The Effects of Word Boundary Ambiguity on Lexical Access in Automatic Continuous Speech Recognition

Anne Johnstone

Ph.D.
University of Edinburgh
1989

Table of Contents

Preface

While some of the research reported in this thesis was completed during my employment on the Alvey/Edinburgh speech recognition project, the major motivations and methodological issues had been worked out before the project began. The first report of that work (Johnstone & Altmann 1984) was incorporated in the project proposal.

The programming work and the system tests formed part of the project's first prototype, RM1. Any mention of the acoustic-phonetic front-end, SEGLAB, or the syntactic component refers to work done in other sectors. As regards the Lexical Access component, I was the only person employed in the Lexical Access sector during the first three years of the project when this research was done.

I declare that:

a) this thesis has been composed by myself, and

b) the work is substantially my own. Any contributions from other members of the project are noted in the acknowledgements and at the relevant points in the text.

Abstract

The results of human speech processing are rarely ambiguous, in that people are usually clear about the words they have heard. Yet there are numerous sources of confusion which make the automatic access of lexical items a very difficult search problem. Our understanding of the relationship between acoustic signal and useful linguistic representations is still very limited. It is also unclear how people use linguistic and general contextual knowledge to overcome errors and ambiguities in the acoustic input, although they are undoubtedly able to do so. One particularly difficult problem is the ambiguity caused by lack of acoustic cues to word boundaries.

Chapter 1 formulates these problems at a level of abstraction which is general enough to capture the common processing aims of very different computational systems, yet also allows detailed analysis of the search problems. Graph-search terminology proved to be a useful framework both for analysing past systems and for guiding research on our own system. Chapters 2 and 3 describe the implementation of a lexical access component using a general graph management system, the Chart parser.

In Chapters 4, 5, and 6 the system was used to assess the effect of word boundary ambiguity on parsing a graph of phonological units into words. We showed that, even when all 44 phonemes of Received Pronunciation were used, a correctly transcribed input utterance of 4 - 10 words could be parsed into in excess of 10,000 word strings. When a less specific, mid-class phonemic representation was used, 71 of 115 test sentences could be parsed in over 10 million different ways. These results imply that, even with accurate mid-class labelling, strong syntactic and semantic constraints must be applied as early as possible in order to prevent a combinatorial explosion of word strings.

In the final chapter we look at the implications of these results for search strategies. We analyse several algorithms and show that certain strategies, such as the island driving strategies used in HWIM and Hearsay-II, are highly inappropriate for this kind of search space, while others, such as beam search will only be effective under certain conditions.

This thesis shows that study of the interactions between knowledge sources reveals problems that would not otherwise be apparent. Attempts to reduce the search space can then be directed to areas where they will be most effective.

Acknowledgements

Firstly, I would like to thank the founding members of the speech workshop -- my supervisor Henry Thompson, Ellen Bard and Gerry Altmann -- who got me into this mess in the first place. Henry and Ellen always provided stimulating debate, and were generous with their expertise in computational linguistics and psycholinguistics. My thanks to Gerry for his support and encouragement and for hounding me until the first paper got written.

I owe a great deal to all the members of the CSTR speech recognition group, but I am especially indebted to Greg Filz, Jonathan Harrington, and Maggie Cooper. Greg, with limitless good humour, implemented much of the RM1 system. Jonathan spent hours over the test runs, and many months discussing the experiments. Without his help much of this work would have remained at the preliminary stage. Maggie provided the lexicon, some good laughs, and put up with the permanent LA smog in the office.

Many, many thanks to Tony for all the sympathy and assorted beverages.

Finally, and most of all, my thanks to Peter.

Chapter 1. Lexical Access as a Search Problem

1.1. Introduction

Continuous speech processing requires the application of very many sources of knowledge in order to decode the utterance. Even if the signal were transcribed into a perfect description of the sound, the utterance might remain partially ambiguous without the application of further sources of information. A major cause of ambiguity is the lack of acoustic cues to word boundaries. For example /t o m ii t s/ could be heard as either *Tom eats* or *Tom meets* depending on its context.

The constraints from diverse knowledge sources are also needed to recover from underspecified or errorful input. For example, we would want to be able to recover *bracelet* from the input *bwacelet*. One of the major areas of research in speech recognition concerns the ways in which general linguistic knowledge can compensate for such errorful or ambiguous acoustic input.

The ARPA project of the mid-seventies relied heavily on the use of domain-specific syntactic and semantic knowledge to constrain the search space. It was generally believed that the poor performance of the front-end systems required this. Cole et al write:

> "Considering the amount of effort that has been devoted to speech recognition research, the "front-end" performance of speech recognition systems is surprisingly poor. Systems developed during the ARPA speech understanding project achieved first choice segmental recognition accuracies of 50% to 60% (Klatt 1977). This is not accurate enough to recognise words

> unless vocabulary choice is highly constrained, and the items at each choice point are acoustically distinct." (Cole, Stern & Lasry 1983)

However, advances in speech science and computer science since the ARPA project had encouraged the belief that front-end processing could be substantially improved. Such an improvement should permit recognition systems with larger vocabularies and generally weaker top-down constraints.

In addition, research on large lexicons seemed to imply that lexicon-based constraints could constrain the identity of a word on the basis of very little phonemic information. Nusbaum & Pisoni (1986) experimented with variable-grained encodings of a huge 126,000 word lexicon. They report:

> "These results demonstrate that detailed phonetic information about some of the segments in a word provides enough constraint, in general, that other segments can be completely obscured or ambiguous without significantly impairing recognition. Moreover, to the extent that some phonetic information is available about other segments, the candidate set will be reduced further, probably to the extent of uniquely specifying the correct word."

The implication of these results and those of Zue (1985) was that, so long as the correct phonemes were in the set of input descriptions, lexical and higher-level knowledge sources would be able to distinguish the correct words.

However, these studies had been carried out on words in isolation, and we felt that, given the ambiguity of word boundaries, the search space might prove much larger and more complex than these studies implied. The ARPA project had already shown that controlling the search in speech processing could be extremely hard.

We also felt that the interactions between different sources of knowledge was still far from clear. On the one hand some psycholinguistic data seemed to point to word-by-word recognition (Cole & Jakimik 1980; Marslen-Wilson & Welsh 1978). On the other hand there was evidence implying that even human listeners needed to hear stretches of speech several words long for accurate recognition (Pollack & Pickett 1963). Again, some

psycholinguists maintained that top-down constraint merely *facilitated* recognition, while others argued that top-down information had to *produce* hypotheses to compensate for inaccurate acoustic information.

1.2. The Alvey/Edinburgh Project

The implementation of a large scale speech recognition machine proposed by the Alvey/Edinburgh project would allow a number of the interaction issues to be explored. In particular, the lexical access component could be used to test the following hypothesis:

Structural constraints in the lexicon are sufficient to substantially reduce the number of word candidates in a string of underspecified phonemic units.

Our first aim was to examine the lexical access components of a number of existing speech recognition systems. We chose three systems from the ARPA project, HARPY, HWIM and HEARSAY-II, and a more recent, connectionist system, TRACE.

HARPY (Lowerre & Reddy 1980) was a highly constrained system, in terms of both architecture and task domain. Although the knowledge sources in HARPY were designed separately, they were subsequently compiled into a unified directed graph representation which was then used to decode the utterance.

HEARSAY-II (Erman & Lesser 1980) used the same constrained vocabulary and grammar as HARPY but had a more flexible architecture. The knowledge sources

communicated through a blackboard data structure. The major constraint on architecture was the decision to use production rules to represent knowledge.

HWIM (Woods et al 1976) used a 1,000 word domain specific vocabulary like HARPY and HEARSAY-II, but a far more general grammar. The system can be described as a set of cascaded ATN networks which communicated via an ordered agenda.

TRACE (McClelland & Elman 1986) is a more recent system whose aims are in some ways similar to ours. The designers also emphasise the interactions between components, and the relevance of psychological models of human speech processing. However, their architecture -- a connectionist network -- limits them to a small vocabulary of 211 words. This is because current connectionist networks must duplicate all the nodes and connections for each unit in the system in order to represent temporal aspects of the problem. No higher-level knowledge source is used.

There are a number of reasons why it is difficult to compare lexical access systems. Normally the lexicons that they use differ in size and content, so the number and type of lexical hypotheses generated may vary greatly. Moreover, the lexical access mechanisms are usually embedded in larger systems, and their success depends on both the performance of the lower level acoustic-phonetic components and on the predictive or selective abilities of the higher level syntactic and semantic components. In addition, most systems use search strategies to eliminate some partial hypotheses from consideration, and it is not always easy to determine the relative importance of the various components of these strategies.

Thus performance statistics alone are not very useful for comparing different systems, since they do not necessarily provide any indication of the superiority of one system over another, nor any insight into the strengths and weaknesses of the various components.

1.3. Graph theory and speech recognition

At a certain level of abstraction there is a sense in which any speech processing mechanism is trying to solve the same search problem (see Goodman & Reddy 1980). The interpretation of an utterance involves the integration of information *across different levels of linguistic description* and *across time*. Ascending levels of abstraction (i.e. phonetic, lexical, syntactic, semantic) typically deal with temporally longer stretches of the utterance. Since information is locally ambiguous, at various stages of processing there will be competing interpretations which require disambiguating information from other knowledge sources and from temporally earlier or later parts of the utterance. We can represent this as a three-dimensional search space (see Fig. 1.1). The search process can then be defined as follows:

> Combine information during the extension of a hypothesis (along the x axis), and during processing by different knowledge sources (along the y axis), in order to create a set of competing hypotheses (along the z axis).

We define a *valid path* within this space as:

> A hypothesis which fits the constraints of the knowledge sources. Most systems also have some method of assigning scores which rank competing hypotheses according to how well they match the constraints, or according to their probability given the evidence.

The *goal* of the search can now be formulated as:

> Find the path spanning the utterance (along the x axis) which fits the constraints of the levels (along the y axis) better than any of the other paths (along the z axis).

In HARPY, all the possible paths are specified explicitly before processing begins and collapsed into two dimensions, while in HWIM and Hearsay-II they are generated dynamically according to rules which combine partial descriptions linearly (across time) and hierarchically (across levels of linguistic description). In TRACE, the links between levels of description are explicit, hard-wired connections, and the links across time are represented by the simultaneous activity of sets of nodes in different time slices.

In each case, the goal is to find the best scoring path through an (explicit or implicit) three-dimensional space of phonemes, diphones, syllables, words, whatever, according to the descriptions of valid paths stored in the knowledge bases.

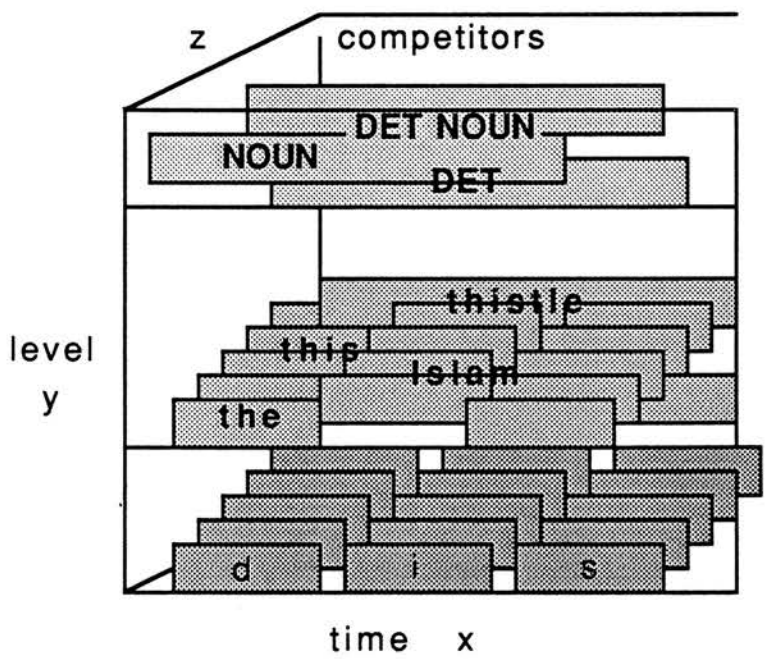

Figure 1.1

Speech recognition viewed as a three dimensional search space.

We can view the space of possibilities at the lexical access level as a horizontal slice through the search space, representing a graph of word hypotheses. This graph can be linked recursively to a graph of lower-level hypotheses and a graph of higher-level hypotheses.

Graph search terminology allows us to give a breakdown of the minimum requirements of the sort of mechanism required to search this space, regardless of (i) how information about valid paths is retrieved, (ii) how the search space is constructed and stored, and (iii) the specific strategies used for searching that space.

> 1) We need some way of deciding where in the graph we should start.
>
> 2) We need some way of deciding where to go next. That is, we need rules or descriptions of some sort defining valid paths. This is what Nilsson calls the *implicit* graph. (Nilsson 1980 p 63)
>
> 3) We need some way of deciding where to go at any particular point in processing. We need a successor operator of some sort that matches the current state against the possible states described by the rules in order to decide which of these should be extended. In Nilsson's terminology this process would create the *explicit* graph.
>
> 4) We will probably need to keep a record of this explicit graph in order to represent and resolve competing interpretations which may be disambiguated by later information.
>
> 5) Finally, we need some way of knowing when to stop.

Describing the lexical access task in these very general terms provides a common vocabulary for comparing design decisions in each of the areas listed above. This will help to distinguish those decisions which are required by the general graph-searching characteristics of the task, those required by the specific nature of the speech processing task, and those required by the particular system architecture.

1.4. Summary and Thesis Outline

The speech processing task consists of co-ordinating many diverse sources of knowledge in the search for an interpretation of an utterance. The graph-based analysis proposed here focuses on the *dynamic* aspects of this task. The complexity of a recognition system is often described in terms of its components: the goodness of the acoustic model, the confusability of items in the lexicon, the amount of constraint provided by the grammar. By concentrating on the kind of search space produced during lexical access, we provide a way of analysing the interactions between these sources of complexity.

As we shall see this method reveals a serious problem, one that a component-by-component analysis tends to conceal. The problem is one of *word boundary ambiguity*. It is known that there are few, if any, reliable acoustic cues to word boundaries. I argue that the effect of this on the size of the search space was concealed during the ARPA project as much by the design of the systems as by poor front-end processing.

In the first chapter we examine four lexical access components (HARPY, HWIM, HEARSAY-II, and TRACE) using the graph-based terminology outlined above. This analysis provided some useful insights which guided (i) the design of a system architecture described in Chapter 2, and (ii) a model of lexical access to be implemented within the architecture. The lexical access component, described in Chapter 3, incorporates some of the best features of the previous systems. It is also motivated by psycholinguistic evidence, although we make no claims of psychological reality for the model.

Chapter 4 describes the evaluation of the lexical access component over data provided by the Alvey project's front-end processor. The evaluation suggested some further experiments which are described in Chapters 5 and 6. These experiments show that the

recognition strategies suggested by structural analysis of large lexicons do not, in their present form, extend to continuous speech recognition.

The final two chapters look more closely at the interaction between top-down and bottom-up information given the experimental results. We are able to explain why certain search strategies (such as island-driving) are fundamentally misguided, and why other strategies (such as beam search) require certain specific conditions for success. The analysis of the search space together with the experimental results indicates why general speech recognition is not possible without highly sophisticated syntactic, semantic and prosodic information.

Chapter 2. A Graph Search Analysis of Four Systems

In the previous chapter we listed the five principal areas of choice for any graph search mechanism. In this chapter we shall analyse the decisions made in each area by the four speech processing systems, HARPY, HWIM, HEARSAY-II, and TRACE. We are interested in distinguishing common problems which cast light on the nature of the speech processing task, and in examining the effectiveness of each system's solution.

2.1. Deciding where to start

Some search problems have a choice about where to begin. While it may seem intuitively obvious that lexical access should start right at the beginning of the utterance and soon after its onset, the four systems tried a number of different approaches.

HARPY

HARPY's unified graph design leaves the system with little alternative but to work left-to-right through the utterance, since there are no individual components whose attention might be directed to one portion or another of the utterance. The input is segmented and labelled and these phone labels are then matched to the initial states in the knowledge network. Each match to a legal state is given a score. Illegal phone labels (i.e. states not specified at the beginning of the network) can be ignored. Each path that matches the input

is extended in parallel from left to right. In theory, HARPY could maintain a path for each legal sequence through the network. In practice, this overloaded the system and various methods were used to prune paths from the search tree.

HWIM

HWIM's architecture permitted its designers to experiment with a number of approaches to this question. It differs from systems such as HARPY in that the search space is fully 3-dimensional, graphs of words and phrases being produced on top of a graph of phoneme hypotheses.

This independence of knowledge sources during processing meant that the lexical access component could be directed to any part of the phoneme graph, and thus left to right strategies as used in HARPY could be compared with island-driving strategies which began anywhere along the time dimension of the search space.

All the search strategies tested with HWIM fall within a common hypothesise and test framework. An initial scan of the phoneme graph produces some number of best matching words bottom-up which are ordered by score on an agenda for processing by higher level components. Island-driving strategies perform this initial scan over the entire utterance. Left to right strategies consider only those word matches whose left-hand end corresponds to a possible utterance initial boundary. Hybrid strategies perform the scan over some specified initial portion of the utterance and work middle-out from any seed words found within this region.

Since HWIM's acoustic-phonetic component cannot be relied upon to correctly segment and label at the phonetic level, a number of alternative hypotheses have to be maintained to ensure that the right word has a chance of being considered. In HWIM, all 71 possible labels are given a probability score. These alternative phoneme labels could support a large

number of word hypotheses, depending on the number and phonetic closeness of items in the lexicon, and these in turn could be combined into a number of different word strings, depending on the constraints of the grammar.

Since HWIM's grammar was far less tightly constrained than the finite state grammar used in HARPY, the system hypothesised a considerable number of words at the beginning of an utterance. Unfortunately, the beginnings of utterances appear to be particularly unconstrained by either the acoustic material or the linguistic interpretation. This, together with HWIM's poor acoustic-phonetic labelling, caused a combinatorial explosion of hypotheses. Therefore, the designers explored alternative means of getting started.

The rationale behind island-driving was the expectation that some words in the utterance would be pronounced sufficiently clearly to be recognised out of context, and could therefore be used as islands of comparative certainty from which to build an interpretation of the rest of the utterance. This would seem to overcome the problem of segmenting and labeling areas of poor acoustic quality, and so minimise the problem of deciding which and how many of the alternatives should be maintained during left to right processing.

Woods (1982) reports that, with the addition of certain efficiency techniques to island-driving, there was little difference in the success rates of their hybrid strategy starting near the left of the utterance, and their island driving strategy starting anywhere along the time-dimension. We will return to this issue in later chapters.

Hearsay-II

The Hearsay-II system also has both left-to-right and island-driving capabilities. Words are first matched against the input on the basis of their syllable structure. These words are then scored on the basis of their match to a HARPY-like pronunciation network.

Finally a control component proposes the best scoring words over the utterance. This is limited to the lefthand end of the utterance when a left-to-right strategy is used. Words are proposed anywhere in the utterance when island-driving is used.

The test runs described by Erman and Lesser (1980) use an island-driving strategy. Erman and Lesser point out one potential advantage of island driving over left-to-right strategies: when more than one initial island is correct this should increase the probability of finding the correct spanning interpretation, since each island represents an alternative possible derivation of the interpretation. However, this must be weighed against the possibility that the island is a *false peak*, i.e. it scores well but is in fact incorrect. (Hayes-Roth & Lesser 1977). In order to reduce the likelihood of this happening Hearsay-II (unlike HWIM) used initial islands consisting of word pairs rather than single words.

TRACE II

The search space in TRACE II is also three-dimensional in that several distinct levels of linguistic description are represented, and the information from all these sources must be combined dynamically to form an interpretation. Unlike HWIM and HEARSAY-II however, these different levels of processing cannot be controlled independently. TRACE II has no choice but to start at the beginning and work rightwards, since input directed to feature detectors automatically results in activity at the phonetic and lexical levels through the hard-wired connections. Lexical access is not an autonomous process which a central processor can direct towards any part of the utterance.

TRACE II maintains competing partial interpretations through the use of massive parallelism. The entire lexicon is duplicated every few time slices. This is feasible because TRACE II has only 211 items in the lexicon. Furthermore the designers are mainly

concerned with lexical access, rather than speech recognition as a whole and the "utterances" are at most a few words long.

2.1.1.Conclusions

The performance of these systems indicated that poor acoustic quality together with lack of top down constraint could cause the graph to grow rapidly, to be very "wide" from the beginning of the utterance. In Hearsay-II for example, the average number of words that can follow any initial portion of the utterance is thirty-four. The average ranking of the correct word is only three, and a number of words (between 5 and 25) have to be hypothesized in the hope of including the correct word. This quickly leads to a very large number of partial hypotheses as words are combined.

The assumption was that later information would allow some of these paths to be eliminated; the graph would "narrow" because hypotheses at some point were acoustically clearer and/or limited by top-down information. The designers were facing the question:- Can the search space be kept within manageable proportions until this disambiguating information is reached, or should one try to find the "narrow" point and start from there?

HARPY and TRACE II took the first option. As we shall see in later sections, HARPY made sure, by structuring the grammar in particular ways, that a narrow portion *would* occur soon enough to prevent a combinatorial explosion of hypotheses. TRACE II does not have a syntactic component, but has a mechanism for such information to be used. The search space is of a manageable size because TRACE II has a small lexicon and short test utterances. Although HARPY and TRACE II "hard-wired" the left-to-right decision, they did so for very different reasons. HARPY was primarily concerned with achieving near real-time speech recognition for a specific, constrained task. TRACE II was primarily

concerned with exploring a very slow but psychologically plausible simulation of speech processing.

Both HWIM and Hearsay-II used an architecture which allowed them to experiment with alternative means of getting started. The island-driving strategy in HWIM looked for "narrow" points based on acoustic quality. Hearsay-II also looked for "narrow" points but used word adjacency scores as well as acoustic quality scores for the initial islands.

The case for island-driving is often argued in terms of efficiency. Goodman & Reddy (1980) for example, write:

> "Proponents of island driving argue that extending the globally best interpretation is more efficient since it approaches the recognition goal in the obvious, direct manner. Further, accuracy is better because the method does not consider portions of the utterances with low credibility until they become possible extensions of the current best interpretation, whereas a left-to-right strategy is forced to deal with unpromising portions as they occur in the utterance. if this happens at the beginning of an utterance, a left-to-right strategy may consume a great amount of time examining interpretations which look good initially, but cannot be completed.
>
> Proponents of left-to-right strategy argue that it is much simpler, requiring far less bookkeeping , and thus leads to greater efficiency. Also, this method can achieve the same accuracy by using a best-few search which explores more alternatives during portions of the utterances where credibility is low." (p 243)

Chapters 5 and 6 of this thesis question the validity of these assumptions. I show that the occurrence of false peaks can be far more frequent than was previously thought. The problem is not just one of poor acoustic input but also of ambiguous word boundaries. Not only are there many accidental matches to parts of the intended utterance (e.g. *ermine* in *terminal*), these false matches can form paths several words long. An island-driving strategy will be seriously misled by these extraneous word matches.

Was the rapid expansion of the search space due to the nature of speech itself, or to our limited understanding of speech processing?

It is undoubtedly the case that HARPY, HWIM and Hearsay-II suffered from poor bottom-up processing, and that this was the major reason why early identification of the initial portion of the utterance was so difficult. However there is psycholinguistic evidence indicating that human listeners also have problems identifying the first few words in an utterance. Pollack & Pickett (63), presenting listeners with stretches of conversational speech in a gating experiment, found that samples of about 140 csec (i.e. about seven words) were required before intelligibility reached 90%, despite the fact that listeners knew how many words were in each sample, and, for later samples, were hearing repetitions of the initial context.

Thus it would appear that there is not always enough information in the speech wave for identification to take place immediately. Later information is necessary to disambiguate earlier stretches. Such *right-context effects* (Thompson 84) are not limited to just the initial portion of the utterance. I shall discuss these effects further in section 2.3.

2.2 Deciding where to go next

Having decided upon a starting place, what can follow from the current state? Earlier we defined a valid path in terms of some higher-level knowledge source. One can think of laying a graph of words over a graph of lower-level units such as phonemes. The restrictions embodied in the lexical graph will eliminate some of the paths through the phoneme graph. Similarly, a graph of syntactic categories will eliminate some of the possible word sequences. With reference to the three-dimensional search space shown in Figure 1.1, we are building a recursive graph along the time and the knowledge-level axes.

There are two main problems in constructing this portion of the search space:

1) **The access problem:** how do we establish a correspondence between elements at different levels of description? (i.e. between a sequence of phonemes or syllables, and a word).

2) **The integration problem:** how do we combine knowledge from different levels? Should the search for a valid path be data-driven or goal-driven?

As Lowerre and Reddy (1980)remark;

> "An interesting aspect that distinguishes the speech problem from many other knowledge intensive systems in AI is the diversity of the knowledge sources (KSs). Each deals with a different aspect of the problem, and each 'speaks a different language.' Yet the KSs must cooperate somehow in decoding an unknown utterance." (p.146)

HARPY

In HARPY's case the implicit graph, the complete set of utterances permitted by the phonetic, phonological, lexical, and syntactic knowledge sources, can be made fully explicit. The use of a carefully structured finite-state grammar allows all the utterances to be specified ahead of time. All the information from each knowledge source is then compiled together in a unified graph representation. The higher levels are "collapsed" during an extensive pre-compilation period giving a two-dimensional graph bounded by the x (time) and z (competitors) axis. Thus the validity of a particular phoneme's identity can be judged immediately within its lexical and sentential context.

HARPY's organisation imposes an essentially goal-driven or top-down approach to recognition. Whereas a bottom-up or data-driven system such as HWIM can hypothesize any of the 1,000 or so items in its lexicon at the beginning of the utterance, HARPY can only hypothesize those words permitted at that point by the grammar. And as the grammar

allows on average only 10 possible choices the potential search space is considerably restricted. However, HARPY's acoustic-phonetic identification is still not perfect even with the top-down constraint provided by left context, and HARPY is forced to maintain a considerable number of paths through the search graph. A final decision is made at the end of the utterance. This use of later-occurring information or right context is discussed in the next section.

Since HARPY laid out each potential utterance as a path it had no need to segment the input into phonetic units before matching. The system compares input segments to spectral templates associated with states in the graph.

The most common criticism of HARPY is that its approach cannot be extended to larger vocabularies and/or more habitable grammars. In the Chapter 7 of this thesis I will discuss the restrictions of the HARPY system in more detail as well as a more flexible system which uses a bi-gram based finite-state grammar.

HWIM

The HWIM system initially took a bottom-up, data-driven approach, at least as far as the lexical level, but encountered serious problems in matching the phonetic input to the stored lexical representations. The matching process had to be relaxed considerably, both by allowing numerous competing phonetic labels, and by changing the lexical access procedure, in order for the correct word to be output at all (Klovstad 1976). This led to a large increase in the number of false positives, yet still did not ensure that the correct word was always the highest scoring.

HWIM's solution was to look for a number of best-scoring seed words bottom-up and then to predict possible extensions (words or word classes) at either end of the seed word. If one of the possible extensions scored well, i.e. the top-down prediction appeared to be

correct, then the island would be extended further, otherwise the seed would be abandoned for the next best seed word generated bottom-up over that area.

Particular search strategies had to be designed to compare, order and extend the different partial hypotheses at different parts of the utterance. These strategies are discussed in more detail in the next section.

In theory, areas of acoustic clarity would be identified on the basis of bottom-up information as seed words. Areas of poor acoustic quality but tightly constrained by local syntactic and semantic information would be identified through top-down prediction. Areas which were both ambiguous acoustically and relatively unconstrained by higher-level knowledge sources would not be processed until a more global interpretation of the utterance had been built up through the extension of various islands. The interpretations would then be predicted from this context.

In practice HWIM found only 2.17 correct words per sentence bottom-up. When doing anchored scans[1], the correct word scored highest less than 40% of the time. If the system were to recover from such poor bottom-up information, it would have to rely heavily on syntactic and semantic constraints. In fact the HWIM system had one of the least constrained grammars with an average branching factor of 196 and its performance suffered accordingly. The reasons for this are discussed in detail in Chapter 7.

HEARSAY-II

Hearsay-II had the largest number of contributing knowledge sources. The search space was organised around a global uniform data-base, the blackboard, and so different configurations of knowledge bases could easily be tested. This is in contrast to HWIM which designed individual interfaces for each of its components.

---

[1] An anchored scan looks for words to the right of a correctly identified word.

All the knowledge bases in Hearsay-II, ideally working in parallel and reacting as soon as relevant data appeared, would bombard the blackboard with knowledge, adding and modifying hypotheses, and thus triggering other knowledge sources in turn. The strategy was essentially data-driven, rather than centrally controlled like HWIM. However the explosion of hypotheses forced the designers to devise various strategies that focused on just some subset of the search space.

Hearsay-II's bottom-up lexical access appears to have been slightly more successful than HWIM's, though still very errorful. According to Mostow and Hayes-Roth (1978):

> "For a typical utterance, the word recogniser hypothesizes 20 incorrect words in the same time interval as each correct word, 4 of them with higher confidence ratings. Of the correct words 20% are not hypothesized at all." (p 472)

Like HWIM, Hearsay-II would have to rely on top-down information to recover the correct interpretation. The best performance results were obtained using the same constrained finite state grammar as HARPY with an average branching ratio of 10.

Even with such a highly constrained grammar, Hearsay-II fell foul of a combinatorial explosion of partial hypotheses. Mostow & Hayes-Roth (1978) attributed the problem to the rigidity of the production system rule schema. They write:

> "Although the data often contained grammatical sequences several words long (e.g. ME ABOUT BEEF), which would have made highly reliable predictors, these sequences seldom corresponded to complete templates and hence could not be detected by the precomputed production conditions. Instead, subsequences (eg ME) corresponding to lower-level templates ($ME) were used most frequently as predictors. These subsequences were usually only one or two words in length, and (...) appeared no more reliable as predictors than the large number of incorrect one- and two-word sequences. To fill in the unrecognised words, it was necessary to make many incorrect predictions along with the correct ones. The execution of an excessive number of unreliable productions tended to explode the search space combinatorially." (p. 479)

Their solution was to insert a new knowledge source. This knowledge source, WOSEQ, looked for highly rated, pair-wise grammatical words based on the bottom-up hypotheses. These multi-word islands were then used, much as the seed words in HWIM, as the basis for expanding the interpretation according to top-down predictions.

Hearsay-II's problems with its syntactic/semantic component highlights the difficulties of accessing pre-defined processing units in speech.

TRACE II

TRACE II consists of a large number of nodes organised into three levels corresponding to features, phonemes and words. Nodes on different levels which are mutually consistent have excitatory connections between them. Nodes at the same level which are mutually exclusive have inhibitory connections between them. Information flow in TRACE is very elegantly and easily controlled, but the system is, in some ways, far less ambitious than the three previous systems. It concentrates just on lexical access -- there is no syntactic or semantic component -- and its lexicon contains only 211 words. In addition TRACE II makes a number of simplifying assumptions about the input, although another version of the system, TRACE I, deals with real speech.

In discussing the previous three systems it was possible to say something about their knowledge sources and data structures in isolation from the processing performed over them. This was because these systems employ passive memory inspected by a sequential central processor. In TRACE II, a parallel architecture is employed in which each of the nodes is a relatively simple processing element which continues to send activation and inhibition to other nodes for as long as it remains active. Given this view of representation as activity, the discussion about *what* lexical and phonological knowledge TRACE II uses, will be discussed in the following section which deals with *how* such knowledge is used.

2.2.1. Conclusions

HARPY compiled all its knowledge into a graph of valid paths. This was an effective method of recognition, but it is generally agreed that the method works only for highly constrained tasks.

HWIM and Hearsay-II define sets of rules describing parts of valid utterances. These rules are applied to the data to construct, over the input, a graph of partial valid paths from which an interpretation of the entire utterance can be extracted.

Both systems were faced with a combinatorial explosion of partial solutions since the input data matched very many rules or partial descriptions. The causes of this explosion, outlined below, make speech processing particularly difficult.

The first problem is that a graph of units is being layed over a continuum. Not only do we not know what these units should be (e.g. phones, diphones, syllables, etc), there are few if any acoustic cues to the boundaries between such units. (Nakatani & Dukes 1977, Cole & Jakimik 1980). In addition, the information required to identify perceptual units such as phonemes overlaps in time with information about preceding and following phonemes. Thus segmentations of the speech wave necessarily *exclude* some information about the unit under consideration and *include* information about its neighbours. The cues to a phoneme's identity also vary depending on its context. Coarticulatory and other effects are increased with faster speaking rates. In addition to acoustic-phonetic effects, there are also phonological effects such as the deletion of /s t/ in the phrase *list some*.

HWIM and Hearsay-II's acoustic and language models were not good enough to map accurately the relationship between the input and linguistic descriptions of the utterance. The HWIM system in particular drastically relaxed matching constraints in order to ensure

that the correct phoneme or word was not excluded. Thus there were very many false positive matches indistinguishable from the correct match.

The solution forced on each system (apart from TRACE II) was to hypothesize as many words as possible top-down. The more constrained the grammar the greater the reduction in the number of incorrect partial solutions.

Even with such constraints the systems still generated too many partial solutions and had to devise scoring methods and control strategies to prune the search space further. These strategies are discussed in the following section.

Clearly there are times when information at one level can and does guide the interpretation of a lower-level unit. Lieberman (1963), for example, found that speakers pronounce words less clearly when they are more predictable from their linguistic or pragmatic contexts. And accordingly, listeners process acoustic information less carefully when dealing with predictable words. Marslen-Wilson and Welsh (1978) found a significant effect of contextual constraint on the rate of fluent restorations of excised phonemes. Furthermore, the same acoustic information can be perceived differently depending on its higher-level context. Ganong (1980) found that the identification of a phoneme on a continuum between /k/ and /g/ varied according to whether it was the first phoneme of *kiss* or *gift*.

