What we are proposing as a model of multilevel parsing preserves all of the important features of a WASP parser. In particular, if inputs are *ill-formed* (Kwasny & Sondheimer, 1981), most language processors are not equipped to handle such problems. Our proposal is perfectly compatible with the ideas expressed by Charniak (1983) for processing ill-formed inputs. In that work, tests performed on the left-hand sides of rules are weighted according to how important it is for those tests to succeed. Each rule is "scored" and the best scoring rule is fired. In this way, *some* rule can always fire, even if no rule scores very well. With a little care, the parsing can proceed even when confronted with an ill-formed input.

Using our elaborated scheme, the weights can be adjusted so as to force parsing even though the sentence might be ill-formed syntactically, semantically, etc. Furthermore, the weights can reflect the desire to have the parsing activity be affected to a greater or lesser extent by syntax, semantics, etc.

8. Conclusion

We have presented alternatives to known methods of parsing which both coordinate structures at multiple levels and aid in postponing premature classification of the input. Our method is superior in its ease of coordination among structures as well as in its ability to work deterministically. The price to pay for such benefits is small. The rules are only slightly more complicated, while the rule interpreter must be aware of the levels in the system. Rules may be developed just as fluently as in other WASP systems.

Every example discussed in this paper has been coded and tested on our system. The parsing framework has been constructed exactly as described here. We have yet to develop any large collection of rules to run with the system, but presently, we are developing more extensive tests for this model. We will have results from the first stages of this development very soon.

Acknowledgements

We would like to extend our appreciation to Daniel Maki and Will Reilly, for providing many comments that were most helpful in preparing this paper.

References

Charniak, E. (1983) "A Parser with Something for Everyone," Chapter 7 (pp.117-149) in: King, M. (Ed.) *Parsing Natural Language*, Academic Press, New York, NY, 1983.

Church, K. (1983) "A Finite-State Parser for Use in Speech Recognition," *Proceedings of the Association of Computational Linguistics*, 1983.

Dorffner, G. (1984) "Eine Kasusrahmengesteuerte ATN für Deutsche Hauptsätze," *Journal of the Austrian Society for Artificial Intelligence*, 3, 1984.

Erman, Lee D., F. Hayes-Roth, V. R. Lesser, and D. R. Reddy (1980) "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty," *Computing Surveys* 12, 2, 213-253, June, 1980.

Fillmore, C. J. (1968) "The Case for Case," in: Bach and Harms (Eds.) *Universals in Linguistic Theory*, Holt, Rinehart, and Winston, Chicago, 1968.

Hindle, D. (1983) "Fidditch User's Manual," Naval Research Laboratory Technical Memorandum #7590-142, June, 1983.

Kwasny, S. C., and N. K. Sondheimer (1981) "Relaxation Techniques for Parsing Ill-Formed Input," *American Journal of Computational Linguistics*, Vol. 7, No. 2, April-June, 1981.

Marcus, M. P. (1980) *A Theory of Syntactic Recognition for Natural Language*, MIT Press, Cambridge, MA, 1980.

Marr, David (1982) *Vision*, W. H. Freeman, San Francisco, Ca, 1982.

Reilly W., and R. Port (1985) "Use of Timing to Discriminate Words," *Journal of Acoustic Society of America*, 78, S21, 1985.

Woods, W. A., R. M. Kaplan, and B. Nash-Webber (1972) "The LUNAR Sciences Natural Language Information System: Final Report," Report 2378, Bolt, Beranek, and Newman, Cambridge, MA, June, 1972.

Woods, W. A. (1980) "Cascaded ATN Grammars," *American Journal of Computational Linguistics*, Vol. 6, No. 1, 1980.