It is an open question whether this higher-level information *dictates* the lower-level interpretation (as in HARPY, HWIM and Hearsay-II) or *enhances* it through some sort of feedback mechanism (as in TRACE), or *selects* it from a pool of candidates. I shall return to this question in Chapter 3.

2.3. Deciding where to go now

In the previous section we concentrated on two axes of the search space: time and level of abstraction. We were concerned with the *access* and *integration* over time of information from *different* levels during the extension of individual hypotheses. In this section we are concerned with the integration over time of information which allows the *discrimination* of competing hypotheses at the *same* level. With reference to Fig. 1.1 we are looking at the search space bounded by the x and the z axis.

The top-down prediction of hypotheses to be matched can be a way of eliminating large parts of the potential search space altogether. The remainder of the search space can be ordered by using scoring techniques to rank competing hypotheses and to schedule the exploration of the possible extensions. The problem in devising a search strategy is to discover what information justifies focusing on just some of the many possible interpretations, and then to decide how that information can be obtained and used. We shall look at these issues in greater detail in the final chapters of the thesis. The following section give an overview of each system's approach.

HARPY

HARPY uses a form of *breadth-first* search based on a dynamic programming technique called the *Viterbi algorithm* (Viterbi 1964). Certain structural aspects of HARPY's finite-state network allows many hypotheses to be eliminated after a few words. Any partial hypotheses which end at the same node are equivalent as far as further right context is concerned, and only the highest scoring need be kept.

In addition, HARPY limited the average number of competitors over a stretch of speech and also constrained their identity, thereby ensuring that the items were easily discriminable as measured by their acoustic match scores.

However, the search still proved too expensive, even given the constraints of the data and the knowledge graph, and only a band or 'beam' of hypotheses was pursued in parallel through the graph. The finite state grammar allowed the width of the beam to be calculated with very little chance that the correct hypothesis would be eliminated.

HWIM

HWIM experimented with several strategies (Woods 1982), some designed to process left-to-right, others designed to work with island-driving strategies. Unlike HARPY, all the strategies fall into the category of algorithm known as *best-first*. That is to say, they attempt to follow up the best scoring hypothesis depth-first and back-track to earlier interpretations only if the current interpretation appears to be failing.

The *shortfall algorithm*, which worked left-to-right, kept abandoning paths and backtracking to shorter partial interpretations. It was excessively breadth-first and failed to find a solution in all ten trials. The *shortfall density algorithms*, which worked outwards from seed words, were more successful, finding the correct interpretation in half of the ten trials. The remaining test utterances were not *incorrectly* identified; the system simply generated too many partial interpretations and so failed to give a response within the resource limits of the trial.

Paxton's SRI trials (Paxton 1977) found that island-driving improved performance for shorter sentences but decreased it for longer ones. Woods does not believe this invalidates the island-driving approach, however. He suggests that the addition of various features used in HWIM would be sufficient to improve the overall performance of Paxton's island-

driving strategies. HWIM and HARPY's search algorithms are discussed in detail in Chapter 7.

Hearsay-II

Hearsay-II's search strategies are highly complex, involving heuristics representing such global properties as:

> *the competition principle*: the best of any local alternatives should be performed first.
>
> *the validity principle*: KSs operating on the most valid data should be executed first.
>
> *the significance principle*: those KSs whose responses are most important should be executed first.
>
> *the efficiency principle*: those KSs which perform most reliably and inexpensively should be executed first.
>
> *the goal-satisfaction principle:* those KSs whose responses are most likely to satisfy processing goals should be executed first.

(Hayes-Roth & Lesser 1977).

The two main strategies tested were phrase-specific (P) and word-specific (W). P was designed to be more depth-first. If the quality of bottom-up information was good, the algorithm could quickly home in on the correct sequence of words. However, if bottom-up recognition was poor, P could easily be misled by extraneous words into following incorrect hypotheses. The W algorithm was designed to be more breadth-first, and more consistent results were expected regardless of the input conditions. The designers found that W performed significantly better than P. They write:

> "A significant amount of tuning of the focusing parameters has been attempted. Nevertheless, the current parameter values are probably not optimal, and it seems clearly impossible to determine what the optimal values are. In addition, owing to the interesting relationships between the desirability of breadth- or depth-first searches and the specific performance characteristics of the particular KSs used in the system, no absolute conclusions are warranted. Only the general focusing problem and our suggested general approaches appear universally valid; statements regarding the validity of particular parameter settings must await major breakthroughs in the development of our mathematical models and analytical techniques."(p 34)

I tentatively infer from the paper cited above that the strategy which worked best for Hearsay-II -- the W algorithm -- explored the search space in a manner very similar to HARPY's beam search, particularly as the same finite-state grammar was used in both cases.

TRACE II

As we saw in the previous section, a great problem for rule-based hypothesize-and-test systems is the difficulty of matching a higher-level description to a partially determined representation of the input. Such systems, working strictly left to right, match unit by unit. Subsequent matches depend on where one has got to in the description. This makes recovery, especially from word initial errors, extremely difficult, since the matching process has no way of realigning itself correctly with the input. This led HWIM and HEARSAY-II to relax the matching constraints, and to try multi-key lexical access and island-driving strategies.

In TRACE II, recovery is possible through an anti-clockwise circle of excitation. Information to the right of an errorful or missing region could still activate a higher-level unit through its own bottom-up excitatory links. This activation of a hypothesis through connections to any part of it is enough in itself to recover the missing information, but TRACE II also has feedback from the higher level which can increase the activity of all the

lower level descriptions which support it. TRACE II thus combines "loose" left to right processing with a form of island driving limited to the length of the spanning higher level-units.

In addition, inhibitory links between competitors at the same level allow better scoring, more active nodes to depress any competitors.

Although this is bought through a massive number of connections it seems to have a number of advantages over previously mentioned strategies:

1) It should have fewer problems with extraneous words than island-driving strategies since it "prefers" to work left to right. For example, it would find *disk* in *discovery*, but not *four* in *California.*

2) Rightwards flowing information predominates only where left context has proved inadequate for disambiguation. In island-driving left context is unavailable, while in the left-to-right strategies discussed previously right context is only available if some match, even if only a poor one, has been made to the beginning of the unit in question.

3) HWIM and HEARSAY-II have only rough measures of the usefulness of the right context, while in TRACE II the amount of feedback is tightly controlled by the number and activity of the higher-level units. The tighter the constraint provided by the context, the greater the excitations and feedback at the higher level. For example, an ambiguous stop segment at the beginning of /ip/ would be less likely to be identified as /d/ than if it were at the beginning of /im/. The excitation of /ip/ would be shared out between *tip, dip, pip*, and *kip*, whereas in the latter case, *dim* would receive most, if not all, of the activation.

2.3.1 Conclusions

The problem each system faced was that the matching process created a large number of partial solutions, and various heuristics had to be used to cut down the search space. Hypotheses were scored on their goodness of fit to the input. But frequently the correct partial solution scored less well than other incorrect interpretations over the same stretch of sound, though later information was able to raise the score of the path that included the correct partial solution. This is what we have termed a right context effect. The pruning methods therefore had to cut out a band that was neither too narrow (and thus pruned the correct solution) nor too broad (resulting in a combinatorial explosion of hypotheses).

The systems essentially devised two ways of incorporating these right context effects.

1) All systems delay identification, thus giving later information time to raise the score of earlier portions of a unit.

2) In addition, HWIM and Hearsay-II can use the right context provided by islands.

While poor segmenting and labelling contributed greatly to the problem, there is psycholinguistic evidence that people can and do use later information to recover from errors and ambiguities in the input. (Ganong, 1980; Warren & Sherman, 1974). This evidence shows that not only does the identity of a particular unit constrain what follows that unit, it in turn is modified by subsequent higher-level information. Therefore some time must elapse before a unit's identity is fixed. It is not clear how long this delay should be. If it is too long it could lead to an explosion of potential interpretations.

The scores used by the computational systems discriminate between hypotheses in two ways. Firstly, they are used in *focusing* the search on promising hypotheses, but

ultimately they are needed for *choosing* the final interpretation of a unit. The system must contain enough information for a *single* solution to be found.

2.4. Representing and Resolving Ambiguity

We have seen that each system was forced to delay making a decision about the identity of a stretch of sound. Indeed, HARPY, HWIM and HEARSAY-II did not make final decisions about hypotheses until the end of the utterance.

Language is made up of units that may be repeated sequentially (e.g. *papa*) or combined recursively (e.g. *Bill saw John in the car*) and so a way of recording competing interpretations is needed which distinguishes between different tokens of the same unit.

HWIM, Hearsay-II, and HARPY, examples of the Symbol Processing paradigm, can construct representations of the processing that has been done so far and operate on the individual hypotheses within that space. The central processor then orders hypotheses, adds, deletes and modifies them.

TRACE II, on the other hand, belongs to the Parallel-Distributed Processing paradigm. The nodes in the network provide both the knowledge representation and the process by which knowledge is applied. A major difficulty with this approach is that structural relations cannot be represented, and so the type/token relationship of items in permanent memory and in working memory cannot be represented. The TRACE II designers had to duplicate the entire knowledge network over and over in order to represent the time course of speech processing. This was only feasible because of the smallness of the lexicon and the shortness of the phrases processed.

The kind of representation used is very important, since different representations will make different aspects of the search space more or less apparent.

HARPY

HARPY takes the types represented by its finite state grammar and compiles out all the tokens -- the possible utterances in the system -- ahead of time. When processing a particular utterance HARPY creates a search tree whose branches are consistent with the connections in the knowledge graph. The search tree is just a sub-tree of the knowledge graph with scores on its branches reflecting the goodness of the match between the input data and the branch description.

HWIM

HWIM was concerned with developing a fixed interconnection of components within which different control strategies could be tested. The system does not use a single data structure to represent the search. As the flow of information was fixed, data structures could be tailored to the specific knowledge bases using them. For example, at the lowest level a phoneme lattice was produced for processing by the lexical access component. Lexical access in turn produces a list of word matches over some portion of the utterance which are ordered by score.

A major drawback with this design strategy was that testing of the complete system had to wait until all the components and their interfaces had been completed.

In addition, the separation of data structures made it difficult to follow the interactions between the knowledge bases in the pursuit of a hypothesis. It was hard to tell, for example, whether the acoustic-phonetic component, the lexical component or the syntactic/semantic component was responsible for the elimination of a correct word.

Hearsay-II

Hearsay-II's blackboard architecture was designed to allow experimentation with the number and type of knowledge sources, as well as with strategies for controlling the knowledge processing. Compatibility between components was ensured by using a structurally uniform global database, the blackboard.

The designers chose to use a production rule format for communication between knowledge sources; a pre-determined stimulus would provoke a particular response. Mostow & Hayes-Roth (78) found that this formalism was inappropriate for many speech tasks. In a review of Hearsay-II they write:

> "... while the uniformity and lack of explicit organization of production systems are touted as their most desirable features, attendant difficulties of dynamically organising and controlling coherent problem solutions must be seriously considered in problem domains requiring careful allocation of computational resources." (p 471)

However, using a global data space does have certain advantages. It makes it easier, in principle, to "see" how many hypotheses are competing for some portion of an utterance, for example, and to decide which are the most promising given the evidence. As we shall see TRACE II takes this principle several steps further. HEARSAY-II, in the sequential central processor paradigm, designed specific strategies to detect and affect such global properties of the search space.

TRACE II

The first incarnation of TRACE II was the COHORT model (Elman and McClelland 1984) In this model current activity was represented in the knowledge structure itself. This caused problems connected with type/token distinctions. For example, the word *cocoa*

would receive twice as much activation as the word *code*, after the input of /k ou/ because the former contains two occurrences of these phonemes.

TRACE-II solution to this problem was to create a bank of phoneme and word detectors in which units were duplicated again and again, but as McClelland & Elman (1986) point out there are numerous objections to such a scheme. They write:-

> "It seems that we need to have things both ways: we need a central representation that plays a role in processing every phoneme and every word and that is subject to learning, retuning and priming. We also need to keep a dynamic trace of the unfolding representation of the speech stream, so that we can continue to accommodate both left and right contextual effects." (p 77)

The most interesting aspect of the trace representation is the use of excitatory and inhibitory links to control dynamically the relative weight given to different kinds of evidence. We mentioned earlier the problem of relaxing phonetic constraints without knowing the lexical context, and whether such relaxation was required. The working memory representation in TRACE can quickly and effectively brings such information to bear. For example, if there was some slight uncertainty about whether an input segment was /jh/ as *legion* or /zh/ as in *lesion* the ambiguity at the lexical level would not be enough to resolve the ambiguity at the phoneme level. One word would not have enough activation to inhibit the activation of its competitor. If, on the other hand, the ambiguous segments were /g/ and /k/, followed by /i s/ *kiss* would probably win out over such competitors as *kick, kitsch*, etc and *give, Gish*, etc.

2.4.1. Conclusions

Why did all the systems find it necessary to use a working memory? The reason was that they could not resolve ambiguity between competing lexical interpretations on a word-by-word basis, and so had to maintain possible interpretations in a representation that was

separate from the lexicon. Even HARPY, which brought its full syntactic and semantic knowledge to bear immediately, could not make a decision on a word-by-word basis.

Such a representation is essential for any adequate model to represent overlapping, ambiguous hypotheses such as *car go/ cargo* (Cole & Jakimik 1980; Nakatani & Dukes 1977). It is also essential when there are local errors in the graph search. If we delay making a decision about an interpretation we can allow later, high-scoring information to pull up the score of the earlier element. This allows recovery from errorful or missing segments (Miller, Heise & Lichten 1951). It permits right context effects of the type reported by Warren & Sherman (1974) where listeners tended to hear /- ii l / differently depending on whether the following phrase was: *--eel of the shoe, of the orange, of the car*, etc.

Psychological models of speech recognition (Cole & Jakimik 1980, Marslen-Wilson 1975) at first concentrated on the way in which sentential context could speed up the process of word recognition in fluent speech. For example, Marslen-Wilson & Welsh (1978) contrasting isolated word recognition with recognition in context (p 56ff.) show no awareness of the problems introduced by word boundary ambiguity.

Similarly Cole & Jakimik (1980) while pointing out the ambiguity of /p l a n s @ m/ (*plant some/plan some*) assume that top-down constraints will always be sufficient to disambiguate on a word-by-word basis. They assume that *plant some* will be preferred over *plan some* in the context *Tell the gardener...* because planting and gardeners are semantically related.

If this were the case there would be no need for a working memory. If recognition proceeded on a word-by-word basis we would know the beginning and end points of each word, and recognition could indeed take place entirely within the lexicon as these earlier models seem to assume.

However, the Pollack & Pickett experiments (1963) cited earlier seem to suggest that word-by word recognition is not the norm. More recent work (Grosjean 1985; Shillcock, Altmann & Bard 1987) has concentrated on late recognition of words and their implications for models of lexical access.

Whereas one would expect the lexicon to be organised in a way that facilitates the *access* of items, one would expect working memory to be organised in a way that facilitated the *discrimination* of competing hypotheses. The TRACE system is particularly interesting from this point of view.

2.5. Deciding when to stop

When involved in any kind of search we obviously need to be able to recognise what we are looking for, the goal state. Without a terminating condition the search could continue indefinitely or at least until the space of hypotheses was exhausted. All the systems used scores of various kinds in order to choose the goal state. The highest interpretation at some point was taken to be the correct answer.

HARPY

HARPY's goal is to find the path through the knowledge graph which best matches the input. 'Goodness of fit' is measured by matching the spectral characteristics of each segment to the templates in the graph, and at each point marking the template with the highest acoustic match probability. At the end of the utterance HARPY traces the optimal path back through the tree. The main reason for the backwards trace is that decisions about the optimal labelling on the forward pass might prove to be local maxima. There may be an overlapping path which later proves to be a better choice. Breadth-first search is

admissible, i.e. guaranteed to find the highest scoring path through the graph. This guarantee is lost when paths are eliminated during beam search. However, the marked path is still the highest scoring *of the paths considered,* and it can be shown that, if all the paths had been considered, it would still have a high probability of being the highest scoring.

HWIM

HWIM, like HARPY, was concerned that the first spanning theory returned by the linguistic component should be the highest scoring, and much research was devoted to devising scoring and scheduling techniques that guaranteed this result.Woods (82) argues that *admissible* algorithms (i.e. algorithms which give this guarantee), or *near-admissible* ones which relax the constraints in a principled way, are to be preferred over what he calls the ad-hoc, arbitrary strategies used by Hearsay-II. His main argument is that, without the guarantee of admissibility ,there is no obvious reason why the first answer should not be amongst the least probable. With admissible or near-admissible algorithms the search can end as soon as a spanning hypothesis has been found. With non-admissible algorithms the order in which spanning hypotheses are found is unknown, and there is, therefore, no principled way of knowing when to stop.

Hearsay-II

Goodman & Reddy (1980) distinguish explicit control strategies (as used by HARPY or HWIM) from distributed strategies. Distributed strategies, they say, are necessary when knowledge sources are independently activated as in Hearsay-II.

The equivalent of an admissible algorithm in a distributed processing system would be one that was guaranteed to settle into an optimum stable state. This is not the case in

Hearsay-II. As we saw in the previous section, the model is extremely complicated. The various parameters were tuned by hand in order to get the desired behaviour.

When a complete spanning hypothesis has been found, competing hypotheses are rejected (deleted) if they fall below a certain threshold relative to the spanning theory. Other hypotheses are deactivated unless they are the highest scoring in their region in which case processing is allowed to continue. Processing stops when all hypotheses have been rejected, or when time or space limits are reached. Thus it is possible for the system to run out of resources before finding the highest overall spanning hypothesis.

TRACE II

The TRACE II system is an example of distributed parallel processing. While acknowledging their debt to Hearsay-II, they point out a number of differences. Firstly, the blackboard is a passive data structure in so far as it is updated by a central processor, while the TRACE II is active, composed of many, very simple, independent, processing units. Secondly, the communication between knowledge sources (levels of units) consists of fixed connections in TRACE II, while in Hearsay-II the system's KSs can easily be reconfigured. In TRACE II the search process is built into the system through the excitatory and inhibitory connections within and between levels. In Hearsay-II focusing is applied through the activation of a focusing knowledge source.

TRACE II, like Hearsay-II had no guarantee that the correct weights had been chosen but found that the behaviour of the system was very robust under parameter variations.

The TRACE II designers acknowledge that the decision mechanisms have not been fully enough elaborated. Most of their examples show the activation of a single word or a short phrase. They are more interested in the search process than in the goal of the search, and record the rise and fall of activity in competing units over a number of processing

cycles. A particularly interesting feature of TRACE II is the way in which the hypotheses create the "path" through the graph. HWIM and Hearsay-II segment the input and then try to fit the lower level hypotheses into predefined sequences. In TRACE II units at the same level fight amongst themselves for the available supporting evidence. A path is (implicitly) created when a set of adjacent hypotheses dominates all others. Or as the designers would put it, segmentation is a result of recognition.

2.5.1. Conclusions

HARPY, HWIM and Hearsay-II all have, as part of their definition of the goal-state, the condition that the interpretation should span the entire utterance. This condition is not sufficient to guarantee that the correct answer has been found because more than one hypothesis may span the entire utterance. But, used in conjunction with a scoring algorithm, it can guarantee that the first answer is one of the optimal ones. (There may be more than one.) The difference between admissible and inadmissible algorithms in speech processing is discussed in greater detail in Chapter 7.

Obviously people do not wait until the end of the utterance before deciding on an interpretation. That human listeners can find the correct interpretation extremely rapidly is dramatically illustrated in Marslen-Wilson & Welsh's (1978) close shadowing experiments. On the other hand, people will make errors if they are forced to make a decision too early. Grosjean (1985) shows that subjects were uncertain of the identity of infrequent monosyllables until, on average, the end of the following word.

It is not clear what conditions must hold in order to decide upon the word's identity. In a later version of the COHORT theory (Marslen-Wilson 86) writes:-

> "...to discriminate the correct candidate it is not necessary to systematically reduce the cohort to a single member. Selection does not depend on simple presence or absence in the cohort, but on relative goodness of fit to the sensory input." (p 35)

What happens when candidates overlap? Their relative goodness of fit cannot properly be judged because the hypotheses do not cover the same input .

The TRACE II system looks at these issues but in a fairly limited way. The lexicon contains only 211 words and so most items will be phonetically distinct. Secondly, TRACE II deals mainly with isolated words and very short phrases. The paper does not specify how a decision can be made about a word's identity during continuous speech. A major problem in speech processing is determining the point at which all the relevant information has been applied and the answer found.

2.6. Conclusions

We have seen that graph-search terminology provides a useful framework for clarifying and examining issues involved in automatic speech processing.

Firstly, the analysis helped to distinguish those requirements of the task which were influenced by the specific nature of the problem from the more general requirement of searching a large problem space.

Secondly, it provided a common vocabulary for describing a variety of complex systems. We could thus highlight the problems faced by each system, and discuss the extent to which each system's approach was influenced by its architecture. We could show when design decisions were forced by that architecture, and when the systems seemed to be facing common problems. We drew on data from speech science and psycholinguistic experiments to discuss what properties of speech might be causing these processing problems.

Finally, the graph search perspective helped to focus on the dynamic aspects of speech processing. In particular it highlighted the necessity of delaying decisions about interpretations at the word level, due to ambiguities in labelling and segmenting. This raised questions about when (and how) enough information could be applied to distinguish a single hypothesis from competing interpretations. The remainder of this thesis is is devoted to that problem.

Chapter 3. The Architectural Framework

3.1. Introduction

This chapter will discuss the usefulness of the graph-search approach, not just as an analytical tool to be used after the fact, but also as a practical tool to be used in the development of a speech understanding system. I will describe a computational framework, the Active Chart Parser (Kay 1977; Thompson & Ritchie 1984). Since this type of parser is well documented, I will simply outline its main features. My main concern is to illustrate why it is a useful framework for speech recognition research.[1]

I will emphasize the graphical nature of the Chart data structure, and the usefulness of such a structure both for representing linguistic data, and for analyzing the results of linguistic processing. I will also discuss the main attributes of the chart parsing *process* with reference to the dynamics of speech recognition.

3.2 Architectural Requirements

Firstly, perhaps the most important lesson to be learned from the ARPA project was that the system architecture should impose no a priori constraints on the development of individual components. Speech processing is so complex and so little understood that we want as few assumptions built into the development architecture as possible. As we saw in

[1] This chapter is based on my contribution to a paper by Johnstone & Altmann (1984). The paper later formed part of the ALVEY speech demonstrator proposal.

the previous chapter, properties of the blackboard model developed for HEARSAY-II turned out to be incompatible with certain characteristics of the speech processing task.

Secondly, we would also like the system architecture to be an already well-understood computational tool. Designing, implementing and understanding a brand new architecture, such as the blackboard system, is complicated enough without trying to solve the problems of speech processing at the same time. And we would like the architecture to be as simple as possible, the idea being that such a self-effacing architecture would help lay bare the problems of speech processing.

Another important lesson from the ARPA project was that the task of controlling the interactions *between* the knowledge bases is at least as problematic as that of *defining* the knowledge bases. Psychological studies (e.g. Marslen-Wilson & Tyler 1980) have shown very close interactions between different types of linguistic knowledge in speech decoding, though the nature of these interactions is still obscure. An architecture is therefore required which will allow flexible and easily visible control over these interfaces.

Finally, the architecture should permit the parallel and fairly independent development of different component knowledge bases and methods of deploying them computationally. This would help ensure that the design of one component would not unduly influence the design of another. It would also allow individual components to be tested, using simulated data, before the entire system is complete. According to Woods et al (1976), the HWIM system, with its individually tailored interfaces, ran into problems because of this kind of delay.

3.3. The Chart Parser

An architecture which seemed to fulfill these requirements was the Chart parsing system (Thompson and Ritchie 1984). Firstly, the chart parser was an existing

framework, originally developed for use in the syntactic domain, but also used in other areas of the speech chain, e.g. acoustic-phonetic analysis (Church 1983) It was specifically designed for automatically building a graph of possibilities, these possibilities being determined by the component knowledge sources, not by the parser. It thus posed few a priori constraints on the individual knowledge bases.

Secondly, the components of the Chart parser are few and relatively simple:

1) A uniform global data structure (the Chart).

2) A multi-level task queuing structure (the Agenda).

3) An algorithm for automatically scheduling additions to the Chart onto the Agenda for subsequent processing (the Fundamental Rule).

This economic and fairly sparse architecture fulfilled the need for simplicity mentioned above. Yet it had also already proven to be a powerful and flexible tool for tackling complex system building tasks as set out in Bobrow et al (1977).

Thirdly, the existence of a global data structure allowed the interfaces between components to be specified in an orderly manner. The global data structure, the Chart, would provide an easily accessible record of what exactly was going on between the different components. The Agenda could be used to test various scheduling strategies. The Fundamental Rule could in principle allow any component to interact with any other component.

Finally, the Chart parser permitted the implementation (both in serial and in parallel) of different rule systems, and the evaluation of strategies for using these rule systems. Since all components communicated via the global data structure, individual components could easily be designed and tested using simulated data. Further components could be added whenever their stage of development warranted it.

The graph-search terminology which proved useful in Chapter 1 could easily be applied during testing of this system. Since the Chart is just a recursive, directed, labelled graph, we could use graph-based techniques to probe the extent of the search problem. The Chart could represent the input, output, and intermediate results of the parsing process. It could, in principle, compute all possible relationships between different parts and subparts of an utterance, and also provide a record of failed interpretations.

The Chart parser thus fulfills the architectural requirements outlined above and, in addition, reflects the paring down of the graph-searching task to its barest requirements with few restrictions on how the graph should be constructed and explored.

3.4. Details of the Chart System.

3.4.1 The Nature of the Graph

The Chart data structure (see Fig. 3.1) is used to represent and extend pathways through time and level of abstraction through a search space. It consists of a set of *vertices*, which may be thought of as marking off temporal units along the x axis, and a set of *edges* linking these vertices. This graph of vertices and edges is acyclic and directed. Each vertex may have a number of edges emanating from it. These edges can be considered mutually exclusive interpretations of some stretch of the utterance defined by the z and x axis. Each edge is labelled and can carry whatever information is needed for the parsing task.

Within the Chart there can be different types of path corresponding to different levels of abstraction (the y axis), each of which is associated with a particular knowledge source (i.e. acoustic-phonetic, phonemic, morphemic, syntactic, etc). New pathways, giving a

different level of description by spanning existing constituent pathways, can be added according to the knowledge bases' rules.

One advantage of the Chart is that it allows one to represent both complete analyses (as *inactive* edges) and partial analyses (as *active* edges).

Inactive edges may span, and hence have pointers to, supporting lower level inactive edges. For instance, a syntactic edge may span a number of lexical edges, each of which may span a number of phonemic edges, and so on.

Active edges, on the other hand carry with them a specification of the supporting edges which they need, but as yet have not found, in order to become complete. Thus they carry a specification of what kind of inactive edge they will become on completion, what kinds of lower level inactive edges they require in order to become complete, and just which inactive edges constitute the partial analysis derived so far. The structure of the chart can be fully defined by four functions[2]:

EDGESET (vertex)	= edgeset-of-vertex
FIRSTEDGE (list)	= first-edge-in-list
EDGEALT (list)	= rest-of-list
GETI (edge label)	= value of label

For any vertex, EDGESET returns the list of all edges originating from that vertex. FIRSTEDGE and EDGEALT return for each edgeset the first edge in the list and the remaining edges in the list respectively. GETI gives access to information contained on any edge. (Active and inactive edge sets are maintained separately.)

A typical portion of a Chart showing vertices, active and inactive edges and different levels of description is given in Fig. 3.1.

---

[2] This definition is taken from Varile (1983).

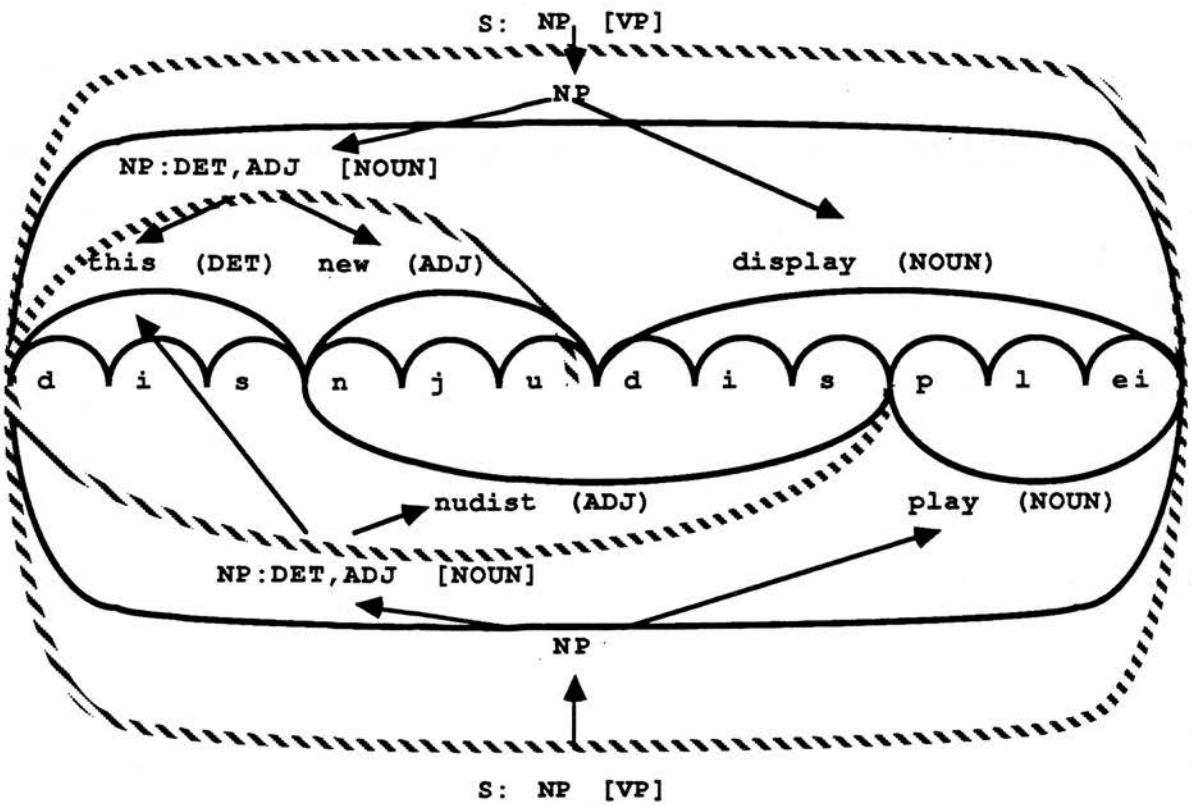

Figure 3.1

A simplified Chart representation of a portion of an utterance which is interpretable either as *this new display* or as *this nudist play*. Note that the /t/ in *nudist* would tend to be omitted when followed by a stop consonant such as /p/ - hence permitting the above ambiguity. Complete hypotheses (inactive edges) are represented as continuous lines. Phonemic edges are labelled with the corresponding phonemic character. Lexical edges are labelled with the orthographic form, and in brackets the syntactic label associated with the hypothesized word. The labels on syntactic edges correspond to a syntactic category (DET - determiner; ADJ -

adjective; NOUN - noun; NP - nounphrase; VP - verbphrase; S - sentence). Partial hypotheses (active edges) are represented as dotted lines. They carry on them a specification of what is hypothesized (e.g. NP), what information currently supports this hypothesis (e.g. DET ADJ) and what information is needed to confirm this hypothesis (e.g. NOUN). The arrows represent pointers connecting a spanning edge with those edges which support it. Normally, edges would have associated with them a confidence score; however for sake of clarity these have been omitted as have been some of the edges which would otherwise have been created.

The Chart is an economical structure since common subparts are shared by competing higher level interpretations. (i.e. the first word of the two nounphrases in the diagram above.) In this respect it can be related to the idea of a well-formed substring table which records all the legal subparts of a sequence of symbols. It is also economical in terms of processing since partial interpretations are computed only once, and can be returned to if an alternative interpretation fails further down the line, thus guaranteeing that all possible parses are found.

Another consideration is that preserving such partial interpretations allows us to see where the recognition process went wrong, either in failing to pursue the correct interpretation, or in abandoning the correct one for some other interpretation. This is a necessity given the ambiguous and errorful nature of speech. We shall see how useful this is in Chapters 4, 5 and 6 where the Chart is used to identify word strings from various kinds of phonemic input.

In choosing the Chart system which allows various models and strategies to be implemented and evaluated, we ensure that no design decision at any level in the system is irrevocable. Thus at the acoustic-phonetic level the Chart could represent, for example, a

segment lattice, which as the project progressed, could contain segmentation and labelling information of increasing complexity. And at the syntactic level we are not committed to any one particular formalism, but rather can consider a number of different formalisms. Since the input and output of each component is defined (though not constrained) by the Chart data structure, such independent design and interfacing is made much more simple.

3.4.2 Searching the Graph

Having described what the Chart *qua* graph looks like we must now consider its properties as a process. As I mentioned above, the Chart parser has two other components in addition to the global data structure: a task queuing structure, the Agenda, and an algorithm for automatically extending interpretations on the Chart, the Fundamental Rule. The Fundamental Rule is defined by Thompson & Ritchie (1984) as follows:

> "Whenever the far end of an active edge *A* and the near end of an inactive edge *I* meet for the first time, if *I* satisfies *A*'s conditions for extension, then build a new edge as follows:
>
> -- Its near end is the near end of *A*
> -- Its far end is the far end of *I*
> -- Its contents are a function (dependent on the grammatical formalism employed) of the contents of *A* and the category and contents of *I*
> -- It is inactive or active depending on whether this extension completes *A* or not."

Note that the Rule does not presuppose a left-to-right strategy. The terms *near* and *far* are used to emphasize the direction-independent nature of the Rule. This is important in speech processing for reasons discussed in section 3.4.2.1. One can also choose between a top-down or hypothesis-driven strategy and a bottom-up or data-driven strategy depending on whether the rule expansion is driven by the addition of active edges or inactive edges. The relevance of this to speech processing is discussed in section 3.4.2.2.

Finally, the rule says nothing about the order in which hypotheses are to be pursued. The placing of additional edges onto an Agenda means that the search can proceed depth-first, breadth-first or according to some more complicated algorithm that might, for example, take into account the scores of different hypotheses. This is discussed in section 3.4.2.3. Sections 3.4.2.4 and 3.4.2.5 address the handling of ambiguity. The fact that the active edge in the Rule above is not modified means that, in principle, all interpretations will be found regardless of the order in which operations are carried out.

In Chapter 2 I gave a breakdown of the five minimum requirements of the graph search mechanism. These were:

1) Deciding where in the graph we should start.

2) Deciding where to go next. That is, rules or descriptions of some sort defining valid paths.

3) Deciding where to go at any particular point in processing.

4) Recording processing to date in order to represent and resolve competing interpretations which may be disambiguated by later occurring information.

5) Deciding when to stop.

I will now discuss the Chart parsing process with reference to these requirements. We shall see that the Chart does not force a commitment in any of the five areas of choice. It is as flexible, in terms of processing and information flow, as the blackboard system used in Hearsay-II. But unlike Hearsay-II's production system framework it does not impose any constraints on the way knowledge is represented. I shall also begin to sketch the *model* that will be implemented within the Chart architecture. In each section a decision will be made about the parameters of the model based on the experience of the systems analysed in Chapter 2.

3.4.2.1 Where to begin

The first decision to be made is whether to add and extend paths strictly from left-to-right through the utterance; or whether to allow a middle-out strategy that extends hypotheses both to the left and to the right of some island of comparative certainty; or whether to implement a combination of these. The chart framework permits any of these strategies to be implemented. This is accomplished through the ordering of edges on the Agenda, and through the direction-independent nature of the Fundamental Rule which looks for possible extensions to hypotheses.

Both HWIM and Hearsay-II support a middle-out strategy, while the other systems in the ARPA project opted for strictly left-to-right processing, variously constrained to follow only valid sequences in a specified network. Strict left-to-right strategies, while intuitively similar to the way we perceive our recognition of fluent speech, force computational systems to tackle immediately stretches of sound, which are of poor acoustic quality, or which are relatively unconstrained by higher level knowledge. Systems such as HARPY have tried to alleviate these problems by relying heavily on the properties of finite state grammars, applying all higher level knowledge constraints simultaneously through pre-compiled knowledge networks, and delaying decisions about the correct acoustic-phonetic interpretation until the end of the utterance. Middle-out strategies, which have been used with more powerful grammars, have an advantage over strict left-to-right strategies in that they can use areas of better acoustic quality as islands of comparative certainty from which to tackle areas of poorer quality. They can also, in principle, build up more global analyses at different points in the utterance, and thus use syntactic and semantic constraints from the right, as well as from the left, in the analysis of uncertain areas. On the other hand, middle-out strategies are computationally explosive methods of search requiring highly

tuned scoring methods and matching strategies to constrain the exploration of all the myriad combinations of possible hypotheses.

Much of the power of middle-out strategies derives from the quickness with which an analysis of some stretch of sound has access to information in a later stretch of sound. Furthermore, experimental data suggest that the human speech processing mechanism also makes use of later occurring information in analyzing earlier parts of sentences (e.g. Pollack & Pickett 1963; Warren & Warren 1970; Nakatani & Dukes 1977). The HARPY strategy can amend low level descriptions in view of right context effects, but only in the final backwards search after the entire utterance has been processed left-to-right. HWIM or Hearsay-II on the other hand can proceed leftwards from any island of certainty. In HWIM, however the best results, in terms of efficiency, seemed to be obtained using a predominantly left-to-right strategy with middle-out analysis permitted only on the initial portion of the utterance where the acoustic material is generally less well specified.

We decided to use a left-to-right strategy in the experiments described in Chapters 4, 5 and 6. The results were then analyzed to see where this approach was causing problems and whether a more flexible approach was needed. The data concerning word boundary ambiguity raised considerable doubts about the efficacy of island-driving approaches.

3.4.2.2 Where to go next

The type and complexity of descriptions specifying valid paths is determined by the knowledge bases defined for the task and not by the parser. For example, in the Chart diagram in Fig 3.1., valid paths at the syntactic level are defined as groups of syntactic categories. This could be expressed in terms of a formal grammar. A path not described by those rules would be rejected. An alternative definition of valid paths used in the ALVEY project's RM1 was based on pair-wise combinations of syntactically tagged

words. Each combination was given a probability score based on its occurrence in a large corpus. In this case, any path is "valid", though more or less probable.

We are also concerned here with the interactions between components. The validity of a particular string is usually judged within the context of a higher level knowledge source. For example, the validity of a string of phonemes depends on what is in the lexical data base. A word, or perhaps just a certain pronunciation of a word, may not be contained in the lexicon. The flow of information between a lower level and a higher level can be handled in a number of different ways which will have markedly different consequences for the speech recognition task. A system may permit *strong interactions* or *weak interactions* between knowledge sources. With the latter, the only permissible interaction involves the filtering out, by one component, of alternatives proposed by other components, so in hierarchical terms, no component determines what is produced by any other component beneath it. A strong interaction, on the other hand, allows one component actively to direct, or guide a second component in the pursuit of a particular hypothesis.

As we saw in the previous chapter, HARPY, HWIM and Hearsay-II relied heavily on strong interactions. Unlike HARPY, HWIM and Hearsay-II were faced with the problem of specifying explicit schemes for controlling the flow of information between knowledge sources. This was a highly complex and difficult task. (Reddy & Ermann 1975; Goodman & Reddy 1980)

Within the psychological literature there has been a growing tendency away from strong interactions towards weak interactions. Marslen-Wilson (1986), describing the evolution of his COHORT model of speech recognition, writes:

> "Early statements of the model (e.g., Marslen-Wilson & Welsh (1978)) assert that candidates drop out of the pool of word-candidates when they do not fit the specifications of context, in the same way as when they do not fit the accumulating sensory input. This runs into similar problems to the all-or-none assumptions about sensory matching that I have just discussed. For the sensory input, the problem was to explain how mispronounced, or otherwise

> deviant words could nonetheless still be correctly identified. For context, the problem is to explain how contextually anomalous words can be identified (e.g., Norris, 1981).... The implication of this is that context does not function to exclude candidates from the cohort. There is no all-or-none matching with context, and no all-or-none inclusion or exclusion of candidates on this basis. This parallels the points made earlier ... prohibiting top-down influences upon initial access. It looks as if contextual factors can neither determine which candidates can enter the cohort, nor which candidates must leave it."

A weak interaction between knowledge sources necessarily gives a *hierarchical* flow of information from one level of description to the next, as activation proceeds bottom-up through the system. This is by far the easiest model of information flow to control. Standard hierarchical models, however, allow too little interaction between the knowledge sources: within a strictly hierarchical system, one cannot interleave the processes associated with each different level of knowledge, and hence one cannot allow the very early filtering out by higher-level components of what might only be partial analyses at lower levels. This situation (considered disadvantageous for reasons of speed and efficiency) arises because of the lack of any common workspace over which the separate components can operate.

The Chart-based model used in the following experiments can be considered a hybrid between the blackboard and the hierarchical models. Unlike the blackboard model, it embodies only weak interactions, whilst unlike the hierarchical model it uses a uniform global data structure.

Alternative pathways at one level of description can be filtered hierarchically through attempts to build pathways at the next higher level. This filtering process can be applied as soon as data becomes available if necessary, since all components are using the Chart to post results.

The problems of *defining* valid word strings will be discussed in detail in the following chapter, as will the problem of finding such word strings from underspecified or errorful data. The feedback model (i.e. TRACE II) is closer to the weak interaction type in that it is

primarily data driven, rather than hypothesis driven, but it does allow some top-down filling in of gaps, and some correction of errors in the input data. The Chart model has a far larger lexicon, however, containing 4,000 lexical items. We were interested to see how the hierarchical model would scale up, given better input data than that used in the ARPA project.

3.4.2.3 Where to go now

Rather than explore all the possible pathways we need some method of ordering the search. In terms of the Chart this means controlling the order in which hypotheses are taken off the Agenda and added to the search space. The ordering of the Agenda can be done in any number of ways, breadth first, depth-first or, in most instances, based on scores of some kind.

In the ARPA project, most of the search strategies relied on properties of finite state grammars to limit the search. A drawback here is that such processes are limited in terms of the power of the grammars they permit. In a less constrained system, such as HWIM or Hearsay-II, the task is to find some method of reflecting the "goodness" of a path, according to the various knowledge sources which contribute to it. A number of issues arise in deciding how to combine such priority scores and how to use them to pursue a complete interpretation.

One of the most important factors is the reliability of the individual sources which contribute to the search space, and particularly the performance of the components involved in lower level analysis of the speech waveform. In Chapter 4 I discuss the performance of the lexical access component based on real input from the RM1 front-end processor. In Chapter 5 and 6 I show the extent of the search problem given (i) perfect acoustic-phonetic input, and (ii) error-free but underspecified input.

The problem in designing an optimal search strategy is both how to combine scores across paths representing different levels of description (derived from the component knowledge sources), and how to combine scores across time (during the extension of a path), such that promising paths are given higher priority.

When processing through time, choices have to be made about how good a hypothesis looks now. But what happens if it fails to fulfill its initial promise? What happens if a hypothesis which looked poor initially benefits from later right context information?

When processing across levels of description, choices have to be made about the relative contributions of each knowledge source. Should top-down predictability affect the score of a hypothesis directly? Should it at times carry more weight than bottom-up acoustic quality? Or should the higher levels simply filter out invalid strings? It is possible that the higher level knowledge sources will have to contribute to the scores of the paths being extended since, at present, no psychological or computational model of bottom-up analysis is powerful enough to guarantee correct recognition based on acoustic quality alone.

3.4.2.4 Representing and Resolving Ambiguity

The Chart provides a uniform global data structure, thus making information about the current state of processing easily accessible to, and modifiable by, any of the knowledge sources used in processing. The amount of information on each edge is determined by how much information a higher level needs in order to decide between competitors. For example, suppose the syntactic component used the pair-wise parsing process mentioned above to decide between different lexical hypotheses. Each lexical hypothesis would carry its current score and its syntactic tag. The syntactic component could look at all the word-pairs for each tag ending at some vertex in the graph and discard all but the highest scoring,

based on the acoustic scores and the pair-wise probabilities, since, given this parsing method, no later information is relevant to the decision. A more complex syntactic component might need both more grammatical information and more time to make a decision. These, too, could easily be represented on the Chart.

A further argument for a graph-based analysis of the issues, as well as for the using the Chart system specifically, is that, although the Chart was designed for use with rule-based systems, it has been used as the basis of a connectionist model of parsing (Waltz & Pollack 1985). The edges in the graph output by the parser can be linked by excitatory and inhibitory connections permitting a connection-activation resolution of competing interpretations.

3.4.2.5 When to stop

Unless some task-specific terminating condition has been specified the parser will automatically explore every possible pathway determined by the knowledge sources, the input, and the procedures for constructing extensions to interpretations. In Chapters 5 and 6 the Chart is allowed to run to completion in order to determine the worst-case effects of lexical ambiguity. It is important to know just how difficult a task the higher levels are facing. One of the main achievements of this thesis is to show that, even with very good input, the search space of possible word strings can range from the large to the impossibly large. The consequences of limiting the search are discussed in Chapter 7.

3.5 Conclusions

In summary, the main features of the architecture are as follows: the problem is viewed as one of directed search of a graph, this search space being contained within a single

global data structure, the Chart. Despite the adoption of a global data-structure, an essentially hierarchical flow of information is imposed which allows for a weak interaction between the knowledge sources. This interaction is under the control of the Chart parser: an existing framework which allows the exploration of various search strategies and input conditions. The benefits and disadvantages of such a model will be explored in detail in the following chapters, which deal with the development and evaluation of the lexical access component of the system.

The general and flexible nature of the framework ensures that no design decisions made early on in the course of development are irrevocable at a later date. It also ensures that the development of each of the system's components can be pursued in parallel, no one stream of research imposing any a priori constraints on any other. In chapter 4, the lexical access component is tested against input from the RM1 front-end, and in chapters 5 and 6 it is tested against simulated data.

The graphical nature of the Chart allows *all* the results (complete and partial paths) to be represented in a perspicuous manner. Studying the graphs helps us analyze the performance of the component, the problems the component faces given certain types of speech input, and the problems the component will present to higher levels of processing.

Chapter 4. A Chart-based Lexical Access Component

4.1. Introduction

The first part of this chapter will discuss in more detail the three main problems of lexical access that were outlined in sections 2 and 3 of Chapter 2. These are:

1) accessing hypotheses,
2) discriminating between competing hypotheses,
3) integrating information with other levels of processing.

We will discuss some of the major theoretical problems presented by each of these tasks, referring back to the systems described in Chapter 2. We are concerned with the theoretical and practical motivations behind the development of the lexical access component used here. In developing the component, we drew on the successes and failures of the earlier systems, and incorporated some of their best features. The component is original in its use of the Chart-based architecture described in the previous chapter. However, the primary aim of this research is to explore some of the problems of speech processing, rather than to explore new computational techniques. To that end, the lexicon is four times larger than any used in the ARPA project. Its content is not determined by a particular task domain, nor do we assume much grammatical constraint.

The transparent architecture should allow us to see where problems of scale become overwhelming.

4.2. The Accessing Function

Lexical access in automatic speech processing is central to the process of transforming a description of a physical event into a description which is composed of units of meaning. The representation must allow us to compute a link between descriptions of acoustic input and stored descriptions. E.g. we must be able to access a word such as *environment* from its many and varied phonetic realisations. In addition, the representation must allow us to key into the meaning or function of those descriptions. We must be able to get at all the different ways in which a lexical item transcribed as /m iit t/ is used.

4.2.1 Units of Recognition

It might seem obvious that the units which provide the interface between sounds and meanings should be words. It is clear, however, from the variety of lexicons used in the systems in Chapter 2 that there is still little agreement about how lexical items should be represented.

4.2.1.1. Phonological Variation

One of the major problems is that of phonological variation both within and across word boundaries. While there may be many advantages in terms of space, processing time, productivity, and so on in using abstract units which capture regularities about, say,

derivational and inflectional morphology in English, it is not clear how such knowledge should be used in speech processing. That is, it is not clear how we get from the spoken word to the kind of abstract representations linguists use. Sounds within words may change or be missed out altogether:

actually -> /a k t y u@ l i/ /a k ch u@ l i/ /a sh l ii /

If words were represented as fairly abstract morphemic forms a speech recognition system of the conventional matching type would have to derive these forms somehow from the acoustic input by the application of phonological rules in reverse. The HWIM system experimented briefly with this method (Woods et al 1976), but it soon became clear that such an approach was unrealistic. An example might be a rule, /d y / -> /jh/. This would allow the matching of *did you.*, but the reverse rule /jh/ -> /d h/ would overgenerate if the input was, say, *judge.*

Thus it seemed clear that the lexical representation used for matching should already include a great deal of information about the word's possible phonetic realisations. One could begin with a base-form pronunciation of each word in the dictionary, and then apply phonological rules to generate valid variations in pronunciation. One possibility is to generate this information as needed during processing; most of the systems took the computationally more efficient approach of pre-compilation. This brings us to the problem of phonological recoding within sentence context.

4.2.1.2. Word boundary effects on pronunciation

The phonemic representation of a word will vary considerably with its context. Klatt (1979) gives the following example:-

Would you hit it to Tom
/w uh jh @ h i d i t @ t a m/

1. Palatalization of /d/ before /y/.
2. Reduction of unstressed /u/ to schwa in *you*.
3. Flapping of intervocalic /t/ in *hit it*.
4. Reduction of schwa and devoicing of /u/ in *to*.
5. Reduction of geminate /t/ in *it to*.

Rules 1, 3 and 5 apply across word boundaries. In order to know which rules apply, one would need to know the context in which the word is appearing. But, of course, that is precisely what one is still trying to find out -- what the words are. Is the /jh/ in the input embedded in *would you* or is it the first phoneme of *judge*?

Both HARPY and HWIM essentially solved this problem by removing word boundaries from the lexicon: HARPY by precompiling a network of the possible utterances in the system, HWIM by creating a wrap-around lexicon. Thus the lexicon contained explicit paths which mapped from the end of *would* into the beginning of *you* via /jh/, or from /d/ to the beginning of the lexicon if there was no boundary effect. However, this makes nonsense of the notion of having word *units* stored at this level. The structure now reflects the continuum rather than the discrete units we perceive.

This was not a problem for HARPY since it represented and recognised complete utterances. But HWIM needed to segment words in order to represent and rank competing partial interpretations. A number of different segmentation and scoring strategies were used (Woods et al 1976), but none were entirely satisfactory.

An alternative to encoding word-boundary assimilations in the lexicon itself, is to try to find primitive units which are less sensitive to such effects. This approach was used by HEARSAY-II and will be discussed in the next section.

4.2.2 The Primitives of the Representation

The primitives of the representation are the results of earlier processing which are used to access words or phrases. The primitives must carry enough information to allow us to discriminate between words. (That is to say, even if it were the case that e.g. broad-class syllables were easy to find, it would be pointless to use them for lexical access if the number of words described by a string of such units was huge.) This information must also be readily and reliably available from earlier processing. (E.g. if fine-class phonemic descriptions unambiguously described spoken utterances, it would still be pointless to use them if it took the processor weeks to find them in the acoustic input and half the time it got them wrong.)

Phonemes are obviously sensitive enough to make fine discriminations between lexical descriptions since this is their function by definition. They represent the set of minimal units required to uniquely specify a word, with the exception of homophones. However, precisely because they can make such fine distinctions at the lexical level, they may well prove unstable for the purposes of recognition. A small change in pronunciation by the user, or a small error in earlier processing will have a large effect on the description at the

lexical level. If there are any missing, extra or erroneous segments with respect to the stored representation then access will fail. If for example the result of acoustic front end processing on the word *actually* was out by a single feature, (either through error, or just because the speaker had pronounced it that way), and reported the string /a g ch @ l ii/ the match would fail, even if the information following the /g/ allowed the word actually to be hypothesized with a fair degree of confidence.

Trying to segment the speech signal into phonemes is in fact likely to be a very errorful process for the following reasons.

4.2.2.1. Non-invariance of acoustic-phonetic cues

The acoustic characteristics of a phonetic segment can vary considerably depending on its context. For example, the last segment in French *piques* and *Paques* differ in form but function as the same phoneme in the language. This particular difference does not appear to be a physical requirement of the articulatory system since the last segments in English *peak* and *park* do not show the variation associated with the different vowel environments.

It appears that listeners don't just ignore allophonic variations that "don't matter", they can actually use the context-dependent information to identify phonemes. (See Klatt 1979). As Abercrombie writes,

> "Allophones are not grouped into phonemes by nature, but by the phonology of a particular language." (1967 p.87)

So conversely, the form may be the same or similar, but the function might be different. The acoustic differences between the initial segments in `gift` and *kift* or `giss` and *kiss* distinguish lexical items from non-words. Experimental evidence indicates that if

the acoustic cues specifying these segments are varied along a continuum with /g/ at one end and /k/ at the other, the same ambiguous segment is much more likely to be interpreted as /g/ in the context of /ift/ and as /k/ in the context of /iss/ (Ganong 1980). This evidence is particularly interesting because it shows an effect of right-context information, that is information received *after* the segment in question.

A further complication is that the function of a particular segment may depend on just which interpretations are most likely at that point in processing rather than on the language system as a whole. The function of the initial segments in *gilt* and *kilt* for example is to distinguish between two lexical items. In the sentence context, *Henry isn't Scottish but he wears a* ----- , this function may not be important. The semantic constraint may take precedence over acoustic information.

4.2.2.2. Parallel encoding of acoustic-phonetic cues

The acoustic cues to one segment frequently overlap with cues to other segments. Information necessary to the decoding of a segment may lie outside the arbitrary boundaries imposed for the analysis of that segment, and some of the information within the boundaries may only be relevant to what precedes or follows the segment. If the system makes too early a commitment on segmentation, recovery will be well nigh impossible.

4.2.2.3. Other problems

Some of the most marked differences between two tokens of speech may not be linguistically significant. They may be the result of such factors as speaker variation, temporal variation, or a noisy environment.

Many of these problems seem to have a parallel in vision processing. Marr (1982) writes:-

> "What was wrong with the idea of segmentation? The most obvious flaw seemed to be that "objects" and "desirable regions" were almost never visually primitive constructions and hence could not be recovered ... without additional specialised knowledge. Edges that ought to be significant are either absent from an image or almost so and the strongest changes in an image are often changes in illumination and have nothing to do with meaningful relations in a scene." (p 272)"

It is clear, given all these factors, that too early a commitment to a phoneme's identity would be disastrous. One way of overcoming the problem is to bet on every horse, so to speak, by assigning a score to every phoneme in the system for every possible segmentation. This was the approach taken by the HWIM system. The possible phonemic function of each allophonic description found by HWIM's Acoustic Phonetic Recognizer was scored by looking up in a long term confusion matrix the vector of 71 phoneme labels that could be associated with the segment's feature description. The resulting search space, given so many possible combinations of phonemic labels, is obviously extremely large, and the scores were rarely indicative of the correct phoneme. In HWIM the correct phoneme was included in the top two scores of the vector only 65% of the time.

Another approach is to try to fix the race by removing some of the uncertainty, using primitives which are less informative but more robust. The HEARSAY-II system defines

words in terms of syllable types, which are described as groupings of broad class phonemes. The keys for lexical access are stressed syllables in the word corresponding to the input syllable type. It should be noted, however, that the problems discussed above with reference to phoneme recognition apply to other units such as syllables as well.

As we saw earlier, the designers of TRACE would say that these approaches are based on some fundamental misconceptions about the nature of speech recognition. They would argue that the input should not be segmented into primitive units and then matched against the lexical representation. The segmentation should be a consequence of recognition rather than a means to it. They argue that segmentation at distinct levels of description is bound to be an errorful process given the highly parallel encoding of information in the acoustic waveform. Instead of trying to specify keys to units in advance, access should be achieved through possibly partial descriptions of their content.

TRACE assumes a single pronunciation per word and relies on the activation of features and the inhibition between words and phonemes to recover a word from some variant pronunciation. While this method may work for some differences, it will not be sufficient to account for major variations in a word's pronunciation such as *the* -- /dh @/ and /dh ii/, or *for* -- /f @/ and /f oo r/.

4.3 Discriminating between Hypotheses

4.3.1. Lexical Recognition Point

As Marslen-Wilson (1986) has pointed out, the concept of a recognition point, i.e. the point at which a word becomes discriminable within the language system, cannot be

determined for a word in isolation. The fact that /p/ is the recognition point for *trespass* depends on the knowledge that there are no other words beginning with *tresp* in the language. A representation or process which allows *sets* of word candidates to be considered will allow such early discrimination points to be used as soon as possible, even before the remaining acoustic information about *trespass* is present. There is considerable evidence from psycholinguistic experiments that people can and do make decisions about a word's identity before they have access to its complete phonological specification.

A tree-structured representation of words as phoneme sequences is one way of implementing this discrimination between all the possible words in the language. If the input segments were correctly and uniquely specified then the tree structure could be used to process left to right through the input, gradual eliminating whole sub-trees through the mismatch of a branch of the tree with the acoustic input, thus exploiting the syntagmatic relations of the language system. Eventually, this matching process would ideally reduce the cohort of possibilities to one, whose functional definition is retrieved from the terminal node reached in the tree. This function of the discrimination tree is basically a set-splitting method of looking up words.

4.3.2. The Recognition Point in Continuous Speech

A major problem in lexical access concerns the registration of the input descriptions, whether they be phonemes, diphones or syllables against the representations in the lexicon. There are few if any reliable physical cues to word boundaries. People's cavalier attitude towards word boundaries can be seen in the derivation of the word *tawdry*, the result of syllable merging across a word boundary in *St. Audrey*.

Some psycholinguistic models (e.g. Cole & Jakimik 1980) assume that the beginning of a word is known, either because it is at the beginning of the utterance or because the previous word has been identified. This is unlikely to be the case, partly because of the indeterminacy in the acoustic input that we have just discussed, partly because, as we shall see, a particular string of phonemes may be parsed into a number of different word strings. Thus, the lexical access component may have to match each word against every possible alignment of the input with the lexicon. Even if lexical information can be used to limit subsequent registrations, it will not be done on a word-by-word basis.

The discrimination function inherent in the tree is suitable only for words heard in isolation. In continuous speech the lack of acoustic cues to word boundaries, together with phonological variations in pronunciation, means that many stretches of speech can be parsed into words in more than one way. The partial utterance

/n y uu d i s/

can be interpreted as,

new dis..

nude is..

nudist ...

Furthermore, the above is transcribed into fine-class phonemes and, as we saw in the previous section, we cannot expect the front end to be so accurate, and indeed we may not want it to try.

The set of all labels which the acoustic front end must decide between is the set of all phonemes in the language. At the lexical level the frame of discernment consists of words

expressed as ordered sets of phonemes. We can, and should, be able to make use of the syntagmatic and paradigmatic relations between these ordered sets to constrain possible identities of input phonemes. As discussed above, a more robust recognition system may result from allowing multiple interpretations of a segment, and letting lexical access make the final decision about whether a segment is /s/ or /f/, for example, in the context of / g i / or /k i/, since neither *gis*, nor *kif* are valid lexical sequences. Similarly we might not want any hard and fast decisions about /g/ and /k/ to be made if the subsequent input is /i s/, since *giss* is not a valid word, unlike *kiss*, which differs by only one phonetic feature.

If we ask the acoustic front end to leave some of the labelling to lexical access it will do so with a vengeance since it has no way of knowing what dilemmas need resolving at the lexical level. If we are not to have too few choices at some point, then we are bound to have too many at others. The ability to mark discrimination points is lost since ambiguity of word boundaries in connected speech, together with multiple choices of phoneme labels, means that a variable number of paths through the tree will be pursued more or less in parallel at any particular point in processing. The discrimination function must therefore be removed from the knowledge representation.

In place of the set splitting inherent in the tree, we will require a dynamic memory structure such as the Chart for recording the multiple word tokens under consideration, and a process which is capable both of rating the goodness of fit of numerous possibly overlapping word hypotheses to the current input, and of determining when enough information has been gathered to select one of them.

The ability of speech recognition systems to discriminate words from acoustic information alone is not encouraging. HWIM attempts to find just the *n* highest scoring "seed" words anywhere in the utterance, the remainder being predicted top-down by the syntactic/semantic component using the seed words as starting points. With *n* set to 12

and using a 1097 word dictionary, the word hypothesizer typically finds two correct words bottom-up. The correct word is the highest scoring 65% of the time, and within the top five 85% of the time. The utterances used were about six words long. The "fan out" of possible paths from highly rated, but incorrect word hypotheses is considerable, and we have no easy way of telling when the correct hypothesis has been reached.

The relative strengths of competing hypotheses may be relevant to the decision about how many words to select for higher level processing, but unfortunately sequential hypothesize and test systems favour the collection and evaluation of evidence for a particular hypothesis in isolation from its competitors. It is difficult to obtain an overall view of the search space at any particular time.

Smith and Sambur (1980) who built the NOAH system have suggested that the discrimination function could be improved by categorising words into four sets based on their acoustic discriminability and semantic content or usefulness. Words could then be rated on their a priori distinctness as well as on their goodness of match. Results of psycholinguistic experiments to do with the intelligibility of words spliced out of context seem to cast doubt on the usefulness of the categorization, however, Lieberman (1963) found that, the word borrower was recognised by 80% of subjects when isolated from the context, *The borrowers were all imprisoned,* but was only 45% intelligible in the context *Neither a borrower nor a lender be.* That is to say, the acoustic evidence for the word seemed to be much weaker when the semantic evidence for it was stronger. The discrimination of a word is inextricably linked to its integration with other types of knowledge.

4.4. Integrating Lexical Processing with Other Levels.

One way of coping with the dilemma of too much or too little bottom-up acoustic information is to use broad, and hopefully robust representation primitives initially to access a number of word hypotheses bottom-up and subsequently use a word verifier for more accurate matching and rating of the hypotheses against the input. Zue (1986) proposes the following:

> "...acoustic parameters are extracted and used to classify the utterance into broad phonetic categories. The coarse classification also includes prosodic analysis that identifies regions where the speech signal is likely to be more robust. The outcomes of these analyses are used for lexical access. The constraints imposed by the language on possible sound patterns should significantly reduce the number of word candidates. Once the phonetic context has been established, detailed acoustic cues can then be used to select the correct answer from the small set of candidate words."

A similar method was used in Hearsay-II but was costly and required careful control. Moreover it may sometimes be the case that the acoustic input is simply insufficient to discriminate between hypotheses. Lieberman's experiment cited above suggests that acoustic clarity decreases with predictability from context. In such cases, lexical access would be unable to distinguish the correct word without considerable input from the syntactic and semantic levels. Yet lexical access stands in the same relation to these levels as the acoustic front end stands to lexical access. Its function is to filter the words the higher levels need consider, yet its favourite words may be completely inappropriate from a syntactic and semantic point of view. It may need more general contextual information to discriminate between words, yet it has to provide the words that determine that context.

This view is supported by the experiments on late recognition of Pollack and Pickett (1963), Grosjean (1980, 1985), Shillcock, Altmann & Bard (1987). The latter report, for

example that, in their experiments with spontaneous conversational speech, 21% of successful word identifications were not based solely on left context together with the word's acoustic description. Subsequent context was required for recognition of the word. Yet this is not to say that lexical decisions should always be left open until the end of the utterance, as in HARPY. Indeed, in the later chapters we will show that this is impractical for any relatively unconstrained speech recognition system. It is also at odds with psycholinguistic evidence on speech processing. People seem to be able to use prior syntactic and semantic context to select a word even before the discrimination point specified by the phonological description (Marslen-Wilson & Tyler 1980, Cole & Jakimik 1980). It would appear to be the case that, for this kind of effect to take place, the word's function must be made available to the syntactic and semantic processes very early on in the access process.

The syntactic and semantic components must select from word hypotheses that are constantly changing their status or score as more acoustic information becomes available. Some of these early possibilities will drop out of sight completely as processing continues. Others may become more plausible as the over-all interpretation of the utterance proceeds. If this is the kind of filtering we require then we must develop an architecture which allows such changes of activity to be constantly monitored and responded to by the higher levels.

4.5 The Lexical Access Component

This section describes the chart-based lexical access component and the theoretical and practical considerations which motivated certain choices in its design.

4.5.1. The Lexicon

4.5.1.1 The Content of the Lexicon

The lexicon contains 4000 words which have the highest frequency of occurrence according to the American Heritage Word Frequency Book (Carroll, Davies & Richman, 1971). A set of phonological reduction rules was applied to this lexicon to derive fast speech forms, which were stored together with the citation form under the corresponding orthographic entry. The application of these rules resulted in the generation of 5300 reduced forms from the 4000 word citation form lexicon. The reduction rules are described in detail in Harrington et al (1986). Since there is no morphological decomposition in this lexicon, there are separate entries for words that are morphologically related. Each lexical item was also tagged with a syntactic category.

4.5.1.2. The Structure of the lexicon

We decided upon a tree-structured lexicon for both computational and linguistic reasons. It has been used in previous systems such as HWIM and Klatt's LAFS system, and so the ways in which it can be modified to deal with certain requirements of the speech processing problem have already been explored. For example, as in HWIM, both phonological variations within words and across words can be represented. Expected phonological variations within words can be captured through the pre-compilation of reduction rules mentioned above. And at least some of the (unpredictable) variations caused by error and ambiguity in the input can be dealt with during the matching process. This is described in the next section. Phonological variations *across* word boundaries can

be captured by creating a "wrap-around" tree. In such a representation each terminal branch in the tree is linked back to the branches for word beginnings. This creates a network of all possible word sequences rather than just a tree of isolated words. The transitions between word ending states and word beginning states are then modified according to word boundary phonological rules. (See Klovstad 1976 for a full description.)

This extension of the lexical representation from single words to pathways opens up a number of possibilities. Firstly, a pathway through the tree need not be a single word; it could be a phrase such as *Dear Sir, Yours faithfully*, or *over and out.* That is, rather than automatically re-entering the tree after a word ending, the path could continue through the tree establishing a close correspondence between certain words, independently of syntactic and semantic processing.

Secondly, pathways do not have to weave their way in and out of a single tree. A number of trees could be linked together, each tree representing (e.g.) a morphological class such as prefixes, stems and so on, thus allowing a word to be decomposed into morphological units. Thus two paths could be followed for the string /y uu n ii @ n ai z/, one for the stem *union* plus the suffix *-ise* and the other for the prefix un-, the stem *ion* and the suffix *-ise*. (See Thompson 1984 for a discussion.)

The primitives in the lexicon are fine-class phonemes, but of course other descriptions can be used on the branches of the tree. The LAFS system, for example, transforms a tree of phonemic descriptions into one based on spectral representations of diphones.

It is clear that the lexical representation is flexible enough to meet at least some of the demands of real speech, and certainly adequate for the experiments that will be described in the following chapters.

The main objections to the tree-structure are as follows. Firstly, alternative pronunciations of the same word are not distinguished in the lexicon from pronunciations of different words. However, word hypotheses compete on the Chart, not in the lexicon, so this similarity of status should not matter from a processing point of view. Hypotheses are checked for redundancy before they are added to the Chart. Different pronunciations of the same word are only added if they cover different portions of the input and therefore provide different word boundary segmentations.

Secondly, if a straightforward match, phoneme for phoneme, is made between the input and the branches of the tree, a failed match would prevent further recognition. To avoid this problem the match can be done on the basis of a segment's features.

Finally, a word's syntactic and semantic properties cannot be accessed until all (or most) of its acoustic material has been processed. Early access appears to be a property of *human* speech processing, but it seems unlikely that it is a necessary characteristic. No current automatic speech recognition system uses early access. Only TRACE has the capacity to do so, but a connectionist lexicon of 4,000 words is not feasible given current technology.

4.5.2.3. Creating the lexicon

The input to the lexicon building process is a list of items containing, for each item, a written form, one or more transcriptions in MRPA units (see Appendix 1), and a key for accessing syntactic information. This list was compiled into a discrimination tree in which, working from left to right, phonemic entries with identical phoneme sequences share the same branch. The first MRPA symbol in the transcription of an item is matched against the first set of branches in the tree. If a match is found, the process matches the

next MRPA symbol in the transcription against the set of branches following the initial branch. This process is continued until either the end of the input item or a terminal node in the tree is reached. If the end of the input has been reached, then one of two conditions might hold.

1) The pronunciation may have been stored before, but as part of a longer word (e.g. *tea* as part of *teacher*). In this case a terminal branch containing appropriate written and syntactic information about *tea* is attached to the node containing the pronunciation of *teacher*, at the point where the input word ended.

2) The pronunciation may have been stored before but the written and/or syntactic information may differ (e.g *tea* and *tee*), in which case the new information is appended to the existing written and syntactic forms.

If a terminal node has been reached then a new non-terminal branch is created, labelled with the first unmatched symbol. Other terminal word branches off this node are checked to see if they have remaining MRPA symbols which have to be distinguished from the current input by creating new non-terminals. Once this has been done, any remaining input MRPA symbols and the graphic and syntactic information associated with the new item are stored on a terminal branch. For example, the last syllable of *teacher* may not need to be distinguished in the tree until the addition of another word of two or more syllables such as *tedious*.

The result of this process is a discrimination tree, any of whose sub-trees contain a set of words beginning with the same phonemic description. Fig. 4.1 represents a fragment of the structure generated, containing the examples mentioned above.

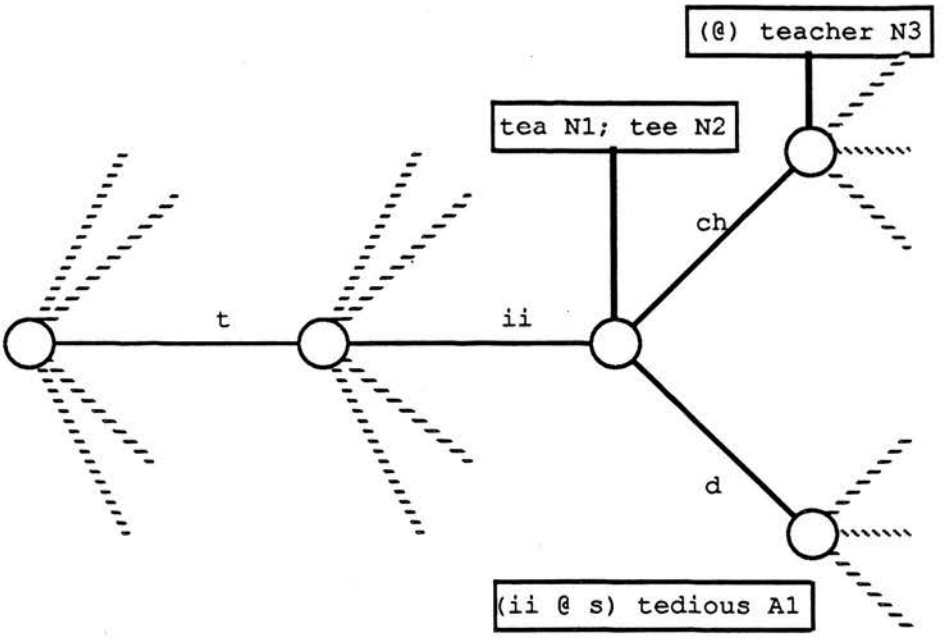

Figure 4.1

A fragment of the tree-structured lexicon

4.5.2. Lexical Processing

4.5.2.1. The Access Function

The ways in which the phonological information in the lexicon constrains interpretations of the phonemic input is as follows.

Input to the process is a graph of inactive edges marked with phoneme labels and possibly other information such as acoustic probability score. If the graph is output from the acoustic front-end it will contain alternative segmentations and labellings.

Lexical access begins as soon as inactive edges are posted onto the Chart by the acoustic front end. This might be integrated with lower level processing, or after all lower level processing has been performed on the entire utterance. Each symbol posted onto the chart is matched against the set of initial branches in the tree. For each match, a new active word edge is created carrying the phonemic symbol just matched, and a pointer to the node in the tree which follows that phoneme branch, indicating the set of phonemes which are expected to follow such a beginning. At each extension a new active word edge is created. The fundamental rule described in Chapter 2 ensures that each possible path in the tree which matches the input string will be followed up. If a match is not found then that path is abandoned.

When a node is reached which has a terminal branch, an inactive word edge is posted for the attention of higher levels of processing, and an active word edge is created with a pointer into the beginning of the tree. i.e. a word boundary is hypothesized. If the node has continuation branches then that path will continue to be followed.

Possible extensions to active word edges are not added to the Chart as soon as they are found but are placed on the Agenda. This means that the order in which hypotheses are followed up can be manipulated. Processing can continue until all possible word matches have been removed from the Agenda and placed on the Chart.

Fig. 4.2 shows one stage of the lexical processing of the fine class input /t ii ch @/ recorded on the Chart.

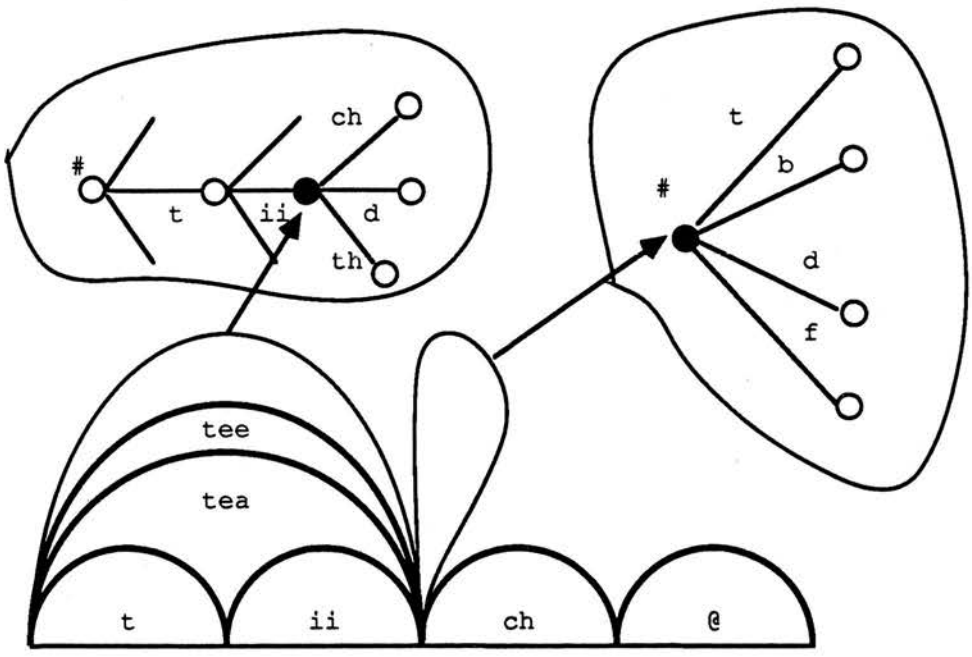

Figure 4.2

The state of the lexicon and the chart at a particular point during processing of the input *teaching*.

In the earlier section on primitives, we discussed the use of (possibly) more robust units such as broad-class phonemes to access words, this approach being used by both Hearsay-II and NOAH. The lexical component described here falls somewhere between this approach and the one used in HWIM. Each word and its anticipated variations in pronunciation are given explicitly in the lexicon, as in the HWIM system, while the acoustic input to the lexical access process may be described in terms of broad-, mid-, or fine-class descriptions. Before lexical access can take place, any broad- and mid-class symbols posted onto the Chart must be superseded by the addition of a fine class label for each fine class description in that set. If the acoustic front end provides a fine class

description, this is used directly, otherwise the required set members are accessed from a look-up table and added to the Chart. The system thus has the ability to access words from variably fine-grained descriptions of their phonological characteristics. If only manner of articulation and voicing are known, but not place of articulation, then all those words or parts of words corresponding to this mid-class description can be accessed.

This relieves earlier levels of the obligation to fully specify the phonetic characteristics of the input, and allows words to be considered as hypotheses through gradual accumulation of evidence, rather than through the binary decisions inherent in the tree structure. However, the less able the front end is to specify the full characteristics of the phonetic input, the more difficult it becomes to distinguish lexical hypotheses. Experiments with mid-class descriptions of isolated words have shown that the resulting equivalence classes are fairly small (Shipman & Zue 1982, Huttenlocher & Zue 1984). As we shall demonstrate in later chapters the word boundary ambiguities present in continuous speech, together with the increased number of homophones produced by the application of phonological rules for variations in pronunciation, relax the lexical constraints to the point where a very large number of paths through the lattice are equally plausible. The extent of such effects is explored in Chapters 6 and 7, where a mixture of broad- mid- and fine-class primitives are used during access.

4.5.2.2. The Discrimination Function

Given the ambiguity of word parses discussed in the previous section, we must keep a dynamic record of hypothesized word tokens that is distinct from the representation of word types stored in the lexicon. This record is the Chart. The edges at one level of description can be seen as mutually exclusive interpretations of the data. They may give

different interpretations of the *same* piece of lower-level information. For example, the phonetic string /p l ii z l e t @ s n ou/ can be parsed into word strings such as *please let us know, please lettuce know, pleas lettuce know, please letter snow, pleas let us no,* and so on. These strings are (acoustically) equally plausible; syntactic and semantic information would be required to decide between them.

In addition, the edges at one level can bring both left and right contextual information to bear on *different* interpretations of lower-level information. For example if the first two phonemes in the string are /r ii / the set of possible interpretations will be reduced to all words beginning *re-* such as *repudiate*, *rebuke*, etc. If the next segment is ambiguous between /p/ and /b/ then the possibilities of *repudiate* and *rebuke* will be maintained as separate competing active edges and will continue in competition until the incoming acoustic-phonetic information serves to discriminate between them.

This right context information can change the interpretation of a particular segment. For example, suppose the acoustic front-end provided confidence scores. The third segment might look very like a /b/ and be assigned a high score on the basis of its acoustic-phonetic features. Say, /b/ is given .8 and /p/ .2. If the process was stopped at that point and the Chart was examined, /b/ would be best interpretation of that segment. If, however, the following segments were those of *repudiate* the score of the word as a whole would serve to lift the interpretation of /p/ over /b/. Thus, the final decision on segmentation and labelling is made as a result of recognition at a higher level.

4.5.2.3. The Integration Function

As can be seen from the above two examples, the process of discriminating between words cannot easily be separated from the process of integrating lexical access with other

levels. Just as the identity of the /p/ or /b/ segment was affected by its integration into a lexical hypothesis, so the identity of a word will depend on its integration into a sentence context. Even a perfect phonetic transcription such as the /p l ii z l e t @ s n ou/ example generates many ambiguous words and word parses, and requires integration with a higher level to discriminate between the possible interpretations. If we decrease the specificity of the acoustic-phonetic information, we are likely to find an increase in word boundary ambiguity, as well as an increase in homophonic strings over the same stretch of sound. In the next chapter, we describe a series of experiments designed to explore the extent of this problem. This is followed by a discussion of how integration with other levels may be used to control the search space.

4.6. Conclusions

A review of previous lexical access components together with an examination of relevant psycholinguistic and linguistic data influenced our design of the lexical access model. The lexicon contained 4,000 citation forms and approximately, 5,300 reduced forms. Inter-word boundary effects were not implemented for the first prototype. The model of lexical access assumes a left-to-right, bottom-up strategy, with word boundaries being hypothesised through the accessing of the previous word. A final decision about segmentation would not be made until at least some words had been integrated into a higher level. The bottom-up status of a unit's identity (phonemes or words) could be changed through its incorporation into a higher-level, spanning hypothesis.

Chapter 5. Evaluating Lexical Access

5.1. Introduction

The evaluation reported in this chapter attempted to quantify three aspects of Lexical Access's performance: (i) success in identifying words, (ii) reasons for failure and (iii) contribution to the overall recognition process. However it must be emphasized that the primary aims of the evaluation were to review the architectural assumptions outlined in Chapters 2 and 3 and to guide further research.

In Chapter 2, I argued that it is very important to test the system as a whole as early as possible rather than to develop components in isolation. This means that in the early stages of a project the components tested will be very rudimentary. The evaluation reported here is of the first versions of the lexical access component (LA) and of the project's acoustic-phonetic front-end (SEGLAB).

The evaluation exercise was mainly concerned with the following:

(i) identifying and implementing useful measures of performance.

The performance of lexical access is often measured by number of words correctly identified. We were also interested in the number of *false positives* (i.e. incorrect words identified as correct). We grouped false positives into two classes: words identified because of errors in scoring and labelling acoustic-phonetic segments (e.g. *girl, curl),* and homophonic phrases caused by word boundary ambiguity (e.g. *party, par tea).*

(ii) identifying problems, and determining what was causing those problems.

Was it, for example, the front-end component, the lexical access component, the model of interaction between the two, or some combination of these? We found indications that homophonic phrases were more of a problem than had previously been thought. Firstly, it was not simply a question of poor front-end processing because even a substantial improvement in the front-end performance would have had little effect on reducing the problem. Secondly, the phrases extended over the entire utterance thus causing a potential combinatorial explosion of partial interpretations at higher levels of processing.

5.2. Success in Identifying Words

5.2.1 Methods and Materials

The first question we asked was, how many of the words spoken by the subject were posted at the correct place in the word lattice? This needed some clarification before we began to look for an answer. 'Words spoken' can mean two things: (i) the words intended by the speaker, the ones a human listener would recognize, but also (ii) the many other perfectly good word hypotheses generated by the same acoustic input. For example, there are straightforward homophones such as *meet/mete/meat*. The sizes of such equivalence classes of words under different phonemic representations can be determined off-line, but typically these are fairly small. There are also equivalence classes of word strings created by lack of word boundary information, e.g.

Patty cut a /pat eke utter .

It is important to realize that since these words use exactly the same phonemes as the intended words they are as valid, acoustically, as the intended words, and have the same acoustic score. We cannot expect to detect boundaries bottom-up, since there seem to be few reliable cues to word boundaries. It will be the task of higher level components to bring the intended words to the surface, according to their syntactic and semantic plausibility. Just as syntax or semantics must determine whether the correct word is *meat* or *meet* , such considerations must ultimately decide whether the correct response is *recognise* or *wreck a nice* .

There is no way of telling in advance how many of these partial matches will occur in any particular utterance. The hope is that they occur infrequently and that most would be eliminated after a word or two by mismatches to the acoustic input.

This evaluation examined LA's performance with respect both to intended words and to the equally valid words. In order to do this we first ran a fine-class transcription through LA. This gave us two kinds of 'correct' lattice: (i) a single path lattice containing only the words intended by the speaker and (ii) a lattice containing all the homophones and homophonic phrases that were valid under the fine-class description.

The phrase 'correct place in the lattice', also requires some more definition. In isolated word recognition only substitution errors of the *meet/meat* kind are possible since the beginning, ending, and duration of the word are known. In connected speech, however, in addition to the two kinds of substitution error mentioned above a correct word may be omitted or an incorrect word may be inserted into the string. How do we determine exactly where a word should begin and end if hypotheses on either side are incorrect? Does it have

to form a continuous path with other correct words? We need to have some idea of the word's relation to surrounding hypotheses.

A typical measure based on dynamic programming techniques aligns the recognised word string with the correct word string and reports correct, missing, substituted and inserted words. As we mentioned above, it only makes sense to count substitutions as errors when a grammar is used. The results reported in this section look at the percentage of correct words with and without substitutions. Section 5.3 looks at reasons why words were missed or inserted, and section 5.4 looks at the effect of substitutions.

When examining the lattice for correct word hypotheses, we simply checked to see if there was a hypothesis matching a word in one of the 'correct' lattices and within its time-frame. "Correct place" was measured according to Least Linear Distance. This is simply the sum of the distances between the corresponding end-points of the actual and hypothesized word durations.

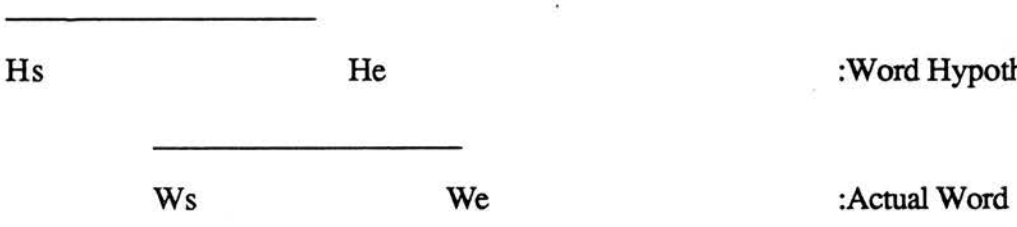

Figure 5.1.

Matching the Word Lattice against the Required Words.

The hypothesis having the minimum value of linear distance is chosen as the closest.

The following description of the materials is excerpted from Bard et al (1987 p. 2). A list of the utterances is given in Appendix 2. The fine-, mid- and broad-class symbol sets used are given in Appendix 1.

> "The materials consisted of five sets of grammatical sentences read by a single RP speaker at a normal, but careful rate, and recorded under laboratory conditions. Each set serves a slightly different purpose:
>
> **Set A: 16 phonemically dense sentences**, each containing numerous examples of a single MidClass. Another reading of these sentences by the same speaker was used in the development of RM1 and is a concise test of acoustic-phonetic rules. It is unlikely that most of these sentences would be uttered in a conversation or used in a meaningful text.
>
> **Set B: 16 MidClass-unique sentences**, containing only words whose MidClass transcription corresponds to a unique entry in the RM1 lexicon. Thus if the acoustic-phonetic rules and the accompanying normalization procedures work perfectly, lexical access should work at its maximal efficiency. These sentences seem unnatural perhaps because they lack short words like *in* and *the* and because consequently their rhythmic foot structure is irregular.
>
> **Set C: 16 sentences from the Golden Passage**. This passage was adapted from an existing text (a horticultural manual) so as to provide sentences in a natural and communicative style which contained among them an unusually broad distribution of MidClasses across environments. Only strings consisting entirely of items in the RM1 lexicon were selected.
>
> **Set D: 16 sentences from Section H of the Lancaster-Oslo-Bergen corpus of written English**. The 'LOB-H' texts are business and government documents such as might be dictated to a speech input workstation. They were used in the development of the syntax and collocational components of RM1 and provide data on which these should do well, while offering a naturally occurring distribution of segments. All words in these sentences were in the RM1 lexicon and most sentences contained at least two words present in the RM1 list of collocations.
>
> **Set E: 16 sentences from the Basic Corpus of business dictations collected during this project**. As a sample of typical input, these provide the most 'realistic' test of a recognition system. Again all words in the sentences occurred in the RM1 lexicon and words in the collocations list were as heavily represented as possible."

5.2.2 Results

Figure 5.2. below gives words correctly accessed by RM1 against words in the utterance. The five groupings correspond to the five sets of evaluation utterances. Within each grouping, the two columns correspond to the two definitions of 'correct' given above. That is to say:

(i) The column to the left records information about the intended words in the utterance, the words the speaker thought he was saying.

(ii)The column to the right refers to words accessed by running a fine-class hand transcription through Lexical Access. That is to say, it includes homophones and homophonic phrases.

The taller, paler columns show, for each test set, the number of words in the utterance given these two definitions of 'correct lattice'. It can easily be seen by comparing the two sets of columns that fine-class homophones together with lack of word boundary information result in a large increase of valid words.

The darker overlays shows the number of these words contained in the lattices output by RM1. This information is also given in terms of percentage words correct in Figure 5.3. The B set did particularly badly on number of words correctly accessed according to definition (i) i.e. intended words (1%), but did better on definition (ii) i.e. valid words (20%). We conjecture that this happened because the B set contain only mid-class unique words (as isolated words) and these words tend to be quite long. Long words are less likely to be identified given current performance, but parts of these words will be, and they will correspond to homophones in the second set of valid words.

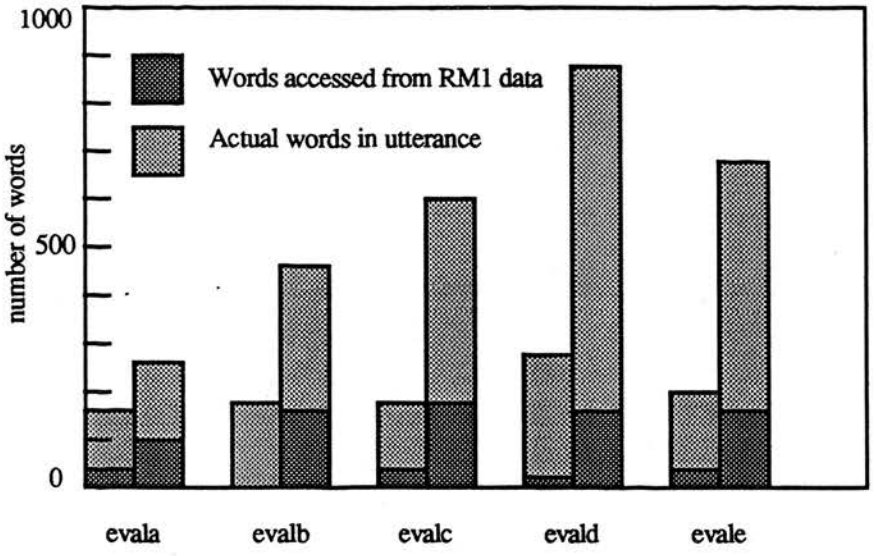

Figure 5.2

Words Accessed by RM1 against Required Words

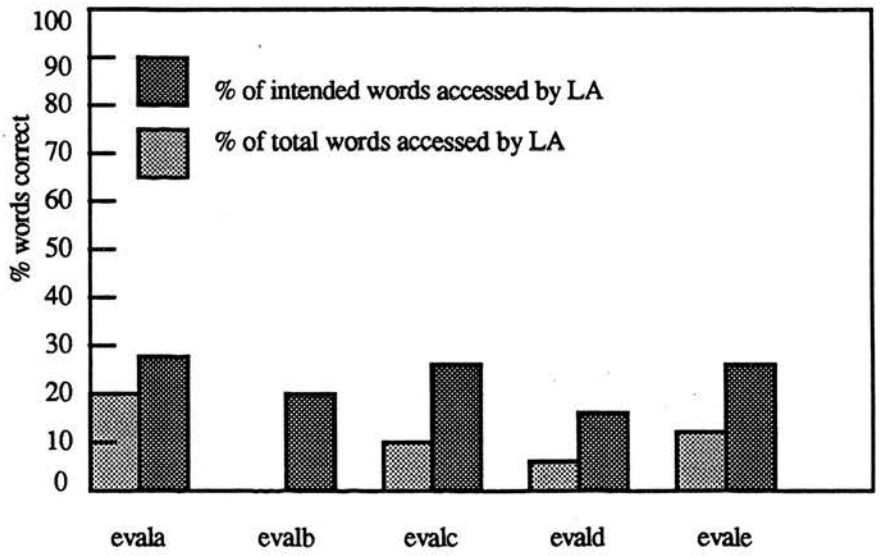

Figure 5.3

Percentages of Words Correctly Accessed

It is interesting to compare RM1's performance with the final performance of BBN's HWIM system (Woods et al. 1976). The number of phonemes correctly identified by RM1 cannot be directly compared with HWIM's figures since the phoneme lattice in HWIM essentially had a label for every phoneme in the system, given as a matrix of probability scores. In HWIM, the correct phoneme was one of the two highest scoring 52% of the time. RM1 correctly identified 48% of the phonemes in the test utterances, though of course, these labels stood for a set of phonemes (Bard et al 1987).

Table 5.1. below gives some performance figures for the HWIM and RM1 lexical access components. HWIM's evaluation was on the basis of the 15 highest scoring words found anywhere in the utterance. They do not include possibly correct but low scoring word matches. RM1, on the other hand, returns all possible word matches, but performs the search strictly left to right.

	HWIM1	RM1 'A' Set	HWIM2	RM1 All Sets
No. Test Sentences	124.00	16.00	99.00	80.00
Av. No. Words	6.20	6.375	9.21	9.60
Av. No. correct words per utt.	2.17	1.25	2.25	0.98
Av. No. incorrect words per utt.	10.13	216.00	*[1]	*
% words correctly identified	35.00	20.00	23.7	10.29

Table 5.1.

Performance Figures for HWIM and RM1

[1] Figures unavailable

5.3. Reasons for Failure

The aim of this part of the evaluation was to see what effects the performance of the front-end had on LA. It was intended to complement the evaluation of SEGLAB documented in (Bard et al 1987). We were not concerned with the front-end's overall ability to recognise phonemes but with the kinds of problems it presented to LA. Although we could expect considerable improvement in front-end processing, we were unlikely ever to have a phoneme lattice which reflected exactly both what the speaker said and the pronunciations in the lexicon. The system must be able to recover from input which deviates from the stored representations. This evaluation should help direct modifications to LA.

5.3.1 Evaluation Method

I examined 48 of the evaluation utterances by hand. The intention was to see what sorts of problems seemed to be occurring and then to create automatic procedures, where possible, for gathering further information. The output from the front-end was compared with the hand transcriptions and used to gather data about a number of classes of problem (Appendix 3). In addition, the reason why each word failed was documented (Appendix 4).

Each portion of the phoneme graph corresponding to a hand-transcribed phoneme was classified, with respect to the intended words, as one of the following:-

1) Correct	There is a hypothesized phoneme bearing the same mid-class label as the hand transcription and overlapping it by at least 50%.
2) Mislabelled	There are one or more hypotheses covering the region, none with the correct label.
3) Missing	The region is covered by hypotheses centered over adjacent phonemes.

The correct phonemes were further classified as:-

4) Merged	One hypothesis covered two or more phonemes.
5) Split	There were two or more correct hypotheses in sequence over the region of the hand-transcribed phoneme.
6) Path Error	The right hand end of one correct hypothesis did not correspond to the left hand end of the following one.

5.3.2. Results

The results of this analysis are shown in the Figs. 5.4. and 5.5. below. The number of times a region was labelled as one of the above classes is shown as a percentage of the actual phonemes in the hand transcriptions.

The analysis indicates the problems LA must cope with even when a large percentage of the individual phonemes have been correctly identified. For example, the utterance EvalA11, *Three chefs face a thief*, provided the most correct lattice with thirteen phonemes out of fourteen correctly identified. However, none of the words were accessed by LA (See Appendix 4). The missing mid-class liquid, L, in *three* caused an FF phoneme sequence (two voiceless non-sibilant fricatives), which blocked all word recognition. Even if this had been overcome, a path sequence error between *a* and *thief* would have caused problems further on.

Some errors could be fixed fairly easily. One example is the FF block mentioned above. In this case, it is easy to see that there is an errorful patch in the lattice, because all active word edges have been blocked before the end of the lattice has been reached. LA could try a number of strategies, such as assuming a missing or inserted phoneme, in order to get over the mislabelled portion. There are also a few cases where we can roughly predict the location of a particular error. For example, extraneous voiceless stops are often hypothesized at utterance onset. We could modify either LA or SEGLAB to reduce this problem.

We mentioned earlier that HWIM performed slightly better than RM1 in accessing the intended words correctly. This was partly because HWIM incorporated a mechanism for coping with split or merged phonemes. LA assumes that a word can be present only if all of its constituent phonemes are present for one of its pronunciations. This is an extremely unrealistic expectation.

It will undoubtedly be necessary to implement a partial matching access mechanism to account for these and missing segment errors. Such a strategy would have to be very carefully designed, however, if it were not to result in a huge explosion of word hypotheses. The HWIM system had only 1,362 entries whereas we had 4,000 potential candidates in the lexicon (excluding reduced forms), with plans to increase this to 20,000.

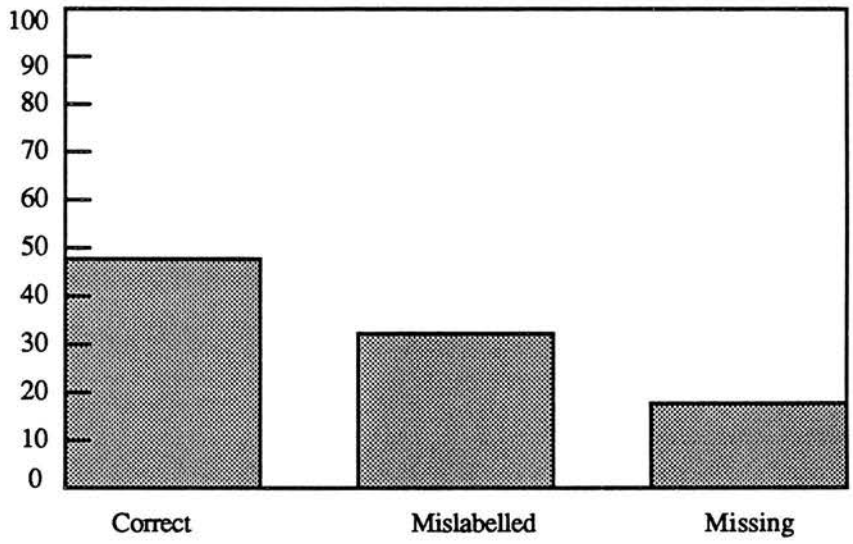

Figure 5.4

Percentages of hand-labelled segments corresponding to above classifications.

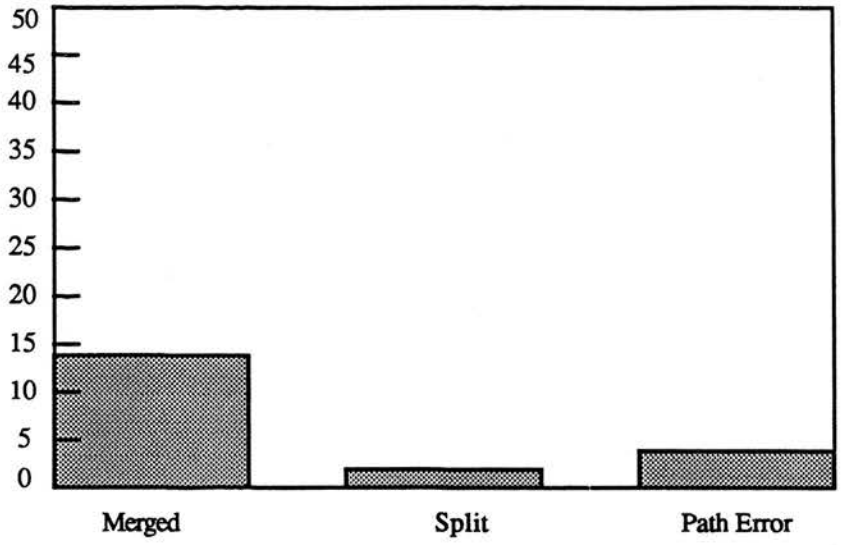

Figure 5.5

Percentages of hand labelled segments which were correctly labelled but which caused an error in LA.

5.4 Contribution to the Recognition Process

LA passes several hundred word hypotheses to the higher level components, most of which are false positives. Some are produced because of errorful labelling at the phonemic level, but others are simply homophones under some phonemic representation. In this part of the evaluation, we were interested in finding out what sorts of problems this generates for the levels above LA.

I mentioned above that, in addition to single word confusions such as *meat/meet*, there are also confusions generated by lack of word boundary information. So not only do we have to take account of the confusability of individual words in the lexicon, but also of word strings. This is particularly so in the case of a recogniser with a very general grammar. A very constrained grammar might not contain any template or rule allowing the

phrase *wreck a nice*, instead of the item *recognise*, at a particular point in processing; a more general grammar might have to consider both interpretations.

The inherent confusability of the lexicon is also affected by the specificity of the input representation. The broader phonetic representation used in the RM1 front-end will produce more homophones and homophonic phrases than a fine-class representation. We were interested in seeing if this increase was significant from the point of view of higher-level processes. Did such substitutions produce very many alternative word parses stretching from beginning to end of the utterance? Or did the partial substitutions usually fail causing dead-ends in the lattice? For example, the phoneme string /t ii ch i ng w i l/ initially produces the word hypotheses *tee*, *tea*, and *teach* but these quickly drop from consideration because there are no lexical items matching / i ng w/.

5.4.1. Evaluation Method

LA was run over three types of input. These were (i) the fine-class transcription of the utterance, (ii) this transcription translated into mid-class symbols, and (iii) the output of SEGLAB. A path-counting algorithm determined the number of words strings for each of the lattices.

5.4.2. Results

The results listed in Appendix 5 clearly show that lack of word boundary information together with acoustic-phonetic uncertainty result in a huge increase in the possible word parses of an utterance. Even the perfect fine-class transcriptions resulted in extremely large numbers of paths (an astonishing average of 862,300), making it very difficult for the syntactic component to distinguish the correct interpretation.

This will lead to even greater problems when there are errors in scoring or labelling some portion of the lattice. It seems impossible to decide when one string out of so many millions should be 'rescued' either by LA or by some higher-level component. A good example is Evala7 (*Patty cut up a potato cake*) in the development set. *Patty, cut, up, a,* and *cake* are all on the word lattice. *Potato* failed to be accessed only because the first vowel was missing. However, there were many other words and word strings covering the region of *potato*, so why should LA apply special procedures to rescue this particular word? At a higher level there was a string *Patty cut up a*, but the extension of this string over the errorful portion lowered the score of the string relative to other interpretations. Again, why should higher level components not abandon this string in favour of the many thousands of better scoring strings?

It seemed probable that the number of word paths would increase substantially with the length of the utterance, and some of the test utterances were very long. Evald08, for example, contained over 63 million paths, but the utterance --*You may wonder what happens to our boys and girls and the answer can best be found in the pages of the old boys and girls magazine* -- is 22 words long. Clearly, we could expect some sort of boundary indication, such as a pause, to occur long before the end of this utterance. Therefore, in the test runs described in the next chapter we cut down the utterance length to no more than 10 words.

5.4.3. Discussion

This experiment shows how different components can work together to compound a problem. The number of phonemically equivalent words returned by LA seems to be affected by the factors discussed below.

5.4.3.1 The content of the lexicon.

It would be possible, though extremely limiting, to exclude all homophones. The number of homophones is increased by allowing reduced pronunciations. The figure below shows a portion of the word lattice for a fine-class representation of EvalA2 (*I'm naming one man among many*). Many of the hypotheses are generated because of reduced pronunciations such as /n/ for *and*. However, such reduction pronunciations appear to be necessary in order to access and represent alternative alignments (e.g. *a name for.../and aim for).*

Although we could not remove all homophones, we could treat differently certain classes of words which are frequently accessed erroneously. One such class includes letter names. These are often homophonous with other words, e.g.

t	tea, tee
b	be, bee
c	sea, see

When a run was done using a lexicon without letter names, the number of paths produced was a few orders of magnitude less than when these words were included.[1] Similarly function words could be treated differently from content words. This was suggested by the NOAH team (Smith & Sambur 1980) who found that short function words, which made up only 1% of their lexicon accounted for 30% of errorful hypotheses and only 10% of correct ones.

[1]Maggie Cooper suggested this change and provided the updated lexicon.

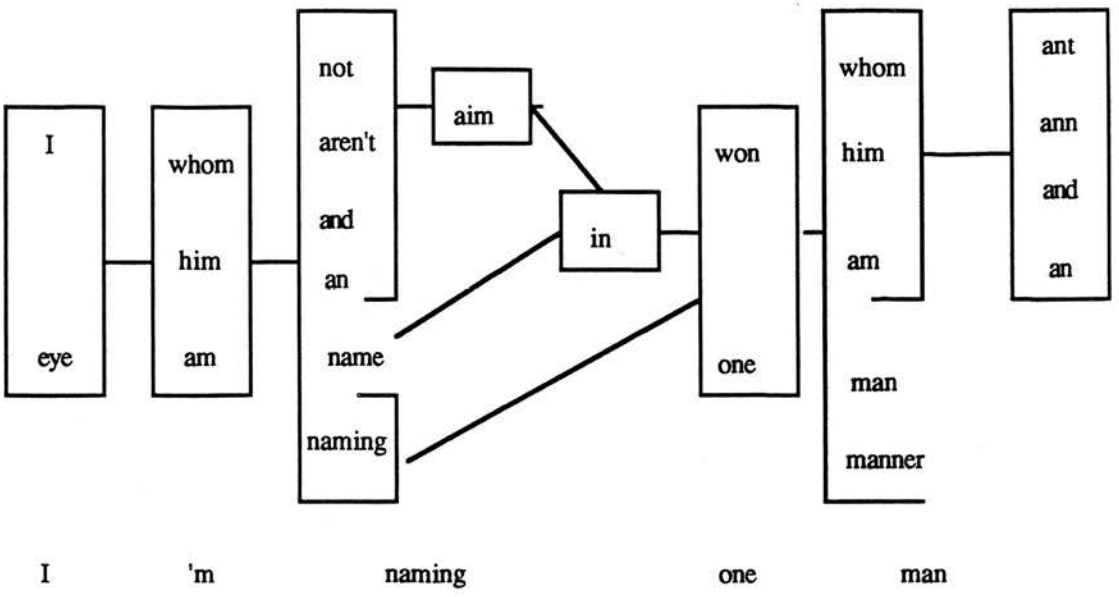

Figure 5.6

Word Lattice for EVALA02

5.4.3.2. Characteristics of the utterance.

The size of the word lattice is increased by homophones which fall within, or cut across, the boundaries of the input utterance, as in:

associate --> a sew see ate

recognize speech --> wreck a nice peach

This would seem to be more likely to occur in longer input utterances. However later tests (Harrington and Johnstone 1987) found that it was not necessarily the case that longer input utterances (where length is defined as number of phonemes or number of words intended by the speaker) necessarily gave the greatest number of parsings into word strings: there was no correlation between number of phonemes in the utterance and number of parses into words ($r = -0.07$, not significant); neither was there a significant correlation

between number of words produced by the speaker and number of possible parses of its phonemic representation into words strings (r = 0.11), although there is a trend to show that these two variables are positively correlated.

Some long utterances may contain phoneme sequences which only match the intended words. An example is utterance Evala16.

Does John believe you were measuring the gun.

This sentence only has one interpretation given a fine-class transcription. The constraint that word beginnings only occur at the point where a previous word ends, prevents *leave* being found in *believe* and *a ring* being found in *measuring*.

5.4.3.3. The phonemic representations.

There is always the possibility that the input pronunciation will differ from the pronunciation in the lexicon. If we impose too stringent constraints on the match, then we will fail to access the correct word. On the other hand, if we relax constraints, we are trading discriminability against robustness. In other words, we might not throw the baby out with the bath water, but we might keep so much water in the bath, the baby drowns.

There have been a number of studies on isolated words which explore the trade-off between discriminability of lexical items, and the amount of phonemic information used in their representation. The evaluation runs mentioned above indicate that the results with mid-class representations do not hold for connected speech. The next chapter reports a more systematic study of these effects.

5.4.3.4. The constraining power of the grammar

In the above runs no grammar was used. Any word could follow any other word, just so long as it matched the phonetic input description. A grammar could, of course, be used to reject certain word candidates or to assign higher probability to some over others, and thus reduce the number of paths. However, the grammar must be able to correctly distinguish word hypotheses or the number of paths will grow exponentially. By *correctly distinguish* I mean that the correct phrase must be the only phrase with a certain score, and it must have the highest score. If either of these conditions is not met locally, the search algorithm will have to pursue a number of hypotheses in parallel for some length of time. If there are just seven equally probable choices on average at any given point after 2 decision points we will have 7^2 paths, after 8, 7^8 = 5,764,801 paths. These issues are discussed in more detail in the following three chapters.

5.5. Conclusions

This evaluation highlighted one of the points made in Chapter 2: what goes on between components is as important as what goes on within any component. There are two reasons why it is important to look at the interactions.

The first reason is obviously that the performance of, say, the syntactic component depends on the performance of other, lower-level components. If we know that there are a few billion equally plausible lexical paths through a mid-class lattice, we may cease to wonder why syntax failed to produce the intended utterance in its top ten.

Secondly, the interaction of different components may also indicate that *less* work can be done at some other level. If higher level components can sort out the intended string from the many false positives, then LA can take fewer risks when matching word

hypotheses to the acoustic input. It can aim for a high probability of having the right word somewhere in the cohort without worrying too much about the size of the cohort.

This opens up for debate the question of what can be done, at each level of the system, to fix some problem. The first step has been to start classifying the problems. The kinds of problems faced by LA were discussed in section 5.3. A rough list is reproduced below.

1) Missing phonemes
2) Mislabelled phonemes
3) Merged phonemes
4) Split phonemes
5) Overlapping phonemes
6) Gaps between utterance onset and first phoneme
7) Gaps between phonemes
8) General lack of robustness at lexical level

In addition, lack of word boundary information and of acoustic cues to a word's identity created very large word lattices.

The question then arises, whose problem is all this? Some problems such as the (possibly labelled) gap between utterance onset and the first phoneme may have an obvious solution. SEGLAB could include a "lip-smackin" rule as in HWIM which detected the sort of voiceless burst which precedes many utterances. Other problems do not have such obvious solutions, nor is the locus of the problem as easy to identify. The *potato* error in EVALA07 will serve to illustrate this.

The acoustic front-end failed to detect the first vowel. Should we then insist that it is all SEGLAB's fault and sit back waiting for it to improve? Given the nature of fluent speech we might have to wait indefinitely.

Alternatively, should LA implement a matching strategy that would allow for the missing schwa? If so should LA instruct SEGLAB to look for evidence of a schwa or should it just make one up? Would the acoustic signal contain any evidence anyway or does the speaker just talk like that?

Why should LA relax constraints on that particular word when there are other hypotheses covering the same region? (i.e. */@ p t ei t @/ --> up to eight a)*. Should LA leave well alone, and let syntax or semantics choose from the competing hypotheses?

These questions concern the model of speech processing. In Chapter 2, I made the distinction between the architecture and the model implemented within that architecture and argued that the graphical nature of the Chart framework would facilitate the evaluation of the model's components both together and separately. The present chapter provides concrete examples of such an evaluation. Section 5.3 shows how the graphs can be studied by hand to find evidence of particular problems, problems that may not have been anticipated during the design stage. Once a problem has been identified, automatic procedures can be designed to provide information on the extent of the problem.

Sections 5.2 and 5.4 show how such statistical information can easily be gathered by running automatic graph-searching procedures over the Chart output. Section 5.2 measures the performance of LA in terms of the percentage of words correctly recognised. Section 5.4 is more interesting, however, in that it gives some indication of the problem that false positives could cause even if LA succeeded in accessing all the intended words. Section 5.4 is also interesting because the problem it raises cuts across the boundaries of linguistically defined levels of analysis. The next two chapters examine this problem in more detail.

Chapter 6. The Size of the Word Graph

6.1. Introduction

In earlier chapters we discussed the possibility that poor acoustic-phonetic processing was primarily to blame for delays or errors in automatic speech recognition. People could transcribe phonemes with a high degree of accuracy, even without syntactic or semantic context, whereas the ARPA systems only achieved bottom-up recognition rates of around 50%. Psycholinguistic models also concentrated on how good people were at recognising speech. Fast word-by-word recognition was often assumed as the basis of the model.

However, we also saw that there were good computational and psycholinguistic reasons for supposing that the phonemic interpretation of a stretch of sound would often be left ambiguous for some time. The evidence indicated that, given the nature of continuous speech, a robust system would not make too early a decision, but would allow higher levels to inform the process of identification. I argued that on some occasions there might simply not be enough acoustic information to make a decision, and cited psycholinguistic experiments on the intelligibility of words in certain contexts. Furthermore, on other occasions the acoustic-phonetic information might be misleading; we would want to be able to recover from /a g ch u @ l i/ or some other minor mispronunciation. Therefore, it becomes necessary to allow several lower level interpretations to remain active, at least for a while.

However, we found indications in the last chapter that this could lead to a very large number of possible word strings. This chapter explores in more detail the effect of such equivalence classes on parsing phonemes into words.

6.1.1. Graph Depth

In Chart-parsing terms, leaving the lower-level interpretation open effectively means increasing the depth of the phoneme graph. While this makes it easier for the acoustic-phonetic component to guarantee the inclusion of the target phoneme, the task of the syntax/semantic component is made considerably more difficult since it has to choose from many more word candidates. It is important to choose a phoneme graph depth which is optimum from both the acoustic-phonetic and syntax/semantics point of view.

The HWIM system did not limit phoneme identity at all in the early acoustic-phonetic stages; each of the phoneme labels used by the system was a possible candidate for each segmentation. In addition, alternative segmentations were permitted thus increasing the lexical access component's chances of finding a suitable path. Of course, it was necessary to come to a decision at some point, and to do this the system relied primarily on the acoustic probability scores attached to each labelling.

The Hearsay-II system used segments categorized by manner of articulation features only, avoiding the more difficult task of specifying place of articulation. These segments can be viewed as descriptions of sets of phonemes (i.e. those sharing the same manner of articulation).

One of the aims of this chapter is to look at the effect on word discriminability in continuous speech of increasing the depth of the phoneme graph. We shall compare graphs of depth one with graphs described in terms of sets of phonemes.

6.1.2. Graph accuracy

One of the major criticisms of the ARPA systems concerned the poor performance of the front-end processors. The systems failed to identify an acceptable percentage of the correct phonemic labels, and also hypothesized too many incorrect labels. Klatt writes:

> "Recent experiments by Mark Liberman and Lloyd Nakatani suggest that listeners can transcribe English nonsense names embedded in sentences (and obeying the phonological constraints of English) with better than 90% phonemic accuracy. It is likely that machine performance must approach this figure before very powerful speech understanding systems are realized." (1977 p.1356).

In this chapter, I look at what would happen if the acoustic-phonetic front-end *did* achieve near perfect performance. We will assume that the correct phoneme label is in there somewhere; 100% recognition is achieved in that sense. The system will not have to contend with errors in either labelling or scoring. We will also assume that perfect segmentation has taken place; there are no competing alignments of overlapping phonemes in the graph. Using this input we will examine the effects of word boundary ambiguity and homophonic word ambiguity on the number of possible word strings.[1]

6.2. Input to the Lexical Access Process

Zue (1985) defines a collection of words having the same representation as an equivalence class. A number of words are homophonous in this way (e.g. *sew/so*, *meat/mete/meet*), but the size of such equivalence classes is very small. When we start to put these words together things start to get a little trickier. A string such as /s ou # m ii t/

[1] This chapter summarises work reported in Harrington & Johnstone 1987.

can be parsed into six different word strings even when the word boundary is known. When the boundary is not known, we begin to get equivalence classes of phrases. As I pointed out earlier, the phonemic description of *This nudist* is ambiguous (at least temporarily) between *This nudist...*, *This new dist...* and *This nude is t....* Part of this study looks at the effect, using such fine-grained phonetic descriptions, of this kind of ambiguity on the whole utterance.

Since we do not expect the front-end to recognize only the correct phonemes we also look at less informative representations. The expansion of phonemes into mid-classes (Table 6.1. See also Appendix 1) might provide a suitable balance between acoustic-phonetic processing and syntactic/semantic filtering of the competing word-strings for two reasons.

Firstly, mid-classes are groups of phonemes which are easily confused at an acoustic-phonetic stage of processing: thus /m/ and /n/ are grouped into one mid-class, since [m] and [n] are are often nearly identical from an acoustic point of view. The substitution of a phoneme /m/ by its mid-class N (i.e. the inclusion of /n/ and /ng/ as alternatives to /m/ in the phoneme graph) will therefore increase the probability of a correct correspondence between the phoneme graph and the acoustic signal.

Secondly, from the point of view of lexical discrimination, Zue in the above paper summarizes studies which looked at equivalence class sizes for a 20,000 word lexicon represented using six broad phonetic categories. Since the phonemes are distributed among only six categories (approximately half the number of mid-class categories), there is a considerable loss of information, and one would expect an increase in the size of the equivalence sets. Table 6.2 is taken from Zue's summary.

Mid-class		Phoneme members
P	voiceless stop	/p, /t/, /k/
B	voiced stop	/b/, /d/, /g/
S	voiceless sibilant fricative	/s/, /sh/, /ch/
Z	voiced sibilant fricative	/z/, /zh/, /jh/
F	voiceless non-sibilant fricative	/f, /th/, /h/
V	voiced non-sibilant fricative	/v/, /dh/
N	nasal	/m/, /n/, /ng/
L	liquid	/l/, /r/
G	glide	/y/, /w/
D	dipthong	/ai/, /ei/, /oi/, /au/, /ou/, /i@/, /e@/, /u@/
FV	front vowel	/ii/, /e/, /a/
BV	back vowel	/aa/, /o/, /oo/, /u/, /uu/
CV	central vowel	/i/, /@@/, /@/, /uh/

Table 6.1.
The relationship between mid-classes and phonemes as described in Dalby et al (1986).

	Equally Weighted	Frequency Weighted
Expected class size	22	34
Median class size	4	25
Maximum class size	223	223
% unique (single word) classes	32	6

Table 6.2.
Representing words by broad phonetic classes (from Zue 1985).

According to table 6.2, nearly a third of the words are still uniquely identifiable, and one would expect the number to be considerably greater under a mid-class representation. Furthermore, the mean size of the equivalence classes is only 0.15% of the size of the lexicon, and the largest class is only equivalent to about 1% of the size.

On the face of it, these results look very promising for word recognition but what are the implications of these results for a syntactic/semantic component? If we strung together a

sequence of these sets the sequence would be, in the best case, completely unambiguous: a sequence of uniquely identifiable words. In the worst case, if we had two adjacent maximum sets in the sequence then the parser would have to consider 223 * 223 = 49,729 possible word-pairs. And this would increase exponentially if a further maximum set followed. The average set size is quite small, however, even in the frequency weighted case.

But these statistics do not take account of the fact that, in continuous speech, word boundaries are more difficult to identify from a given mid-class string compared with a phonemic string. Thus, while at a phonemic level the sequence /m g l/ (e.g. *same glass*) can only be parsed into /m # g l/ (Lamel & Zue, 1984) at a mid-class level (i.e. /N B L/), the unambiguous identification of the word boundary is no longer possible. Since the mid-class category N include /n/ and B includes /d/, N B L could also be parsed as N B # L (e.g. *sand layer*), or indeed N B L (e.g. *sandal*). And since phonemic constraints across word boundaries often no longer successfully apply at the mid-class level, the total number of ways of parsing a given mid-class string into words is likely to increase considerably, despite the fact that the lexicon remains highly discriminable when represented in mid-classes. The experiments in this chapter are designed to determine the magnitude of this increase and to assess whether this would place an unmanageable burden on syntactic and semantic filtering.

All of the analyses reported here were based on hypothetically perfect transcriptions of the target utterances into whatever recognition units were implemented. So none of the experiments take account of the possible errors and ambiguities that can arise as a result of processing the acoustic waveform by the acoustic-phonetic front-end. Furthermore, while it was possible to take into account many of the phonological variations attributable to fast speech production, the analysis excluded a consideration of phonological assimilation across word boundaries. If such effects were included, the number of homophones would

probably be still greater. Even using fine-class representations, a word such as *hand*, for example, becomes homophonous with other lexical items in certain contexts:

/h a m/ -- as in /h a m # m i # dh a p # b u k/ (*hand me that book*)

/h a ng/ -- as in /h a ng # k r i s p s # a r au n d/ (*hand crisps around*)

This effect is likely to be greater with mid- and broad-class representations.

To summarize, we expect there will be homophones and over-lapping, homophonic phrases at various places in the input and that the size of such equivalence classes will increase with a decrease in the specificity of the acoustic-phonetic information. If such phrases always fail to extend more than a few words, through failing to match the input, then we can afford to delay the interpretation without placing too much of a burden on higher level components. However, if this is not the case we may find a combinatorial explosion of word strings. Our objective is to see how many more word strings there are, other than the correct interpretation, spanning a fair portion of the utterance. We have chosen to limit the input utterances to an average of seven words, though a speaker could easily say more words without pausing.

6.3 Method

Phonemic transcriptions were made by a trained phonetician of the 80 sentences listed in Appendix 1. All utterances that were more than 10 words in length were broken down into clauses of less than 10 words thereby producing a total of 115 utterances with an average of 7.07 words and 26.56 phonemes per utterance.

These phonemic transcriptions were then converted automatically to their corresponding mid-class representations. In addition, since we suspected that the mid-class utterances would parse into an exceptionally large number of word-strings, the phonemic utterances

were also automatically converted to a mixed broad and phonemic representation. Only the voiceless stops, voiced stops, non-sibilant fricatives, and nasals were converted to the classes shown in Table 6.3, but all other segments were left in their phonemic form. As a result, 37.6% of the phonemes in the 115 utterances were converted to broad-classes in the "mixed" utterances. The choice of the broad-classes in the mixed case reflects the finding of a high degree of confusability in the separation of their respective members in the acoustic-phonetic analysis of the speech waveform (Dalby, Laver & Hiller, 1986).

	Broad-class	Phoneme members
P	voiceless stop	/p, /t/, /k/
B	voiced stop	/b/, /d/, /g/
NSF	non-sibilant fricative	/f/, /v/, /th/, /dh/
N	nasal	/m/, /n/, /ng/

Table 6.3.
The broad classes in the mixed sets

For the purpose of comparison the three different types of representation for the utterance *The order goes in by late November* are shown below:

[Henceforth the following abbreviations will be used: **Pu** (unstressed phonemic); **Xu** (unstressed mixed); **Mu** (unstressed mid).]

/dh i oo d @ g ou z i n b ai l ei t n @ v e m b @/ **(Pu)**

/NSF i oo B @ B ou z i N B ai l ei P N @ NSF e N B @/ **(Xu)**

/V CV BV B CV B D Z CV N B D L D P N CV V FV N B CV/ **(Mu)**

It is emphasized that all the input utterances were based on hand transcriptions, rather than the output of acoustic-phonetic processing. As such, the hand transcriptions can be considered to be a perfect analysis by the acoustic-phonetic component into a string of phonemic, mixed, or mid units, excluding any representation for word or syllable boundaries and excluding the effects of assimilation across word-boundaries. The parsing process takes place as described in the previous chapter. Only complete parses of the phonemic string from beginning to end of the utterance were counted.

6.3.1 Path counting[1]

We want complete paths because these are equal competitors, acoustically and lexically, with the final, correct interpretation. Partial paths compete temporarily but then drop out because of a mismatch with the incoming phonemic material. While a profusion of partial paths may strain the resources of the computing machinery, they need not affect the interpretation of the complete utterance.

The paths through the graph are calculated through a process of graph reduction. The following reductions are applied iteratively until there are only two vertices, one defined by the start of the first word in the utterance, the second by the end of the last word. These vertices are joined by a single edge labelled with the number of paths between the vertices.

ReduceEdge *(a, b)* Replace the edges between *a* and *b* with a single edge *a-b* whose label is the sum of the paths between *a* and *b*. (Fig.6.1)

ReduceVertex *(a)* If *a* has just one incoming edge *x-a* and one outgoing edge *a-y* replace *a* and all its edges with an edge *x-y* whose label is the product of the paths on *x-a* and *a-y* (Fig 6.2)

[1] This algorithm was written by Julian Kupiec

ReducePaths *(a b c)* If *b* has one outgoing edge, *b-c* and two or more incoming edges with different start points *a-b* and *x-b*, replace edge *a-b* with an edge *a-c* labelled with the product of the *a-b* edge and the *b-c* edge. (Fig. 6.3)

Sub-graphs that are disjoint to the main graph (i.e. do not have a continuation through to the utterance final vertex) are not included on the final edge count.

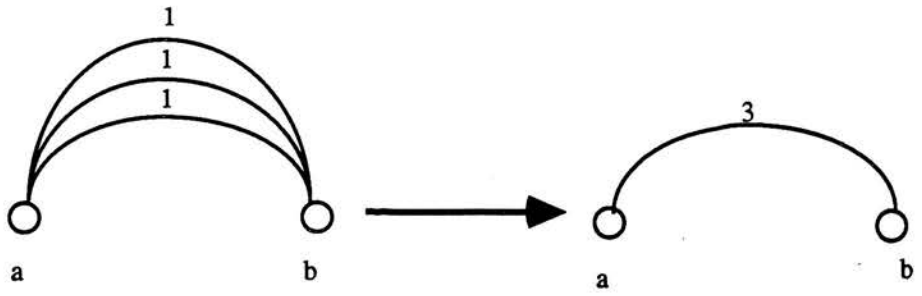

Figure. 6.1. ReduceEdge

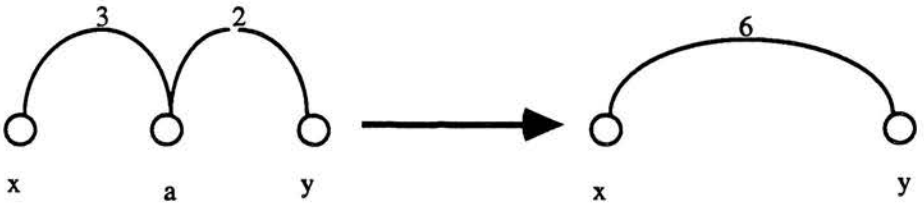

Figure. 6.2 ReduceVertex

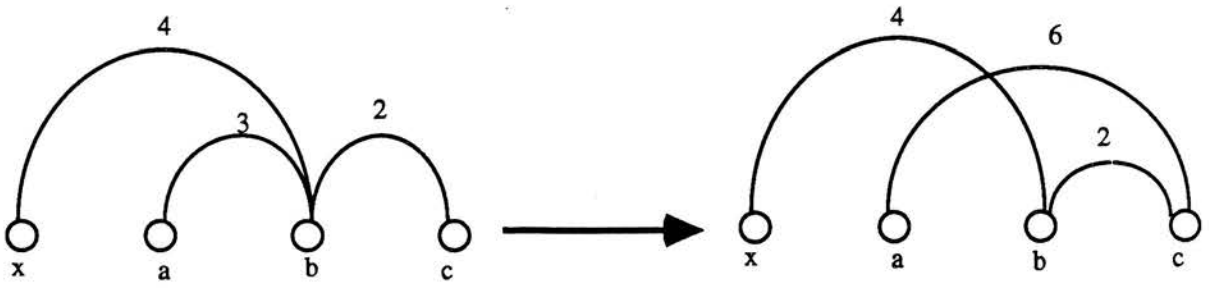

Figure. 6.3 ReducePaths

Graph reduction functions in the path counting algorithm

6.4. Results

Figures 6.4 -- 6.6 show histogram distributions for the total number of derived word-strings when the input utterances are matched against the lexicon.

The results show that in the mid and mixed sets at least, some utterances were parsed into an exceptionally large number of word-strings. For **Mu**, 71/115 (62%) of the utterances were parsed into 10 million or more word-strings (Fig. 6.4), but only 9/115(8%) utterances were parsed into 1000 or less word-strings. The average number of word-strings for **Mu** was 3.88×10^{16}. For **Xu**, 15/115 (13%) of the utterances were parsed into 100,000 or more word strings (Fig. 6.5), and 34/115 (29%) of the utterances were parsed into 100 or less word-strings. The average number of word-strings for **Xu** was 6,228,298.1. For **Pu**, 18/115 (16%) of the utterances were parsed into 1000, or more word-strings (Fig 6.6) and 30/115 (26%) of the utterances were parsed into 10 or less word-strings. The average number of word-strings for **Pu** was 1790.9. An example of one of the unstressed phonemic utterances which produced just under 16,000 alternative word-strings is given below.

/b r aa n sh @ z aa r @ m uu v d @ n t i l dh e@ i z jh uh s t w uh n l e f t/

(branches are removed until there is just one left)

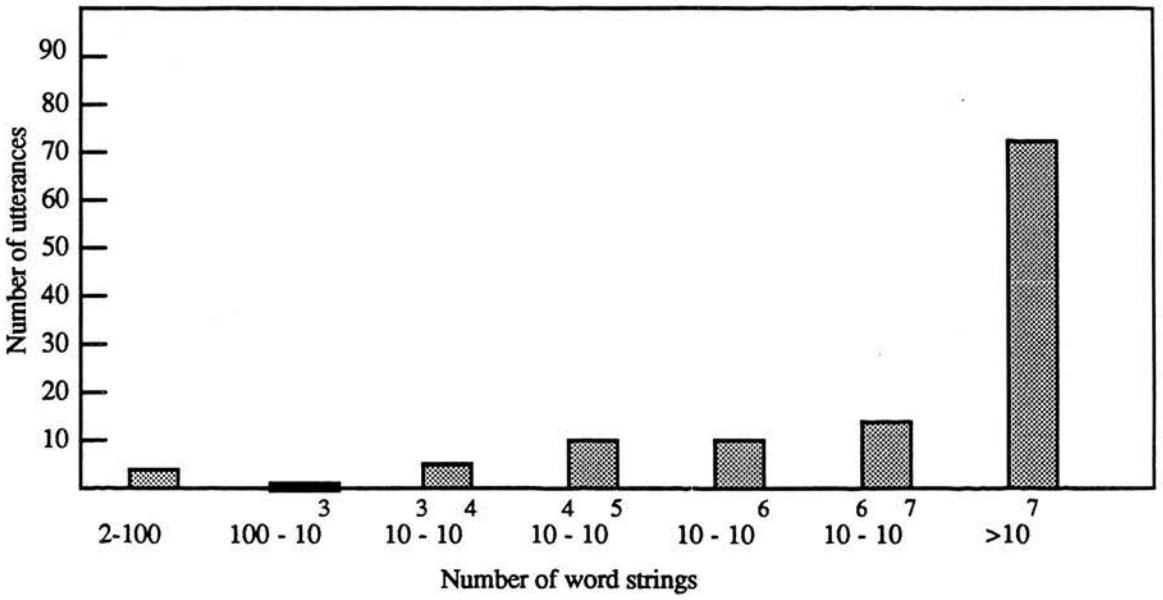

Figure 6.4.

Distribution of the numbers of word strings derived from the mid-class representations (Mu) of the 115 utterances.

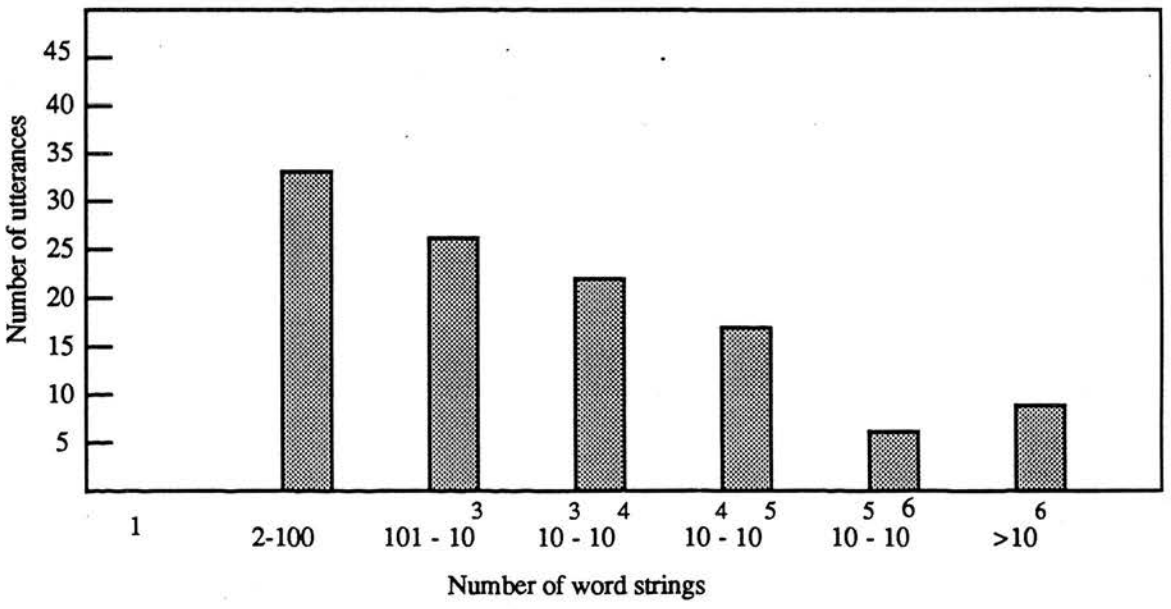

Figure 6.5.

Distribution of the numbers of word strings derived from the mixed representations (Xu)of the 115 utterances.

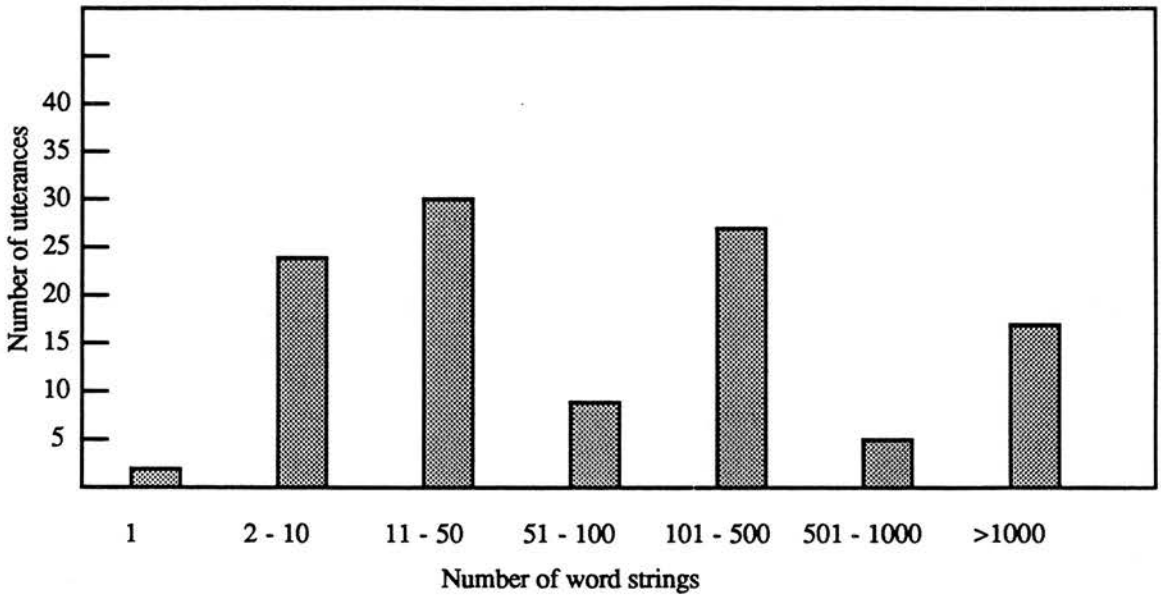

Figure 6.6.

Distribution of the numbers of word strings derived from the phonemic (Pu) representations of the 115 utterances.

6.5 Discussion

6.5.1. The problem of errors

This study has shown that when phonemes are selectively encoded as broad classes (the mixed sets) or entirely encoded as mid-classes, utterances are sometimes parsed into over a million word-strings. This places an unmanageable burden on syntactic and semantic filtering which has to select the target word-string from all its competitors. Considerably fewer word-strings are derived in the phonemic sets, in which all utterances are parsed into less than 70,000 word-strings. But it must be remembered that these results

are based on hand transcriptions which represent a perfect analysis by the acoustic-phonetic component into whatever recognition units are used. If the acoustic waveform is incorrectly analyzed at certain points, the lexical access component may fail to derive the target word string from the imperfect string of recognition units. In this case, the syntactic/semantic component not only has to filter out many improbable word strings , but also to identify the location of possible word errors. Therefore, if

/th a ng k s f oo s e n d i ng **n** ii y oo l e t @/

were incorrectly derived from an acoustic waveform that in fact corresponded to an intended production of *thanks for sending me your letter*, the syntactic/semantic component would not only have to reject all the improbable competitors such as *thanks force ending knee your letter*, but also to identify that the word *knee* has been incorrectly derived instead of *me*.

6.5.2. Reducing the size

The results of this experiment would seem to preclude an analysis of the acoustic waveform by the acoustic-phonetic component into anything other than a single string of phonemes. Since, in reality, a phonemic string that is the output of phonetic processing of the acoustic waveform is likely to contain many errors, it is necessary to allow phonemic competitors to guarantee the inclusion of the correct phoneme. But if phonemic competitors are included, some means of resolving the phonemes identity and filtering out improbable words must be applied as quickly as possible to stop the proliferation of word strings.

In the following chapters we discuss possible ways of reducing the number of word strings. The methods may be roughly grouped under the following headings.

(i) increasing the specificity of the acoustic information.

(ii) using lexicon-based constraints.

(iii) bringing syntactic/semantic information to bear.

These options are not, of course, mutually exclusive. However research efforts in the past have tended to concentrate on one or the other.

The TRACE system has explored the use of mutually exclusive lexical constraints. It may be the case, as McClelland argues that this type of constraint requires a certain type of architecture.

The use of tight syntactic/semantic constraints with lexical access was fundamental to some of the recognition systems developed on the ARPA project, and indeed was part of the original specification of the task. (Klatt 1978, Lea 1980). In HARPY, for example, all higher level knowledge constraints are pre-compiled into a recognition network. Since no words are hypothesized other than those expected by the syntactic and semantic component at that point in the network, a large number of words in the vocabulary need never be considered as possible hypotheses. HWIM and Hearsay-II focus the search by predicting the words on either side of a seed word found bottom-up; only this subset of the lexicon is matched against the phoneme graph. In a less restrictive system such a top-down approach would not be feasible. In the Edinburgh system, all word possibilities are stored on the word graph and higher level knowledge is used only to filter out possibilities, rather than to dictate the set of possibilities. These and other search strategies will be examined in more detail in the next chapter. However, it is worth pointing out here why we have counted complete strings from one end of the utterance to the other.

6.5.3. Reducing the length

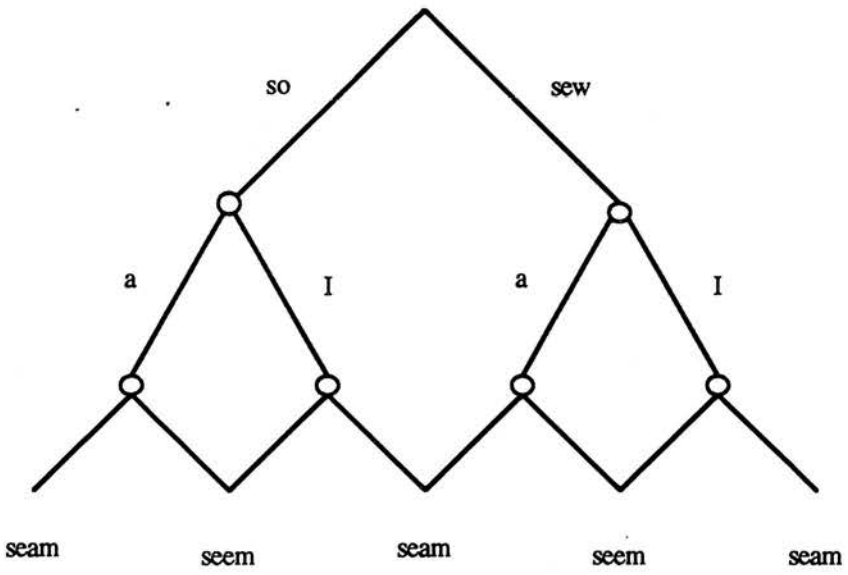

Figure. 6.7.

A word graph containing some of the alternative possible parsings of the phoneme string /s ou @/ai s ii m/.

One version of the Edinburgh syntactic component used pair-wise transition probabilities between words. Word pairs were scored on acoustic-phonetic probability and syntactic tag transition probability and all but the highest scoring hypotheses were pruned from the graph at each decision point. In Fig. 6.7 therefore, a choice has to be made at the top of the tree between *so* and *sew* based on the transitional probability scores to the following words *a* and *I*. If for example, *so* is preferred, the entire right hand side of the tree (i.e. the continued paths from *sew*) is no longer considered. Since this strategy enables the elimination of many word strings as the transition probabilities between successive words are calculated, it may provide a solution to the problem of the large

number of word strings that are derived when words are hypothesized from a mid-class input.

However, a knowledge of pair-wise transitional probability is often insufficient to select the target word string. In Fig 6.7 for example, the selection of the target (*sew a*) from the four possibilities *so a, so I, sew a , sew I,* must be based on a knowledge of the third word *seam (i.*e. we can only choose between *so a* and *sew a* once we know that the following word is *seam*).

If, as this example suggests, partial word strings of more than two words are necessary to guarantee selection of the correct word-string, then the number of alternative partial word strings that has to be considered at any one time will increase correspondingly, particularly when words are hypothesized from a mid-class input. In this chapter, we have assumed the worst possible case -- i.e. the syntax/semantics component needs up to ten words of the utterance in order to prefer one of the alternatives -- and this is why the statistics are based on the total number of complete word strings derived from the different kinds of input to the lexical access component. We do not think this is excessive since Pollack & Pickett report that human listeners sometimes require 140 csecs of conversational speech before reaching high accuracy. If we allow 200 msecs per spoken word, on average, then the human listener can require up to 7 words before being sure of an interpretation.

6.5.4. Number of paths vs average branching factor

There are a number of reasons why I have chosen to express these findings in terms of number of word-strings rather than in average number of words at each choice point (branching factor). The latter term is usually applied to the average number of word choices permitted by the system's grammar. As no grammar is used above -- any word can be followed by any other word -- the number of word choices in the system is *potentially*

equal to the number of words in the lexicon. That is, 4,000 x 4,000 words have to be checked against the acoustic, phonological and syntactic constraints.

As far as the *actual* word graphs output from a particular utterance, it is hard to give a clear answer to the question: what are the average number of choices on the word graph at any particular point? This is because it is not clear what we mean by "choice point". The test sentences had an average intended word length of about 7, and an average phoneme length of 26, but the word graph contained many overlapping words. Do we take the intended words as choice points? Every word boundary as a choice point?

Indeed, average number of words could be a misleading measure of the search complexity. To see the extent of the overlap effect, compare the following figures for two hypothetical seven word utterances:

1) Assume no overlapping words and an average number of competitors per word (average branching factor), $b = 4$, which is the average equivalence class size for a *broad* class lexicon given by Zue (1985). The number of paths according to our path counting algorithm would be b^n where n is the number of words in the utterance:

$$4^7 = 16{,}384.$$

2) Take the 7th root of the average number of paths through our *mid* class utterances in order to estimate the average number of words over each region of the intended word:

$$234.3^7 = 3.88 \times 10^{16}$$

The two numbers should not be compared because the first refers to words which have the same word boundaries, and the second refers to words in the same region. But both could be (confusingly) described as average number of competitors over the same region as the intended word.

Table 6.4 below gives a series of *n*th root of the average number of paths for the mid-class, mixed and fine-class sets. This gives a rough idea of the number of choices that have to be carried forward in order to produce word graphs of this size. For example, if after every 3 phonemes there are on average seven hypotheses of equal probability, the number of paths will exceed 5 million in an utterance of 27 phonemes.

	Av. Paths	1 ph. n=26	2 ph. n=13	3 ph. n=8	4 ph. n=6
MU	3.88 x 1016	4.34	18.88	118.46	581.84
XU	6,228,298	1.82	3.33	7.06	13.56
PU	1,791	1.33	1.77	2.55	3.38

Table 6.4.

Average choice points for each utterance type

6.6. Conclusions.

Although the study of Huttenlocher & Zue (1984) suggests that words in the lexicon remain highly discriminable when represented in broad-classes, or a mixture of broad-classes and phonemes, an excessively large number of word-strings can be derived when these kinds of phonological representations are parsed in continuous speech.

The results suggest that there is insufficient information in a mid-class representation (or average phoneme graph depth of around three equally ranked phonemes) for post-lexical processing to select the target word-string. Therefore, either the acoustic-phonetic component must reduce the phoneme graph depth further while still guaranteeing the

inclusion of the target phoneme in all successive vertical slots in the graph -- but it would be optimistic to suggest that this will be possible in the foreseeable future -- or else, lexical and post-lexical processing must be enriched to constrain further the alternative number of derived word strings. In the following chapters I will look at different methods of applying other sources of knowledge to constrain the search.

Chapter 7. Reducing the Size of the Word Graph

7.1 Introduction

In this chapter I discuss two possible approaches to reducing the size of the word graph:

(i) increasing the specificity of the acoustic information, and

(ii)) using lexicon based constraints.

We increased the acoustic information by incorporating lexical stress into the representation. This resulted in a decrease in the number of word paths, though the numbers were still extremely high. We incorporated lexicon based constraints by implementing a heuristic which preferred longer words over short ones. This also resulted in a slight decrease in the number of word paths, but often resulted in the loss of the correct word.

7.2. Increasing the specificity of the acoustic information.

As a means towards reducing the number of word paths, we considered the possibility of increasing the number of 'sound units' by using allophones in both the input utterance and the lexicon. The fact that the number of word paths should decrease using an allophonic representation is easily demonstrable. Phonemically, *plea* is represented as /p l

ii/ which also embeds the lexical item *lee*, phonemically /l ii/. On the other hand, *lee* would not be embedded within *plea* in an allophonic representation since these would be encoded as [li] and [pl̥i] respectively.

However, this advantage would be lost if the allophones that were the product of word-internal context-effects were also caused by context-effects across word boundaries: thus if /l/ in *lee* were realised as a voiceless [l̥] in a moderately fast production of *...tip leewards..*, *lee* would once more be embedded within *plea* even at an allophonic level of representation.

There is some experimental evidence (Bladon & Al-Bamerni 1976) to suggest that such word-boundary coarticulation of /l/ is possible. If the majority of identifiable allophones can occur as a result of coarticulation both across word boundaries and word-internally, the case for introducing this kind of phonetic representation is considerably weakened.

7.2.1. Lexical Stress

An alternative means of increasing the number of units in the input utterance, and thereby decreasing the number of word paths found, would be to include stress in the lexicon and input utterance. It is generally recognised that there is a great deal of information in the speech wave - particularly prosodic information - which we are not yet able to isolate and use. We therefore assumed some identification of lexical stress.

Three stressed sets of test utterances were included, derived from the corresponding unstressed sets. In the phonemic utterances, lexically stressed vowels were differentiated from lexically unstressed vowels by inserting a "*" symbol before the vowels of the former. No distinction was made between different levels of stress and only primary and secondary stressed vowels are differentiated from the remainder of the vowels. Thus, *conversation*, which is normally marked for secondary stress on /k o n/ and primary stress on /s ei/ is represented as /k *o n v @ s *ei sh @ n/ in stressed, phonemic form.With this

type of representation, in which each vowel phoneme (except schwa) can be marked for stress, an additional 19 units are introduced into the phoneme inventory. The unstressed mixed and unstressed mid utterances were converted to their corresponding stressed representations in the same way. The three different stressed representations for the utterance *The order goes in by late November* are shown below:

[The following abbreviations will be used: **Ps** (stressed phonemic); **Xs** (stressed mixed); **Ms** (stressed mid).]

/dh i *oo d @ g *ou z i n b *ai l *ei t n @ v *e m b @/ **(Ps)**

/NSF i *oo B @ B *ou z i N B *ai l *ei P N @ NSF *e N B @/ **(Xs)**

/V CV *BV B CV B *D Z CV N B *D L *D P N CV V *FV N B CV/ **(Ms)**

The method of generating and counting the word paths is as described in Chapters 4 and 5. For all types of input, the entries in the tree-structured lexicon have **Ps** representations [the lexical entries in the tree for *conversation* include, therefore, /k *o n v @ s *ei sh @ n/ (citation form), /k *o n v @ s *ei sh n/ (reduced form), /k *o m v @ s *ei sh n/ (reduced form)]. Accordingly, when either mid or mixed input utterances are matched against the lexicon, they are first expanded into all possible phonemic representations and each such representation is matched against the tree as described above. For example, the **Xs** representation of teaching, /P *ii ch i N/, (*teaching*) is first expanded into the forms shown in 1 - 9:

1 /p *i ch i m/

2 /t *i ch i m/

3 /k *i ch i m/

4 /p *i ch i n/

5 /t *i ch i n/

6 /k *i ch i n/

7 /p *i ch i ng/

8 /t *i ch i ng/

9 /k *i ch i ng/

and each of these forms is then matched against the lexicon. The corresponding **Xu** representation (/P ii ch i N/) would be expanded into all possible **Pu** and **Ps** forms (thus / p *ii ch i m/, / p ii ch i m/, / t *ii ch i m/, /t ii ch i m/ etc); each such form is then matched against the lexicon.

7.2.2. Results

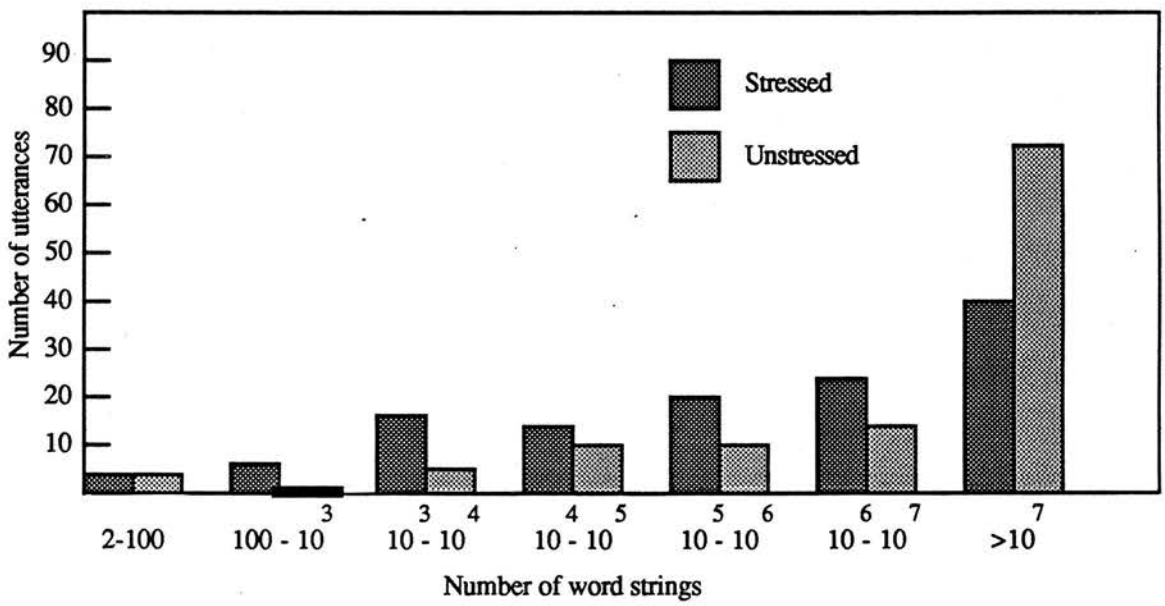

Figure 7.1.

Distribution of the numbers of word paths derived from the mid-representations (Ms & Mu) of the 115 utterances.

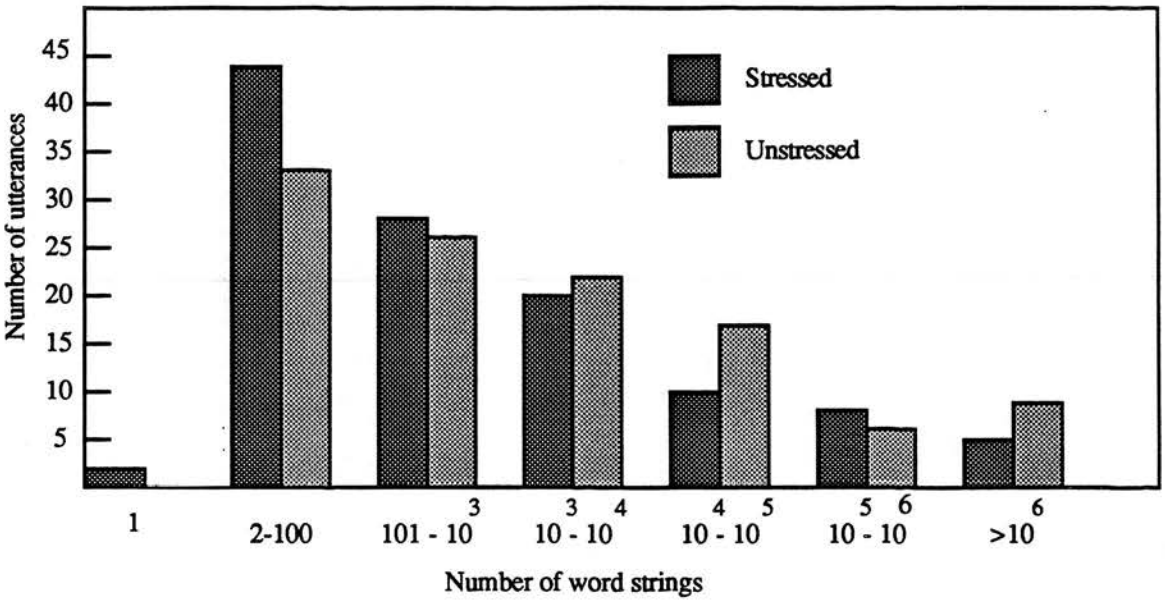

Figure 7.2.

Distribution of the numbers of word paths derived from the mixed representations (Xs & Xu) of the 115 utterances.

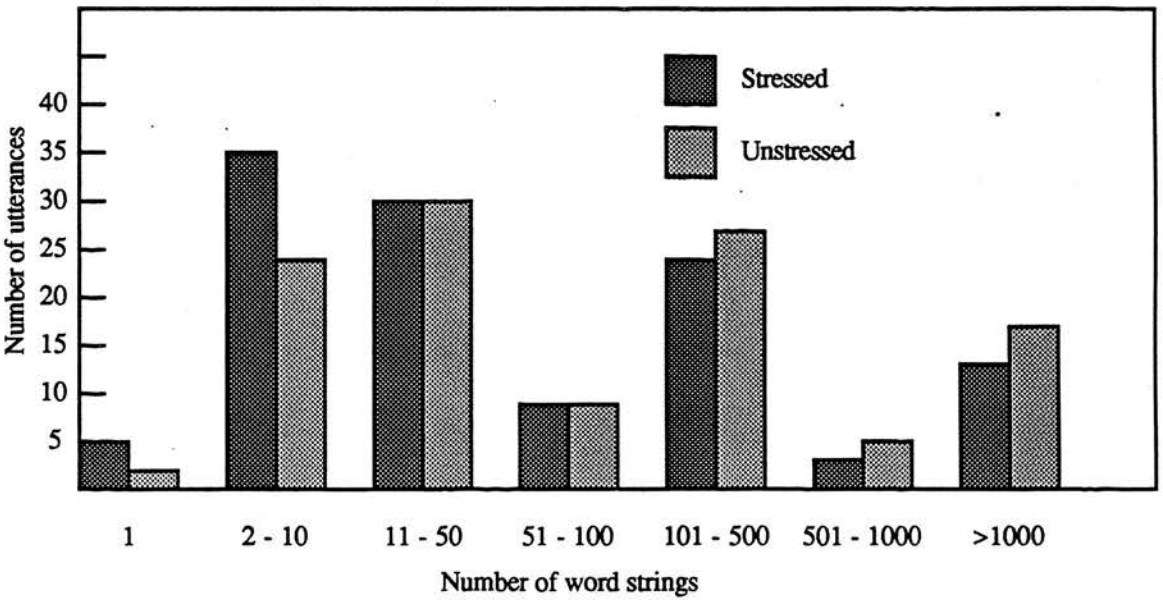

Figure 7.3.

Distribution of the numbers of word paths derived from the phonemic representations (Ps &Pu) of the 115 utterances.

Figures 7.1 - 7.3 show histogram distributions for the total number of derived word-paths when the input utterances are matched against the lexicon. I have reproduced the unstressed results from the last chapter for the purposes of comparison.

The results show that for **Ms**, 40/115 (35%) of the utterances were parsed into 10 million or more word-paths (Fig. 7.1), but only 13/115 (11.3%) utterances were parsed into 1000 or less word-paths. The average number of word-paths for **Ms** was 3.73 x 10^{13} . For **Xs**, 10/115 (9%) of the utterances were parsed into 100,000 or more word paths (Fig. 4.2), and 46/115 (40%) of the utterances were parsed into 100 or less word-paths. The average number of word-paths for **Xs** was 518,058.3. For **Ps**, 10/115 (9%) of the utterances were parsed into 1000, or more word-paths (Fig 7.3) and 40/115(35%) of the utterances were parsed into 10 or less word-paths. The average number of word-paths for **Ps** was 475.8.

The stressed sets produce fewer parsings into word-paths compared with their corresponding unstressed sets in all three cases. A paired sample t-test showed that there were significantly fewer word-paths derived from **Xs** compared with **Xu** ($t + 2.01$, significant at the 97.5% level for a one-tailed test) and fewer word paths from **Ps** compared with **Pu** ($t = 2.08$, significant at the 97.5% level for a one-tailed test).

7.2.3. Discussion

This study has shown that the inclusion of lexical stress reduces the number of word-paths derived from input utterances, significantly so for the mixed and phonemic sets. As suggested in Harrington and Johnstone (1987), the finding that fewer word-paths are derived from stressed utterances may be attributable to the fact that there are more phonemes in these stressed, compared with unstressed, representations.

The problem remains of whether the identification of stress is possible; some experimental work reported in Lea (1980) points to some progress in this field.

It is interesting to compare this work with that of Lee (1988) on Hidden Markov Modelling of speech recognition. Lee found that function words were a considerable problem for his SPHINX system. He writes:

> "Among the 684 errors in our system when no grammar was used, 334 were function word errors. Function words take up only 4% of the vocabulary, or about 30% if weighted by frequency, yet they are accountable for almost 50% of the errors." (p. 91)

Lee's solution was to train certain phonemes within the context of the function words. Thus the system would differentiate between a context independent unit for /a/ as in *band* and a unit that had been trained on just the instances of /a/ occurring in the context of such words as *and* or *an*. Lee selected 42 words (see Table 7.1) including some non-function words such as *find, show, give,* etc. Lee found a significant improvement in bottom-up word recognition -- 45.3% to 53.4%.

When we introduced lexical stress we found that all content words but only a small proportion of function words were marked for lexical stress, and that the number of word paths was reduced primarily because these function words no longer matched parts of content words. The selected function-word-dependent phone models in SPHINX are perhaps distinguished from content words by similar stress and co-articulatory information, which we have represented explicitly in the lexicon.

Of course, the stressed representations do not take account of the distribution of sentence stress in the utterances. Since *can* (auxiliary) is marked as unstressed in our lexicon, the parse of an utterance such as,*Yes, he can, sir* (with the emphasis on *can*) could contain confusions with lexical items such as *cancer , can,* etc.

A ALL	AND	ANY	ARE	AT	BE	
BEEN	BY	DID	FIND	FOR	FROM	GET
GIVE	HAS	HAVE	HOW	IN	IS	IT
LIST	MANY	MORE	OF	ON	ONE	OR
SHOW	THAN	THAT	THE	THEIR	TO	USE
WAS	WERE	WHAT	WHY	WILL	WITH	WOULD

Table 7.1.

The list of 42 function words that SPHINX models separately taken from Lee 1988.

7.3. Lexicon-based constraints

As we have seen many inappropriate parsings are constructed primarily from function words and their reduced forms. For example, since /@/, /i/, /d/, /m/, /z/, and /uu/ are reduced forms of *a/her, he, had/would, am ,is* and *who*, the short word parsings shown in Fig. 7.4 at the end of this chapter can be derived from *associated* and *Missouri farms* when the input utterances are matched against the lexicon.

A possible modification to the word parsing strategy, which is designed to eliminate many short word parsings, is shown in Fig. 7.5. In the first stage of this modified word-parsing strategy, only those matching words which are greater than two phonemes in length are stored on the word lattice; all other one and two phoneme words which match the phonemic input are stored in a temporary buffer. If a match to long words fails, then the short words are retrieved from the buffer and stored on the word graph at the appropriate point. In *I'm naming one man among many*, therefore, an initial attempt is made to match a long word to the sequence / ai m n ei../ (Fig 7.5.1). Since there are no long words that

begin in this way, the short word I, /ai/, is retrieved from the buffer and stored on the word graph (Fig 7.5.2). Subsequently, since there are no long words beginning with the sequence /m n ei.../, the short word *am*, /a m/, is retrieved and added to the word graph (Fig. 7.5.3).

Following *am,* both the long words *name* and *naming* can be matched (Fig. 7.5.3). However, since only the continued path from *naming* can be matched to a long word, the path from *name* is discontinued (without storing the short word *he* on the word graph). If there are several competing paths, only those paths that can be matched to long words are continued, and a short word will only be retrieved from the buffer if none of the possible paths can be matched to a long word.

It is clear that such a strategy would produce an incorrect parsing when a short word followed by a long word is homophonous with a long word followed by a short word. Consider for example, that /y oo r @ p l ai/ (Fig. 7.6) will be parsed as *Europe lie* rather than *your reply*, since /y oo r @ p l ai/ is matched to the long word *Europe* and / l ai/ is matched to *lie.*

Recovery from this sort of error is possible by increasing the size of the buffer. When continued parsing from a given long word fails, not only could all short words that immediately *follow* the long word be retrieved from the buffer, but also all short words that are *subsumed within the span* of the long word. Thus, in Fig 7.6.3 when the short word *lie* is retrieved from the buffer, so is the short word *your* which is subsumed by the preceding long word *Europe*. Paths could then be continued from all new short words that are stored on the word graph.

7.3.1. Results

This modified strategy was run over the same sets of mid, mixed and phonemic utterances reported in the previous section. The results in Table 7.2 clearly show a reduction in the number of word-paths for all sets; however there were also many utterances for which the correct word-path was not included as one of the alternative word-paths (Table 7.3). The correct word-path is excluded from many mid and mixed utterances because an alternative parsing into a long word is preferred. For example, the mixed representation of *a brief account*, /@ B r ii NSF @ P au N P/, will be parsed into *agree the count*, because the first four phonemes can be matched to the long word *agree* and because there is a possible parsing into words of the remaining phonemes.

		Modified	Original
Mid	S	1,555,278	$3.73 * 10^{13}$
	U	1,036,274	$3.88 * 10^{16}$
Mixed	S	4,106.7	518,058.3
	U	6,455.3	6,228,298.1
Phonemic	S	181.1	475.8
	U	315.0	1,790.9

Table 7.2.

Mean number of words for two parsing strategies

	Stressed	Unstressed
Phonemic	2 (1.7%)	4 (3.5%)
Mixed	3 (2.6%)	12 (10.4%)
Mid	27 (23.5%)	59 (51.3%)

Table 7.3

Incorrectly parsed utterances using the modified parsing strategy

7.3.2. Discussion

Other systems have used different strategies and have obtained similar results. In the SPHINX system, for example, the HMM's erroneous assumption about independence between adjacent phonemes means that the acoustic evidence is underestimated. That is to say, the Bayesian multiplication of adjacent phoneme probabilities penalises longer words unnecessarily. SPHINX uses a *language model match factor*, to compensate. If this parameter is set high, SPHINX prefers longer words to shorter ones. Lee writes:

> "In general, we found that most of the errors are reasonable confusions between similar words or sequences of words, such as *arriving -> arrive in, were in -> weren't, on first -> Connifer's, that are in -> centering*. These types of errors are most frequent when no language model was used, because there were many more combinations of word sequences that may be confusable. When the language model match factor ... was large, these errors tended to swallow smaller words (or phones) into larger ones; when the language model match factor was small, these errors tended to split larger words (or phones) into smaller ones. We have found that better recognition was obtained with a larger language model match factor, which prefers longer words over shorter ones, and deletions over insertions." (p. 119)

We discuss SPHINX's performance in more detail in Chapter 8.

Much interesting work has been done in the PDP paradigm on the use of lexicon-based constraints to reduce the word graph size. The buffer-based heuristic implemented in the Chart could probably be recast in terms of a delayed activation model, or an activation-inhibition model by having weighted links between edges. If level of activation rather than presence on the Chart defined a word's status, we could perhaps more easily allow top-down effects to over-ride the preference for longer words.

The TRACE model uses activation and inhibition to suppress certain words. In the case of *party*, for example, *par* will start off at a higher activation level than *party* because of inhibition effects, but as the model begins to process /t ii/, *party* will be sufficiently active to suppress *tea*.. McClelland & Elman (1986) write:

> "In 189 of the 211 word pairs tested in the simulation experiment, the model came up with the correct parse, in the sense that no other word was more active than either of the two words that had been presented. Some of the failures of the model occurred in cases where the input was actually consistent with two parses, either a longer spanning word rather than a single word (as in *party*) or a different parse into two words, as in *part rust* for *par trust*. In such cases TRACE tends to prefer parses in which the longer word comes first. There were, however, some cases in which the model did not come up with a valid parse, that is, a pattern that represents complete coverage of the input by a set of nonoverlapping words. For example, consider the input /parki/. Though this makes the two words *par* and *key*, the word *park* has a stronger activation than either *par* or *key*." (p64).

To summarise, we used a large lexicon and a sizeable number of utterances, to test an heuristic which preferred long words over short ones, and found that the strategy reduced the number of word paths in all cases. However, the number of missing words was unacceptably high in the mid-class case.

7.4. Conclusions

We tested two strategies for reducing the word graph which used different kinds of information. In the first test, we applied acoustic-phonetic constraints by introducing lexical stress into the representations. In the second, we applied lexicon-based constraints by implementing an heuristic which preferred longer words over short ones. Although both strategies eliminated some incorrect interpretations, the number of word paths was still very high, especially in the mid-class case.

It is clear that the derivation of the high number of word paths from mid-classes and the problem of filtering them out at the lexical access stage means that syntactic/semantic information must be brought to bear as soon as words are accessed. We shall discuss the application of higher level constraints in the next chapter.

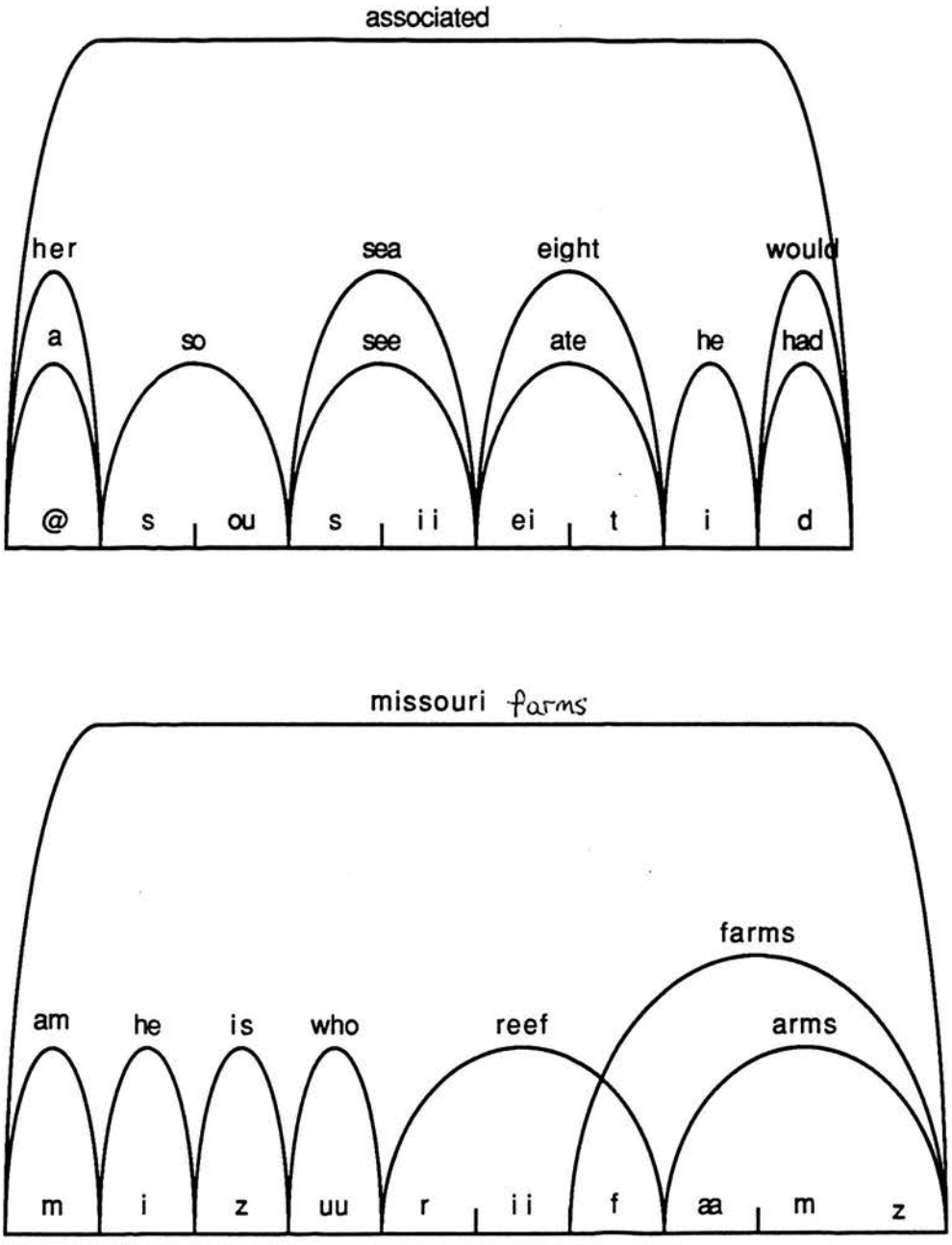

Figure.7.4.

Some of the possible short word parsings of associated and Missouri

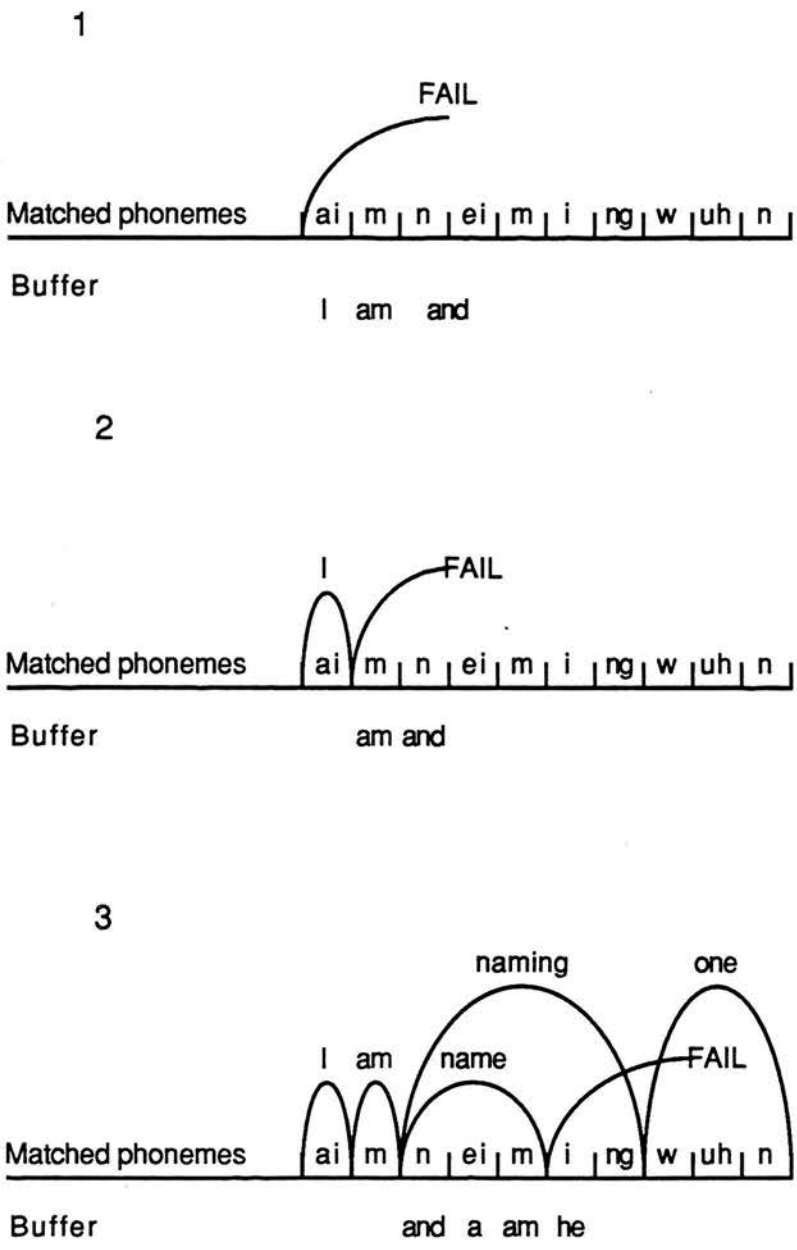

Figure. 7.5.

The modified parsing strategy in which words of greater than 3 phonemes are preferred over shorter words

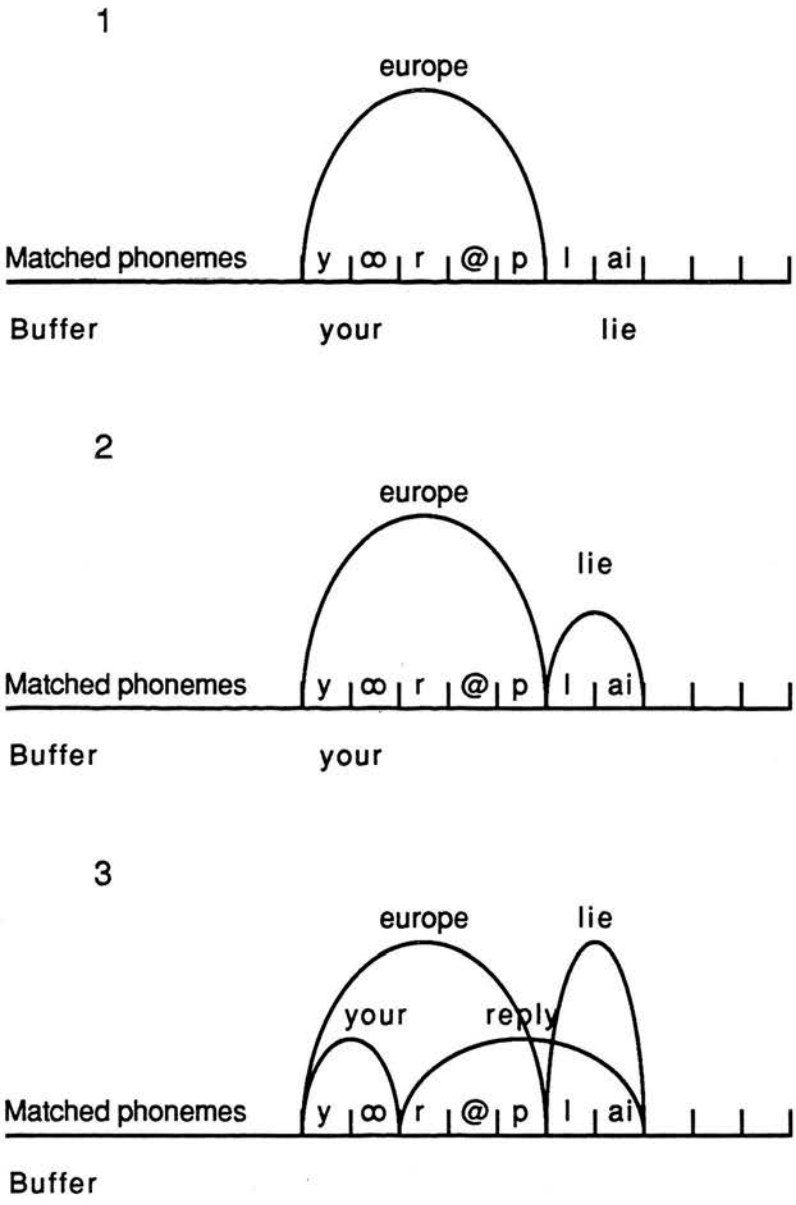

Figure 7.6.

The parsing strategy in Fig. 7.5. modified to retrieve from the buffer all words subsumed by the long words.

Chapter 8 Search Strategies

8.1. Introduction

The experiments in the previous chapter showed the effects that word boundary ambiguity can have on the size of the word graph given different phonemic inputs. We now face the important question, how do these results affect our understanding of the search problem in speech processing?

This chapter will analyse the abstract nature of the search space drawing on relevant aspects of state-space search theory. I will discuss the two major admissible search algorithms used in state-space search -- breadth first and A* -- and the complexity issues associated with each algorithm.

The following chapter will look at specific strategies in the context of the speech search problem. I will examine the strategies used in HARPY, HWIM and two more recent systems, SPHINX and RM1, and discuss how each system has adapted one or the other of the basic admissible algorithms in order to reduce the potential combinatorial explosion of word string hypotheses. The results of Chapters 5 and 6, together with the abstract analysis of graph search in this chapter, will give us a clearer understanding of why these strategies performed as they did.

8.2. Admissible Search Algorithms

The most efficient search algorithm to use is determined by (i) the goal of the search and (ii) the nature of the search space. Let us say that our goal is to find the highest scoring[1] interpretation spanning the utterance. In that case, it is usually best to use an algorithm which has been proved *admissible*. Admissible algorithms guarantee that the first spanning path found will be the highest scoring. If the algorithm is not admissible then the decision to end the search is fairly arbitrary. Even if the best of the top n interpretations is taken, there is no guarantee that it is the best of all possible interpretations.

The *type* of admissible algorithm used depends on the nature of the search space. There are two standard classes of A.I. algorithm which are guaranteed admissible: (i) breadth-first strategies which explore every possible path in pseudo-parallel, and (ii) A* type algorithms which try to take the score of the whole path into account. Most speech systems have used variations on these two types of admissible search algorithm. However, breadth-first and A* are not unrelated. In the next section we will look at the relationship between breadth-first and A*, and a third algorithm, uniform cost, which is also related.

8.2.1. Breadth-first search

Breadth-first search extends all the paths at one level of the search tree (the siblings) before going on to the next level (the descendants). The space requirement is an exponential function of the length of the path at any time. If the goal of our search is to find the highest scoring path that spans the complete utterance a straightforward breadth-

[1] Speech literature usually takes 'best' to mean highest score. Search literature refers to the best path as having the lowest cost. It should be clear from the context which sense is being used.

first algorithm would have to check approximately b^n paths, where b is the average number of words at each choice point and n is the length of the utterance. In other words, a breadth-first search of one of the mid-class word graphs could produce a tree with millions of terminal nodes representing each of the equally valid word-string paths.

8.2.2. Uniform Cost

Rather than explore a tree in layers of equal depth like breadth-first, uniform cost expands nodes in layers of equal, cheapest cost. c(n, n') is the cost associated with a path from node n to n'. If the costs on the paths are non-decreasing with length then this algorithm is guaranteed to find the cheapest path first. When all paths have the same cost associated with them, equal cost will be synonymous with equal depth and the algorithm will perform breadth-first.

8.2.3. The A* Algorithm

The A* algorithm is given in Pearl (1984, p 64) as follows:

1. Put the start node *s* on OPEN.

2. If OPEN is empty, exit with failure.

3. Remove from OPEN and place on CLOSED a node *n* for which *f* is minimum.

4. If *n* is a goal node, exit successfully with the solution obtained by tracing back the pointers from *n* to *s*.

5. Otherwise expand *n*, generating all its successors, and attach to them pointers back to *n*. For every successor *n'* of *n*:

a. If n' is not already on OPEN or CLOSED, estimate $h^*(n')$ (an estimate of the cost of the best path from n' to some goal node), and calculate

$f(n') = g(n') + h^*(n')$ where $g(n') = g(n) + c(n, n')$ and $g(s) = 0$.

b. If n' is already on OPEN or CLOSED, direct its pointers along the path yielding the lowest $g(n')$.

c. If n' required pointer adjustment and was found on CLOSED reopen it.

6. Go to step 2.

A formal proof of the admissibility of A* can be found in Pearl (op cit.) or Nilsson (1971). Informally, the idea is that every path in a graph which passes through a node n can be thought of as having a score $f(n)$ consisting of the cost from the start of the search to that node $g(n)$, and the cost from the node to the terminal state, $h(n)$. Since $h(n)$ is not known during the search we can try to estimate it in some way. The algorithm chooses for expansion the node having the best actual score so far, together with the best estimated score, $h^*(n)$. If the estimate of the cost is guaranteed always to be optimistic, i.e. less than or equal to the actual cost for that path, we are guaranteed to find the optimal path, since no node will be left unexplored for which

$g(n) + h(n) < C$

where C is the cost of the best path through the search space. To see how this works, let us assume that we have reached the penultimate node in the search space, as in Figure 8.1 below.

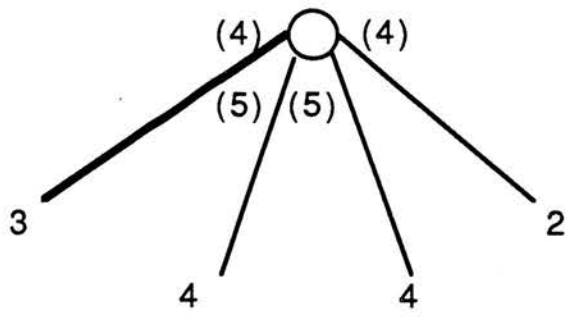

Figure 8.1

The estimated costs, h*(n), are given in brackets. In this case the estimate is pessimistic: h*(n) is greater than the actual cost, h(n).

If the estimates of the costs on paths from the penultimate node to the terminal node are *greater* than the actual costs on each path, as in Fig. 8.1., then the algorithm might be misled into taking a non-optimal path. Even if the actual cost turned out to be quite high, it might still appear better than the other paths for which we have high estimates, estimates which may be considerably greater than their actual cost. In Fig. 8.1. the algorithm is misled into taking the leftmost path because its actual cost, 3, is less than the estimate, 4, (though not the actual cost, 2) of the correct, rightmost path.

If, on the other hand, the estimates are always *equal to or less* than the actual cost for a path, as in Fig. 8.2., then even if the algorithm initially takes the wrong path, its actual cost will be worse than the other, optimistically estimated paths. It will therefore backtrack to try those paths.

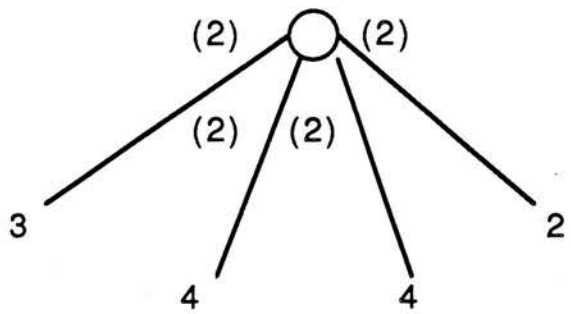

Figure 8.2

In this case the estimate is optimistic: h*(n) is equal to or less than the actual cost.

In Figure 8.2 the algorithm will initially take the leftmost path but will then backtrack and eventually take the correct, rightmost path, because the estimate for the rightmost path is less than the actual cost of any other path.

The value of this algorithm lies in its behaviour on off-track nodes, i.e. nodes not on the correct path. Whenever the cost of such a path increases, the algorithm will backtrack to a better looking node. In general, the closer the heuristic estimate to the actual path costs, the fewer off-track nodes will be explored.

8.3. Complexity Issues

At first sight A*, and breadth-first search appear to be quite different: the former tries to keep close to the correct path by comparing costs, the latter blindly and exhaustively explores each level. However, it can be shown that breadth-first and uniform-cost are just special cases of A*.

Uniform cost is equivalent to A* with the heuristic estimate $h^*(n) = 0$. Thus the cost of a path is:

$f(n) = g(n) + 0$

and the algorithm is guided just by the cost so far, $g(n)$.

Breadth-first behaviour is obtained when $h^*(n) = 0$ and the cost $c(n,n') = 1$ for each branch. Uniform cost is now the same as uniform depth.

Thus under certain conditions we can expect A* to behave in the same way as breadth-first search. If $h^*(n)$ is wildly optimistic (i.e. the cost estimate is essentially 0) then the algorithm will be guided by $g(n)$ the cost computed so far.

If many paths pick up additional cost at each expansion, then the cost of a path will increase with its length. Under these conditions, however good the heuristic estimate, the algorithm will keep abandoning paths that fail to live up to their initial promise in favour of untried paths that are promising a little more than they will deliver (Pearl 1984). Equal cost will start to look like equal depth (i.e. breadth-first search) and the algorithm will explore a broad band of hypotheses.

Pearl (1984) proves that, for a binary tree model whose branches have a cost of 1 or 0 with probability p and 1-p respectively, if $p > 1/2$ then any admissible algorithm will run in exponential time.

In the section on specific speech systems, we shall see under what conditions the speech search problem is of this type. If the search problem in speech is indeed such that A* starts to behave like breadth-first, then the only way out of the combinatorial explosion produced by breadth-first search is to keep the search tree small.

8.4. Reducing the Search Space

The number of paths through a search tree is b^n where b is the average branching factor, and n is the length of the path to a solution. We can think of the word graph as a search tree with each path through the tree corresponding to each of the word strings. b would correspond to the average number of word choices at each choice point, and n would correspond to the average number of choice points in the utterance. However, we have already noted that the definition of "choice point" is not easy. We shall return to this point later.

One way to reduce the size of a search tree is to split it into smaller sub-trees representing more manageable sub-problems. The sub-problem must have certain properties if the correct solution to the whole problem is still to be found. It must be:

(i) *self-contained* -- it must not need information from another part of the problem.
(ii) *small* -- a solution must be recoverable within the resource limits of the system.
(iii) *solvable* -- a solution to the sub-problem, preferably a unique one, must exist.

This kind of problem reduction can be performed over stretches of speech if we can define the problem in such a way that the stretch is *self-contained.* That is to say, the lexical interpretations are supported by the same acoustic evidence and need no further acoustic information, and the syntactic/semantic interpretations require no further acoustic-phonetic, lexical, prosodic, or other information. A stretch such as /m e n / would not be self-contained if the system allowed utterances to begin with *Men.. Many.. Mental...* It would be self-contained if the system only allowed *Men..*

8.4.1. Backwards Pruning

One way to define a sub-tree is to reduce the depth to a partial solution. *Depth* in the context of the speech problem can be taken to mean *time*. If a decision can be made about a partial solution after every three words, say, then a search space with two words at every choice point could be searched with $2^3 + 2^3 + 2^3 \ldots = 24$ paths instead of $2^3 \times 2^3 \times 2^3 \ldots = 512$ paths, an exponential saving. We will call this backwards pruning because, at the point where more than one path has the same successor, the algorithm can look back the way it has come, and mark or retain only the highest scoring path.

These decision points can be thought of as nooses that pull together two or more nodes in a tree. (See Fig. 8.3). They reduce the average branching factor and hence the potential combinatorial explosion of paths through the graph. But they also reduce the flow of information through that node. For example, a decision after the competing pair of hypotheses*so a*/*sew a* would mean that only the path *so a* was available by the time the competing pair *seem*/*seam* was processed.

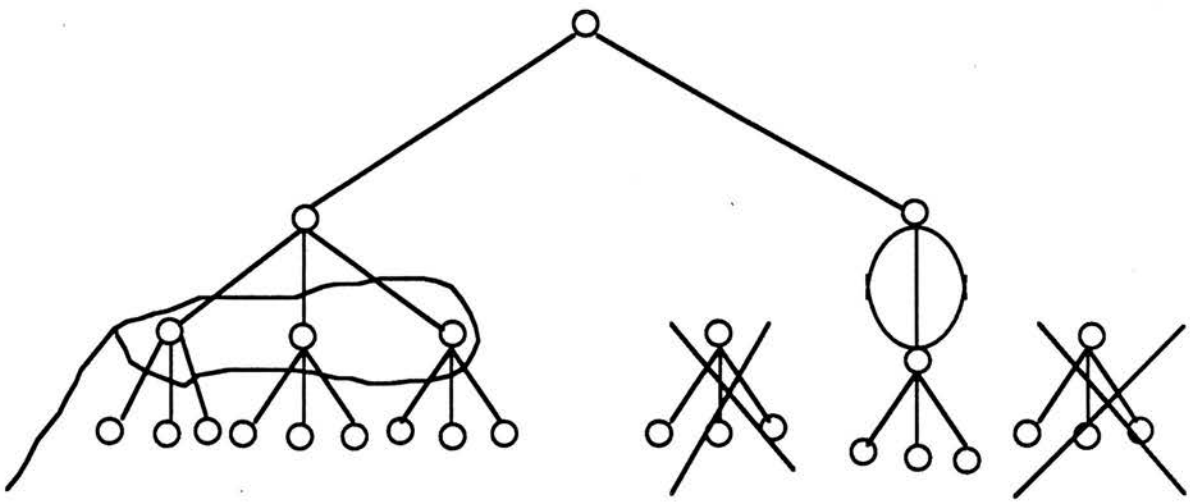

Figure 8.3.

Joining nodes which have the same successors gives exponential savings.

If some paths have the same successor then the tree can be redrawn as a graph as in the righthand side of Fig. 8.3 and in Fig. 8.4. The backwards pruning decision point is marked by line A in Fig. 8.4. At each such decision point only the highest scoring path need be retained since what follows has no influence on the scores of paths leading up to A.

Backwards pruning decision points can often be determined off-line by analysing and structuring the problem in a certain way. For example, HARPY uses a finite-state machine which allows many sub-parts of an utterance to be treated as a self-contained recognition problem.

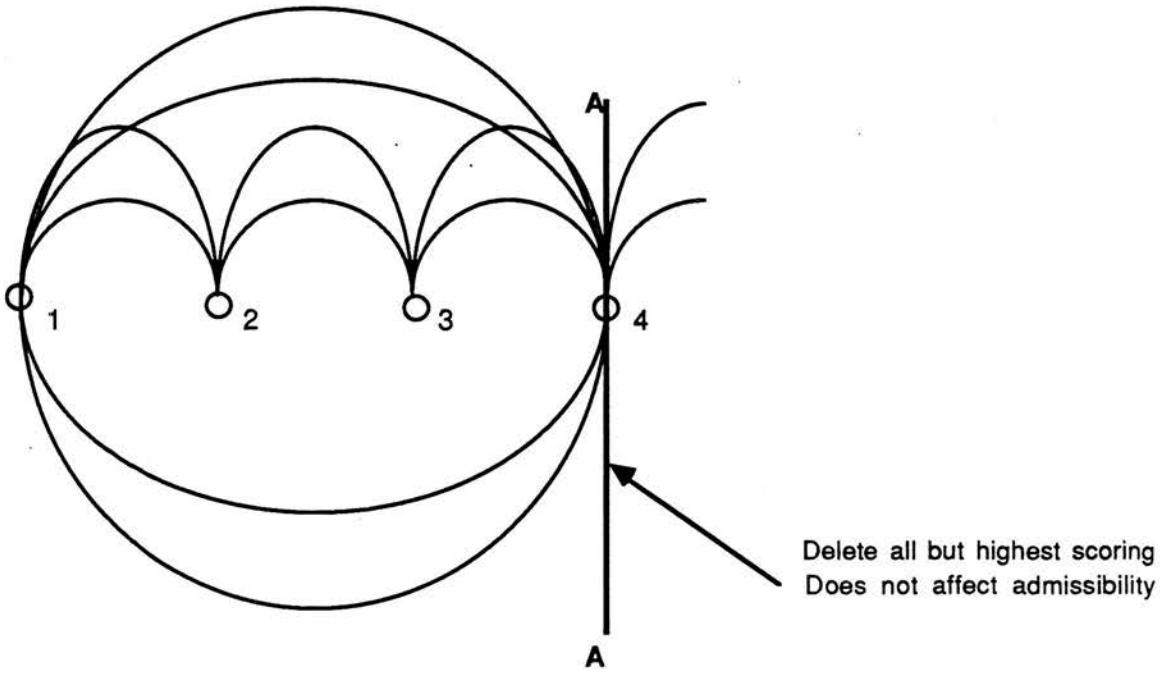

Figure 8.4.

Backwards pruning decisions do not affect the optimum path

We shall discuss ways of structuring the speech problem in more detail in the next chapter. But looking at the diagram in Fig. 8.4 we can identify two general methods of keeping the search space *small* :

1) Reduce the average branching factor by keeping the number of arcs spanning n(1,2), n(2,3), and n(3,4) low and thus limit the combinatorial explosion of paths.

2) Reduce the depth to a (partial) solution. The more frequent the decisions, the smaller the search space. If we could place decisions at nodes 2 and 3 as well as at node 4 there would be 5 paths through the graph instead of 12.

In general, the following savings apply:-

1) The number of paths in a tree with branching factor b at depth d $= b^d$

2) The number of paths in a graph with a join at depth j $= b^d - b^{d-j}$

3) The number of paths in a graph with n joins at depth j $= b^d - n(b^{d-j})$

4) The number of paths in a graph with a join of m nodes at depth j

$$= b^d - (m-1)b^{d-j}$$

As we noted above forwards pruning decisions must be *solvable* if the method is to usefully reduce the search space. That is, the paths leading up to the decision point must be differentiated by score. If they all have the same score then the decision to carry just

one forward is arbitrary, or they must all be carried forward just as though a tree were being searched.

Secondly, the decisions must be about units which are *self-contained* according to the problem definition. That is, any information after the decision point must be irrelevant to the question which is the best path leading up to the decision point. An admissible algorithm such as breadth-first or A* is guaranteed to find the optimal path through a graph such as the one in Figure 8.4. However, we may decide, after the fact, that the optimal path through the graph is not the answer we want. That is, we may find that we do need a flow of information through the backwards pruning decision points, that we need information about *seem/seam* to make a decision about *so/sew*.

8.4.2. Forwards Pruning

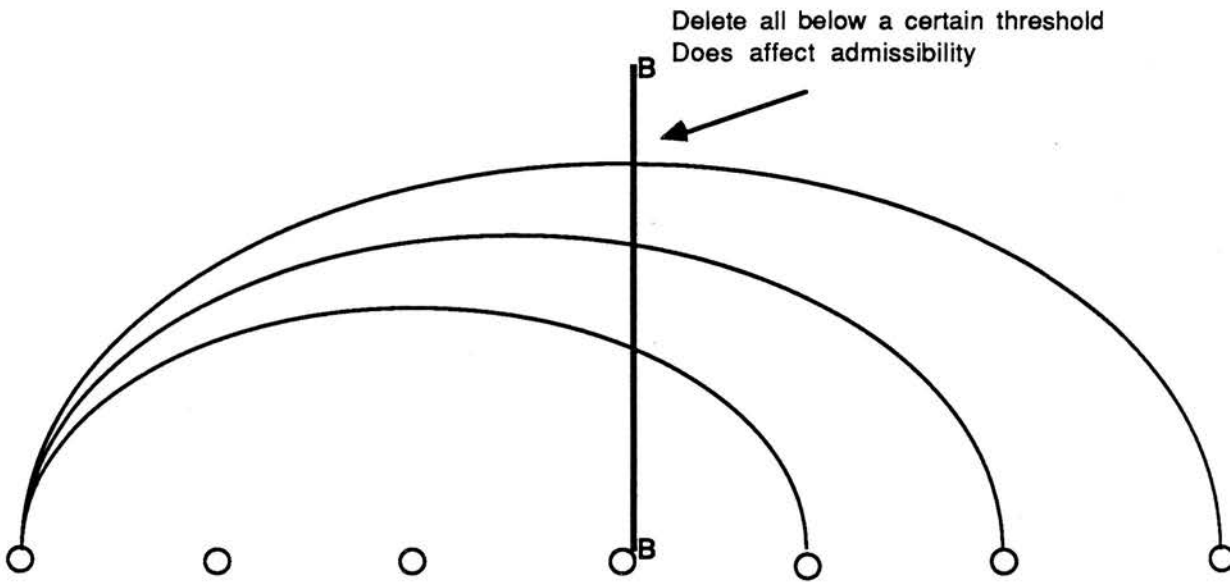

Figure 8.5.

Forwards pruning may prune the optimum path

The search space may be such that we need to prune paths early before we have all the information relevant to their identification. At decision point B in Fig. 8.5. we do not know the full scores of the paths, but we can try to look ahead and estimate them or extrapolate from current scores. A* algorithms make a decision at point B about which path to follow up. However, the algorithms are admissible because they do not prune any of the arcs at B, and may backtrack to them later.

Some algorithms forward prune any paths which fall below a certain threshold. This method, which is commonly known as *staged search* , does affect the admissibility of the algorithm. There is no longer any guarantee that the optimal path through the graph will not be pruned. However, if certain conditions hold, it can be shown that the algorithm is near admissible. If the search space is such that the optimal solution is almost always within a narrow band of competitors throughout the length of the path, then all the paths which fall outside this band can be pruned (see Pearl 1984).

8.5. Conclusions

We have looked at two admissible graph search algorithms: breadth-first and A*. We have noted that, given a certain type of search space, the A* algorithm may behave in the same way as breadth-first search, since the latter is a special case of the former.

We saw that the combinatorial explosion of hypotheses produced by breadth-first search could be limited by cutting down the depth or the branching factor of the search space, and we described two general methods of doing this:-

(i) Backwards pruning: the problem is formulated in such a way that the average branching factor and/or the depth of the tree are kept low. Decisions can be made

periodically on the basis of information gathered so far. An admissible algorithm can then be used to find the optimal path *according to the problem definition.*

(ii) Forwards pruning: estimates are made on-line of the possible utility of later information. If the estimates are used to prune paths, the algorithm is no longer guaranteed to find the optimal path, though it may be possible to prove that the probability of doing so approaches 1. Again, it must be born in mind that the definition of optimal path depends on the problem.

Finally we discussed the conditions under which each method could best be applied.

We shall now examine the performance of four actual speech processing systems and explain their performance in the light of this theoretical discussion. We will show how these systems use branching factor, depth and scores in attempts to limit the search, both by formulating the recognition problem in a particular way, and by pruning during the search process itself.

Chapter 9 Complexity Issues in Speech Processing

9.1 A* or Breadth-First?

We are interested first of all in the question, is the search space in current speech processing such that A*, or indeed any admissible algorithm, behaves like breadth-first search? If this is the case, then we must concentrate on formulating the problem in such a way as to avoid the combinatorial explosion of hypotheses associated with unrestricted breadth-first search.

9.1.1. HWIM's shortfall algorithm

In the case of the HWIM system the answer seems very clearly to be "yes". The first admissible algorithm[1] used in HWIM was the *shortfall algorithm* (Woods 1982). This algorithm computes the shortfall score of a hypothesis $f(n) = g(n) + h(n)$ [see the A* algorithm above] from its score so far:

$g(n) = m(n) - q(n)$

where $q(n)$ is the acoustic probability score of the hypothesis and $m(n)$ is the maximum probability achievable by any hypothesis for this region. The estimated shortfall is computed as:

[1]Full details of the algorithm and the scoring methods can be found in Woods (1982). The argument presented here applies to *any* admissible algorithm, so we are mainly concerned with the fact that it can be proved to be admissible.

$h^*(n) = T - m(n)$

where T is the maximum score possible for the whole spanning hypothesis. Hypotheses are ranked by decreasing shortfall, the goal being to find the path with the smallest shortfall covering the whole utterance. As the estimate is simply the maximum that any hypothesis can achieve for the remainder of the utterance, the estimate is always optimistic and the algorithm is admissible.

In all trials, the shortfall algorithm failed to produce an interpretation. Woods writes:

> "Unfortunately, experience with the HWIM system has shown that the shortfall algorithm is excessively conservative. It amounts to assuming that any theory will obtain the maximum possible scores in the regions not yet covered. This is clearly overly optimistic in almost all cases, and it in fact leads to an excessively breadth first search." (Woods 1982 p304).

Woods assumes here that the estimate, $h^*(n)$, is at fault in being too optimistic. Short hypotheses looked more promising than long ones simply because more of their score was an estimate; they had not picked up actual shortfall.

However, as we have seen in the previous section, there may be an alternative explanation for the breadth-first behaviour. If there are too many hypotheses with the same or very similar scores, and the cost of the path increases with its length then the algorithm will explore the search space on a broad front, regardless of the accuracy of the estimate. But first let us follow up the possibility that the problem *is* entirely to do with the estimate.

9.1.2. Improving the heuristic estimate

Woods tried to improve the estimate by devising the *shortfall density algorithm*. This averages the current score of a path over its length. The score of a path is calculated as:

$f(n) = [d(n) * l(n)] + [d(n) * L(n)]$

where $l(n)$ is the the length of the utterance covered by the current hypothesis, $L(n)$ is the length of the remainder of the hypothesis, and $d(n)$ is the shortfall score divided by $l(n)$. Since $l(n) + L(n)$ is a constant, only the $d(n)$ term of each hypothesis need be considered.

Dividing the current score by its length means that shorter hypotheses no longer necessarily have a much more optimistic estimated score than longer ones. This should reduce the breadth-first tendency of shortfall algorithms.

But since only the score achieved so far is used, no account is taken of the possibility of right context effects increasing the score of the whole path. Later information could score better than the path so far, the estimated score (extrapolated from the current score) therefore no longer functions as a reliable upper bound, and so the algorithm is no longer admissible.

Woods solution was to use this algorithm with the island-driving search strategy. Since this strategy works outwards from the highest scoring words, it is possible to show (see Woods 1982) that islands always incorporate words which have a density score no greater than the words already in the island. Thus the cost of a path cannot increase, it can only stay at the same level or decrease, and the algorithm is admissible.

Woods reports that the shortfall density algorithm is superior to the shortfall method alone, returning the correct interpretation in 5 out of 10 trials. The reason for the density method's increased efficiency is that there has indeed been some improvement in the estimate of $h^*(n)$. The algorithm is not continuously diverted from depth-first into breadth-first search simply by virtue of picking up additional cost, as is the shortfall method. Provided the correct path does in fact score markedly better than other paths, the algorithm will return the correct solution without exploring the entire search tree.

How do we explain the instances when the density algorithm failed to return an interpretation as was the case in *half* of the trials? Woods writes of the shortfall density method:

> "This promotes the refocusing of attention from a region where there may happen to be high quality accidental word matches to events whose word match quality may not be as great, but are the best matches in their regions. If this were not done, then many second best, third best, etc matches in the high scoring region could be considered before any theories worked their way across the low scoring regions." (Woods 80 p306)

Woods seems to assume in the passage cited above that there are only a few regions where high quality accidental word matches take place. But as we saw in Chapters 5 and 6 there may be very many extraneous word strings which are homophonous with the correct words, and which extend some if not all of the way through the utterance. Although HWIM's lexicon contained only 1000 words, the grammar was very general: an ATN with an estimated branching ratio of 196. This high branching ratio together with the fairly poor performance of the acoustic-phonetic component could easily produce lattices with very many overlapping, highly probable word strings. Many of these will include one or more of the intended words. So the middle-out density algorithm may proceed breadth-first on many island fronts trying out possible combinations of phonemically equivalent words.

This hypothesis about the reason for the density algorithm's failure is born out by Paxton's results (Paxton 1977). He found that middle-out methods decreased performance on longer sentences solely because of space requirements, not because of poor hit-rate of initial seed words. He writes:

> "The bad effects of island driving on the long sentences were not caused by an increase in the number of false alarm seeds. The average rank of the first hit in the sequence of words for use in forming islands was 4.8, and the rank did not increase with sentence length... For sentences 1.7 seconds or longer, instead of an increase in the number of seeds necessary to get a hit, there was an increase in the amount of storage consumed per island. Perhaps the greater length allowed the islands to grow in both directions, whereas in shorter sentences the sentence boundaries blocked one direction or the other." (p. 204)

We noted in Chapter 4 that it is hard to predict which sentences will have very many overlapping interpretations, that it depends on many factors including the phonemes in the utterance, the content of the lexicon, and so on. I would conjecture that the longer sentences tested by Paxton, and the unsuccessful sentences in the HWIM trials happened to contain many alternative interpretations. These words would all have the same or a very similar score and would combine exponentially into word paths.

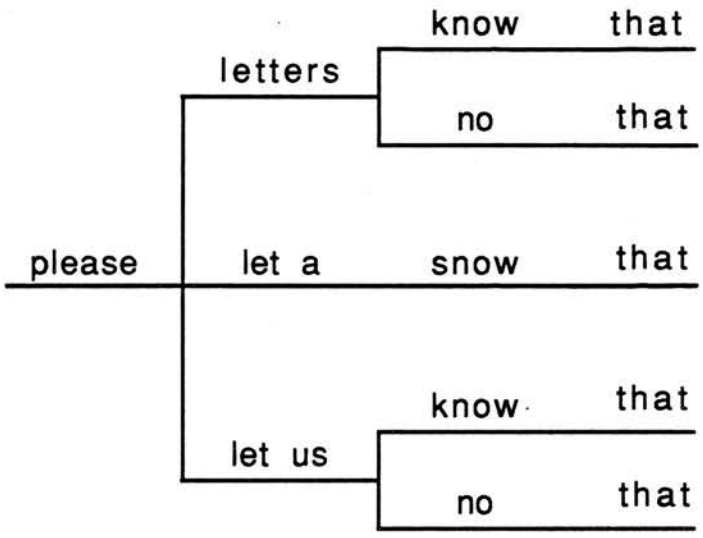

Figure 9.1

A partial tree of word strings

Let us see how the shortfall algorithm would search the tree in Fig. 9.1. Assume all the branches, except the ones with *that*, have an actual cost of .2 and an estimated cost of .1. *that* has an actual cost of .3 and an estimated cost of .2. At the beginning, all the paths would have an estimated cost of .1 + .1 + .1 + .2 = .5. Whichever path the algorithm

took, that path would pick up shortfall or cost of .1. Its path score would be .2 + .1 + .1 + .2 = .6. which is greater than the estimated cost, .5, of any of the other paths. Thus, the algorithm would backtrack to one of the apparently cheaper paths and extend that one. Even if the estimated cost of a branch costing .1 was .0999999, the algorithm would backtrack, exploring the search space breadth-first.

The shortfall density algorithm would pursue one interpretation depth-first until it came to an area that scored less well, such as *that.*. The cost of the path would increase to (.6 + .3)/ 4 = .225 as a word from this region was incorporated, and the algorithm would backtrack to one of the apparently better extensions where the cost was .8/4 = .2. Each path would be expanded in a depth-first, backtrack, depth-first, backtrack pattern until each possible combination had been extended over the poor patch.

It is important to realise that we are not taking into account all the competing paths produced by poor acoustic-phonetic labelling. These may well be ignored by the algorithm if they score sufficiently poorly. We are concerned just with the exponentially increasing "top band" of word paths which are homophonous with the correct interpretation.

In addition to the acoustic-equivalence problem, we would also expect the island-driving aspect of the density algorithm to exacerbate the search problem. In Chapters 5 and 6 we counted only those paths which spanned the entire utterance, anchoring the beginning of the search at the left-hand end and ignoring all paths that failed to match through to the right-hand end. An island-driving algorithm *will* explore those paths that ultimately fail to match the beginning and ending of the utterance. We would expect an anchoring of the path at one end or the other to reduce at least a portion of the breadth-first search. Paxton's remarks in the above passage seem to support this supposition.

9.1.3. Conclusions

Thus it does appear that the search space can be such that an admissible algorithm will perform in an excessively breadth-first manner. Many utterances can be parsed in a number of ways which are phonemically similar though lexically different. This uncertainty about lexical identity which is reflected in the acoustic scores, means that, even when a good heuristic estimate is used, an admissible algorithm will proceed breadth-first on a large front.

At first it seems plausible that admissible algorithms such as A* fail because the score of incoming speech $h^*(n)$, is hard to estimate. However Pearl (1984) has shown that there is a crucial relationship between the heuristic estimate $h^*(n)$, and the costs on the paths of a binary tree search space and that, when most of the branches have a cost associated with them, exponential search is inevitable. I have shown that these conditions can apply in speech processing.

Is it possible to arrange the search space such that these conditions do not arise? We shall now look at ways of reducing the potential combinatorial explosion of breadth-first search.

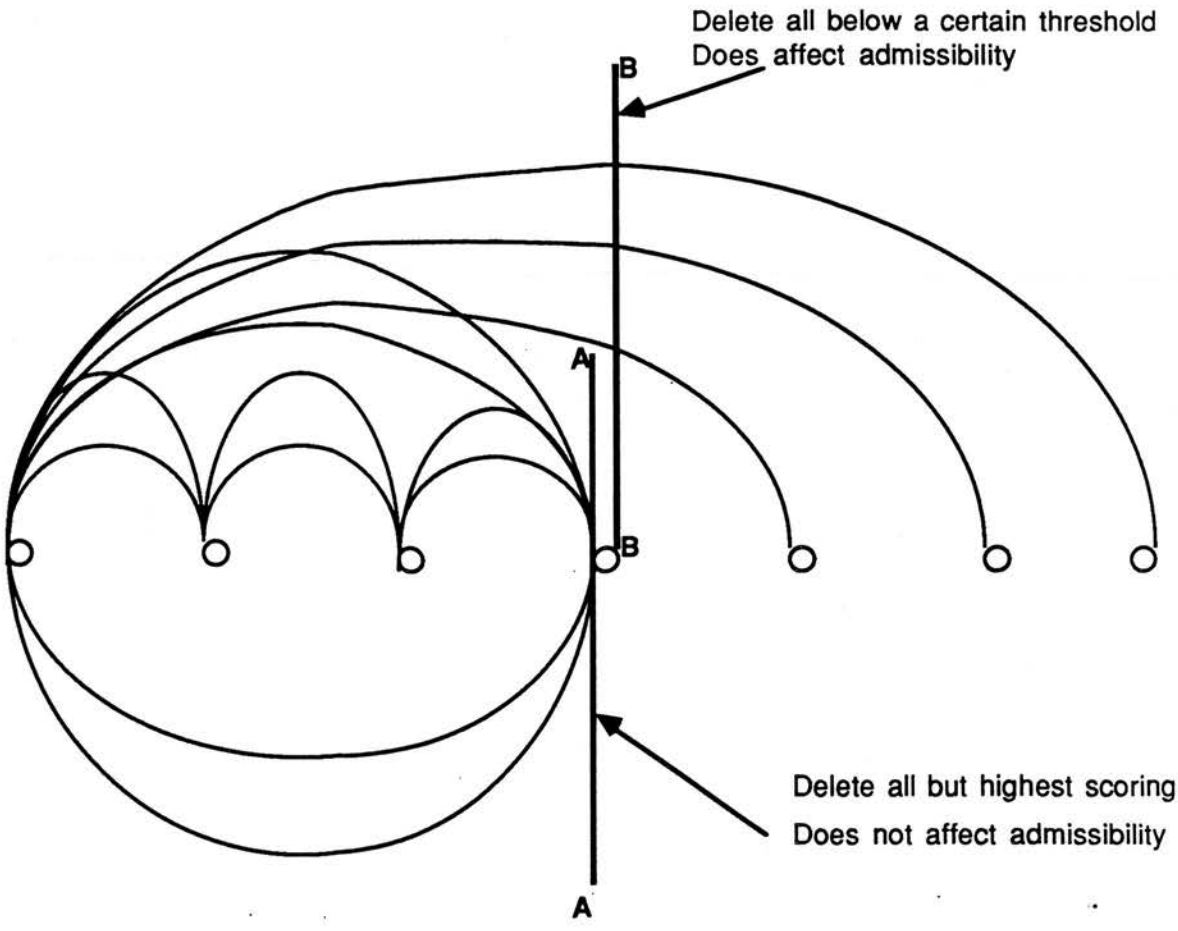

Figure 9.2.

Points at which the graph may be pruned

9.2. Breadth-First Search in Speech Processing

We will now discuss the breadth-first search algorithms used in HARPY, SPHINX and RM1. We are interested in the conditions under which each system is able to reduce the search space, and the effect of these reductions on their performance.

All three systems use search algorithms based on a dynamic programming algorithm known as the Viterbi algorithm (Viterbi 1967). The algorithm is breadth-first; each state at time *t* is advanced before updating states at *t+1*.

9.2.1. HARPY: a constrained system

It is well known that HARPY avoided the combinatorics of breadth-first search by an advantageous structuring of the search space. HARPY's use of a finite-state grammar permitted backwards pruning since a finite-state grammar has the property that any string of words leading to a given state is equivalent to any other string leading to that state as far as future states are concerned. Only the highest scoring of such strings need be remembered.

In addition, the combinatorics of paths between such decision points was controlled in HARPY by using a grammar with a low average branching ratio. The average number of choices at each point was only 10.

However, even with such limitations the search space was too large to search exhaustively. The algorithm was modified to allow forward pruning. This modified algorithm is known as *beam search*. As we have seen an algorithm which *prunes* at point B (Fig. 9. 2) can produce very similar behaviour to one which *orders* at these points. Why was such forward pruning successful in HARPY and not in HWIM? The reason is that the recognition network was arranged in such a way that acoustic similarities were minimized. Woods writes:

> "For example, *What are their affiliations* is in the grammar, but no other sentences starting with *What are their* are possible. The only two sentences starting with *What are the* are *What are the titles of the recent ARPA surnotes* and *What are the key phrases*. These three sentences will almost certainly find some robust difference beyond the initial three words that will reliably tell them apart." (Woods 1982 p314)

So the HARPY search space fulfilled the conditions for successful pruning. Firstly, most of the competing hypotheses scored sufficiently less than the correct hypothesis and

so could be eliminated by forwards pruning from the search on acoustic grounds (see Fig. 9.2, point B) , thus preventing the growth of an increasingly large band of hypotheses.

Secondly, even when some paths did have to be pursued in parallel because their scores were above the threshold, the finite-state arrangement of hypotheses restricted the candidates for any particular slot, and cut short the combinatorics through backwards pruning (Fig. 9.2, point A).

9.2.2. SPHINX and RM1: more general systems

It was thought that the kind of strong reliance on grammatical structuring used in HARPY was necessary to compensate for the generally poor level of acoustic-phonetic processing achieved during the DARPA project (see Klatt 1977). Therefore the designers of more recent systems hoped that an improvement in bottom-up recognition would permit them to achieve good recognition rates even with less advantageously arranged grammars.

The SPHINX system (Lee 1988) and the RM1 syntactic component use similar filtering methods at the syntactic level. Both systems use transition probabilities between syntactically tagged form class pairs to prune the search. Unlike HARPY and HWIM which used only acoustic scores, SPHINX and RM1 combine the a priori scores with the conditional acoustic probabilities. The average branching ratios of these grammars were reasonably high, compared with HARPY[1] .

It must be noted that the syntactic filter would first *add* complexity to the search space by providing alternative syntactic tags for certain lexical items. *Order*, for example can function as a noun or as a verb and the transition probabilities of both senses to following

[1]SPHINX's word pair grammar had a perplexity of 60. Its bi-gram grammar had a perplexity of 20. HARPY's perplexity is 4.5 Lee defines test-set perplexity as the geometric mean of probabilities at each decision point for the test set sentences. (Lee 1988).

words would have to be computed. However, because both systems used information just about adjacent words, they were able to make backwards pruning decisions after every word pair. (i.e. they chose between *so* and *sew* after *so/sew a.*) Since partial interpretations did not have to be kept distinct over stretches longer than two words the combinatorial explosion of word string hypotheses should have been considerably reduced.

However Fig. 9.2 reminds us of the conditions under which such pruning would not be successful. Firstly, backwards pruning at point A is only possible without loss of accuracy if the correct hypothesis is the highest scoring. Secondly, word hypotheses might overlap and leapfrog each other's backwards pruning decision points. This was the case with SPHINX and so the designers implemented beam search using forwards pruning. However, this kind of pruning is only feasible if few of the word hypotheses have similarly high probabilities.

The SPHINX system could prune large numbers of competing hypotheses without loss of accuracy because the competitors were sufficiently differentiated by score. This was due in part to the goodness of the acoustic model, but also to the smallness of the lexicon and to the restrictions of the grammar (in comparison with RM1). The SPHINX system was designed round a specific resource management task. It uses a 997 word lexicon, and a bi-gram grammar extracted from 900 test sentence templates. These templates model mainly questions and commands relating to a database system. When the syntactic component was switched off, *the error rate rose from 4% to 30%*. On average one in three words of the utterance was incorrect.

In contrast, the RM1 system uses a 4,000 word lexicon and a bi-gram grammar, whose probabilities were extracted from a large corpus of general business and government documents. We shall now see how this larger system failed to prune successfully.

The RM1 syntactic component was run over mid-class and fine-class transcriptions and achieved 65.6 and 97.2 recognition rates respectively. These results (Bard et al 1987) appear to be very good, but when one looks at the ranking of the output utterances from which the recognition rates were computed, 30 out of the 79 best-matching utterances were not output first, although they ranked equal first in score.

The table for the fine-class E Set is reproduced below (Table 9.1.) to show the extent of the problem. The lowest best match was at position 27 of equally ranked hypotheses. In the mid-class runs, 43 best-matches out of 73 utterances were not output first ; 16 of these were at or below position 10. Even when the correct word was given a high probability, there were many other words with an equally high probability.

Even with fine-class input the syntactic component was unable to discriminate between the intended utterance and various homophonic utterances produced by lack of word boundary information. (It is possible that the SPHINX system also has some of these problems but is unaware of it. Markov processes typically retain pointers only to a single best path. The path that is output might have been any one of many equal interpretations.)

These results show that the problem of overlapping lexical items is pervasive even with fine-class data and a fairly small lexicon. The bi-word filter eliminated some possibilities but was not good enough to find a single interpretation. Although the syntactic filter could no doubt be improved, it probably could not compensate for a 20,000 word lexicon and/or poorer acoustic-phonetic discrimination.

It is important to bear in mind that HWIM and HARPY used only acoustic probability scores and so were particularly susceptible to the effects of homophonic word strings. SPHINX and RM1, which used syntactic a priori probability scores in addition to conditional acoustic probability scores, could, in principle, tease apart the acoustically similar strings. SPHINX was much more likely to successful in this than RM1, since SPHINX computed its scores from a specific, limited, task domain. RM1's use of a

general corpus could even cause the correct words in a particular utterance to be depressed below the scores of competing homophonic phrases.

9.3. Conclusions

In this chapter we have applied graph search theory to the performance of four speech systems. One of the aims of this thesis was to use graph search terminology to clarify certain speech processing problems and the method has proved itself in a number of ways.

Firstly, we were able to pinpoint a particular problem that arose only in the context of a complete system. The extent of the overlapping homophones problem was not apparent from reports of earlier systems, which assumed the search problems were caused almost entirely by poor front-end processing. Nor could these problems be predicted from studies of the inherent confusability of the lexicon which concentrated on isolated words.

Secondly, the analysis of the search space showed precisely why these hypotheses fell through the net of syntactic constraints, and what effect this had on performance. Our analysis of the search space showed how a potentially depth-first search strategy could be diverted into breadth-first, even given good bottom-up processing. We drew on this analysis to explain the performances of HWIM and HARPY.

We then discussed the conditions under which breadth-first strategies with forward and backward pruning techniques would be successful. Although bi-gram grammars allow pruning after every two words, overlapping, acoustically similar word strings may escape both backward and forward pruning efforts. The forward and backwards pruning points define the limits of left and right context for a system. They are the points at which further information will have little or no effect on the ranking of the current set of hypotheses. SPHINX did not need more information than that provided over word pairs to distinguish

the correct hypothesis. A larger, more general system would need considerably longer stretches of the utterance, and/or considerably more complex top-down information.

The analysis of the search space carried out in this chapter should help to focus attention on the discriminating requirement of top-down information in terms of the number and similarity of hypotheses competing over a stretch of the utterance, and of the distance between pruning points (i.e. the grammar "chunks"'), the two factors which determine the potential combinatorial explosion of hypotheses.

Woods writes of the ARPA project:

> "Although it seemed natural to expect that some word match scores should be good enough that they could be considered correct, thereby eliminating attempts to find alternatives to them, in fact all attempts to implement such an intuition seemed to have led to at best indifferent results and usually to positive degradation. In retrospect, the fact that perfect matches of other words or short word sequences can occur by accident in completely accurate transcriptions of sentences (e.g. 'four' within 'California') should suggest that there is no magic threshold above which one can consider a given hypothesis correct without verifying its consistent extension to a complete spanning theory. It seems, therefore that the absolute value of the local quality score is not what matters in deciding the most likely interpretation. The relative scores of competing hypotheses are more relevant, but what really counts is the eventual quality of the complete spanning theory." (Woods 1982 p. 321)

We have shown that these word matches occur so frequently that competing hypotheses *cannot* be pursued and compared over the length of the entire utterance. Sophisticated higher-level procedures must prune these perfect, accidental word matches before they combine exponentially into a huge space of alternative partial interpretations.

Sentence	Position	Correct	Percentage
E1	**3**	12/12	100.0
E2	**1**	13/13	100.0
E3	**1**	11/11	100.0
E4	**1**	16/16	100.0
E5	**19**	14/14	100.0
E6	**2**	10/10	100.0
E7	**4**	14/15	93.3
E8	**15**	17/17	100.0
E9	**27**	9/10	90.0
E10	**13**	18/22	81.8
E11	**1**	7/7	100.0
E12	**9**	12/12	100.0
E13	**1**	6/6	100.0
E14	**1**	17/17	100.0
E15	**1**	8/8	100.0
E16	**14**	23/23	100.0

Table 9.1

Results of running a syntactic filter over the word graph produced from the fine-class E set.

Chapter 10. Conclusions

10.1. Summary of Research Aims

One of the the most obvious difficulties encountered during the ARPA project was the problem of specifying and controlling the interactions between the contributing knowledge sources. The most successful system, HARPY, owed much of its superior performance to the fact that the interactions were pre-compiled, "fixed" into the knowledge network. HEARSAY-II, using the same grammar and vocabulary, performed less well. This difference in performance was caused mainly by the explosion of partial interpretations on the blackboard, and the difficulty of focusing attention dynamically on the best hypothesis. The control problem was greater for larger or more general systems. HWIM, for example, performed less well than HEARSAY-II, partly because it used a far more general grammar.

In addition to the explosion of *valid*, partial solutions caused by local ambiguity, there was also the problem of *error* caused by poor acoustic or linguistic knowledge. It was not obvious where to lay the blame for an error. Was poor acoustic-phonetic processing always to blame, or were the acoustic cues sometimes simply missing from the signal?

The primary aim of this thesis was to develop a methodology that highlighted the *dynamic* aspects of the speech processing task. We needed ways of representing and analysing the interactions between the knowledge sources. This would help us to explore ways of reducing the combinatorial explosion of valid partial hypotheses.

In addition , since the knowledge sources not only co-operate in finding the correct solution but also co-operate in finding incorrect solutions, we wished to study the genesis and development of particular processing errors. We wished to explore the ways in which left and right context from different knowledge sources could interact to reduce the search space without excluding the correct interpretation.

In the ARPA project, a great deal of weight had been given to the contribution of higher-level knowledge sources. All the systems used lexicons of only around 1,000 words and domain-specific vocabularies and grammars. It was felt that such constraints were necessary to compensate for the poor understanding of bottom-up speech processing.

However, in the fifteen or so years since the project, there had been considerable advances in speech science and phonetics. It was thought that these improvements could support much more ambitious speech recognition machines.

We decided to focus on the interactions of various knowledge sources during the bottom-up processing of words. We used both real speech input from an acoustic-front end that looked for broad-, mid- and fine-class phonemes, and simulated data which assumed the successful recognition of these categories. We used a lexicon which was four times the size of the ones used in the ARPA project.

10.2. Results and Main Contributions

Research on the content of large lexicons encouraged the belief that an improvement to the broad-class level could be sufficient to reduce the cohort of word candidates to a manageable size. Zue (1985, 1986) and Nusbaum & Pisoni (1986) showed that the structural properties of words in the lexicon could constrain considerably under-specified phonemic input. They also claimed that these results could be extended to continuous speech.

Zue (1986) writes:

> While the discussion leading to this model has focused on isolated words, the model can, in principle, deal with continuous speech as well. Instead of working with a set of word candidates, the verifier would deal with a *lattice* of word candidates. Provisions would then be made to determine and compare the relative goodness of words and word strings, subject to phonological, syntactic , and semantic constraints."

We found that:

(i) *even an accurate transcription into mid-classes or a mixture of broad- and fine-class phonemic categories led to a combinatorial explosion of overlapping word strings that spanned the entire utterance.* If the transcription contained errors, the search space would be even harder to manage.

(ii) *any admissible algorithm would tend to search this space breadth-first,* unless very sophisticated higher-level knowledge was available to prune or order the hypotheses.

The main practical contributions of this thesis are:

(i) The implementation of a specific model of Lexical Access within a flexible, graph-based architecture.

(ii) Experimental results, using a large lexicon and different input conditions, which have important consequences for any speech processing mechanism.

The main theoretical contributions are:

(i) The application of graph theory to the analysis of several very different speech processing systems.

(ii) The analysis of the major search algorithms used in speech processing in the light of the experimental results, and a discussion of the conditions under which each algorithm would fail.

We analysed the search space and discovered how overlapping words caused by ambiguous word boundaries contributed to the pruning problem. The analysis made explicit the ways in which bottom-up and top-down information, and left and right context effects are linked, and showed how such information can be used to limit the search space.

10.3. Further Work

Lexical access models can incorporate a huge number of variables (e.g. the size of the lexicon, its content, the specificity and accuracy of the input, the length of the utterances, the number and complexity of the grammatical rules, etc). For the experiments described in this thesis, we made a number of simplifying assumptions. We chose not to vary the lexicon; a 4,000 word lexicon with multiple pronunciations was used throughout. We devised simple but plausible (according to current linguistic opinion) input descriptions and varied these. We did not use syntactic or semantic information, but rather discussed in the abstract the possible constraining influences of such knowledge sources.

Thus, there are many further areas of work suggested by varying these parameters. A few are listed below.

1) We could use the fairly large body of data from the experimental tests to evaluate different syntactic and semantic schemes. In particular it would be interesting to compare different methods of parsing word graphs produced from mixed or fine class inputs.

2) We could vary the lexicon in several ways. Both TRACE and SPHINX use one pronunciation per word together with fuzzy matching to capture many phonological variations. We could see whether such a representation would reduce the number of mismatched words without eliminating the correct word.

3) We could vary the processing e.g. by implementing some of the characteristics of the TRACE system on the larger scale permitted by the Chart. Some researchers in connectionism are becoming interested in graph-based solutions to the stable-state problem. Shastri, at a recent seminar, discussed one pass, acyclic, directed, graph search as a more promising solution than e.g. the computationally expensive alternative of simulated annealing. We could connect the competing and confirmatory edges in the word graphs using inhibitory and excitatory links, and then test different distributed search methods. It is interesting that the graph-based methods which have proved so useful here also appear to have an application in the parallel distributed processing paradigm.

10.4. Final Comments

Some of the results presented here may appear to be negative, but there are also many positive aspects to the work. Firstly, we have demonstrated the usefulness of the Chart architecture in exploring reasonably large and general speech processing problems. Most importantly, however, we have shown how important it is to discover the potential limits of an area of research. While acoustic-phonetic and other lower-level areas of research present

a great many challenges, the higher-levels of the system should not be neglected. We have shown that, even with a substantial improvement in front-end processing, general speech recognition will fail without the immediate application of syntactic, semantic and prosodic constraints.

Appendix 1. Phonemic Symbols

Phonemes

Vowels

/i/	bid
/ii/	bead
/e/	bed
/a/	bad
/aa/	bard
/uh/	bud
/@@/	bird
/@/	the
/o/	pot
/oo/	port
/u/	put
/uu/	boot

Dipthongs

/ei/	day
/ou/	go
/au/	cow
/ai/	eye
/oi/	buy
/i@/	beer
/e@/	bare
/u@/	tour

Consonants

/p/	pea
/b/	bee
/t/	tea
/d/	dye
/k/	key
/g/	guy
/m/	me
/n/	name
/ng/	sing
/f/	fan
/v/	van
/th/	thin
/dh/	then
/s/	sea
/z/	zoo
/sh/	she
/zh/	beige
/ch/	chew
/jh/	judge
/h/	hat
/w/	way
/y;/	yes
/l/	lay/
/r/	ray

	Mid-class	**Phoneme members**
P	voiceless stop	/p, /t/, /k/
B	voiced stop	/b/, /d/, /g/
S	voiceless sibilant fricative	/s/, /sh/, /ch/
Z	voiced sibilant fricative	/z/, /zh/, /jh/
F	voiceless non-sibilant fricative	/f, /th/, /h/
V	voiced non-sibilant fricative	/v/, /dh/
N	nasal	/m/, /n/, /ng/
L	liquid	/l/, /r/
G	glide	/y/, /w/
D	dipthong	/ai/, /ei/, /oi/, /au/, /ou/, /i@/, /e@/, /u@/
FV	front vowel	/ii/, /e/, /a/
BV	back vowel	/aa/, /o/, /oo/, /u/, /uu/
CV	central vowel	/i/, /@@/, /@/, /uh/

	Broad-class	**Phoneme members**
P	voiceless stop	/p, /t/, /k/
B	voiced stop	/b/, /d/, /g/
NSF	non-sibilant fricative	/f/, /v/, /th/, /dh/
N	nasal	/m/, /n/, /ng/

Appendix 2. Test Sentences

SET A. Phonetically Dense Sentences

1. Our lawyer will allow your rule.
2. I'm naming one man among many.
3. I'm well known among men.
4. When will our yellow lion roar?
5. Bobby did a good deed.
6. Did George do a good job?
7. Patty cut up a potato cake.
8. Katie tacked up a cute picture.
9. Which tea party did judge Baker go to?
10. They use our azure vials.
11. Three chefs face a thief.
12. His vicious father has seizures.
13. Weave me a web above a poppy.
14. The judge's short decision really touched the youth
15. A thick-set officer pitched out her hash.
16. Does John believe you were measuring the gun?

SET B Mid-Class Unique Sentences

1. Using natural speech causes problems that differ from those that occur with careful speech.
2. If everyone helped, this wouldn't require tremendous effort.
3. Such industries seldom survive without government aid.
4. Yesterday English professional football attracted splendid newspaper reports.
5. Especially complicated programs often involved several tests.
6. Perhaps additional laboratory research wasn't altogether necessary.
7. Monkeys enjoy apples, oranges, bananas, etc.
8. Staff with serious difficulties sometimes leave rather suddenly.
9. Scientists couldn't identify that particular skeleton correctly.
10. Generally speaking, adult education classes receive huge public support.
11. Nobody noticed its wonderful flowers.
12. Various international market figures indicate considerable growth.
13. Examine that patient quickly.
14. Throughout Africa farmers face terrible conditions with few resources.
15. Shorter sentences appear helpful.
16. Under that system, studying complex subjects presents little trouble.

SET C. Golden Passage Sentences

1. Those species which make trees are trained to a central leader.
2. There are many kinds.
3. Remove all the branches growing between these as well as any growing in other directions.
4. Spread the remaining branches out and tie them to horizontal training wires.
5. Their forms are also grown against walls.
6. Support the main stem.
7. If the main stem is not long enough, train in a leader.
8. Keep to a single leader.
9. Paint on a thick layer.
10. Branches are removed until there is just one left.
11. At the first signs of disease, cut the affected branches back to healthy ripe wood.
12. After a few months, tie in more growth.
13. Cut hard back to this in spring.
14. Sometimes they are grown for their leaves and bark.
15. Burn all wood covered by silver leaf disease as soon as possible.
16. During the following growing season, remove all the larger branches until sufficient length has been gained.

SET D LOB-H Sentences

1. Their experience and knowledge in the wide field of business will be greatly missed.
2. The price ranges for milk, cheese, sugar, bread and flour were very narrow.
3. Matters of particular medical interest are discussed in greater detail in the industrial health report.
4. This will prove a useful source of income to the group.
5. Three systems of local government exist side by side.
6. A brief account of activities in this field during the year is given in the following paragraphs.
7. This does not mean, however, that larger units would not be better if they could be set up,
8. You may wonder what happens to our boys and girls and the answer can best be found in the pages of the old boys' and girls' magazine.
9. This arrangement has proved helpful particularly when the teacher of the class takes both sections.
10. The following paragraphs contain a brief account of the measures.
11. The authority which has to deal with the planning, traffic and road problems of Greater London must exercise a real responsibility.
12. It will also depend on whether workers want to set their own standards.
13. We decided, however, not to take evidence from outside bodies.
14. It is to be expected that both parties will try to express these ideas in behaviour.
15. Training is too rarely continued and developed.
16. I will touch upon that point again in a moment.

SET E Basic Corpus Sentences

1. I shall look forward very much to hearing from you about this.
2. Your name is mentioned in the first paragraph of the deed of trust.
3. I am sorry you were unable to contact me by telephone.
4. Details still need to be filled in, because I was not sure how to do these.
5. I have no doubt that it will be a most interesting and successful occasion.
6. Many thanks for sending me the copy of your letter.
7. We may only hope that the books will turn out to be of more use.
8. Although I like the approach very much myself, I have had no experience of teaching with it.
9. I look forward to your reply as soon as possible.
10. These are not far from each other and from the training point of view they more or less represent a single unit.
11. The order goes in by late November.
12. Students will see it as a relatively fresh source of research material.
13. The next will be around January.
14. I hope you will be able to attend this meeting and that your are now keeping well.
15. The current building will no doubt be discussed.
16. It was, as you may remember, discussed at length a year ago and we hope the interest shown then will still be there.

Utterance	Category	Initial	Mid	Final	Totals
evala05					
	Correct	3	3	2	8
	Mislabelled	1	1	1	3
	Missing	1	1	1	3
	Merged	1	0	0	1
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	6	5	4	15
evala06					
	Correct	2	3	3	8
	Mislabelled	3	1	1	5
	Missing	1	0	1	2
	Merged	0	0	0	0
	Split	0	1	0	1
	Path-Error	1	0	0	1
	Other	0	0	0	0
	Totals	7	5	5	17
evala07					
	Correct	6	6	4	16
	Mislabelled	0	0	2	2
	Missing	0	1	0	1
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	6	7	6	19
evala08					
	Correct	5	7	4	16
	Mislabelled	0	1	1	2
	Missing	1	1	0	2
	Merged	0	1	0	1
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	6	10	5	21

Utterance	Category	Initial	Mid	Final	Totals
evala09					
	Correct	5	5	4	14
	Mislabelled	3	2	4	9
	Missing	0	0	0	0
	Merged	0	0	0	0
	Split	0	1	0	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	8	8	8	24
evala10					
	Correct	2	3	3	8
	Mislabelled	1	1	2	4
	Missing	2	0	1	3
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	5	5	6	16
evala11					
	Correct	5	4	4	13
	Mislabelled	0	1	0	1
	Missing	0	0	0	0
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	1	0	0	1
	Other	0	0	0	0
	Totals	6	5	4	15
evala12					
	Correct	4	5	2	11
	Mislabelled	0	4	2	6
	Missing	1	1	0	2
	Merged	0	0	0	0
	Split	0	1	0	1
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	5	12	4	21

Utterance	Category	Initial	Mid	Final	Totals
evala13					
	Correct	2	7	1	10
	Mislabelled	5	0	1	6
	Missing	0	0	2	2
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	7	7	4	18
evala14					
	Correct	2	6	3	11
	Mislabelled	5	6	5	16
	Missing	1	1	0	2
	Merged	1	0	1	2
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	9	13	9	31
evala15					
	Correct	6	6	3	15
	Mislabelled	1	4	3	8
	Missing	0	0	0	0
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	7	10	6	23
evala16					
	Correct	3	6	2	11
	Mislabelled	2	1	2	5
	Missing	3	2	3	8
	Merged	1	0	0	1
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	9	9	7	25

Utterance	Category	Initial	Mid	Final	Totals
evalb01					
	Correct	7	15	5	27
	Mislabelled	3	4	9	16
	Missing	4	8	4	16
	Merged	0	0	0	0
	Split	0	2	0	2
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	14	30	18	62
evalb02					
	Correct	2	4	9	15
	Mislabelled	4	11	2	17
	Missing	2	4	1	7
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	8	19	12	39
evalb03					
	Correct	4	5	7	16
	Mislabelled	2	7	4	13
	Missing	1	8	0	9
	Merged	0	0	1	1
	Split	1	0	0	1
	Path-Error	0	0	1	1
	Other	0	0	0	0
	Totals	8	20	13	41
evalb04					
	Correct	4	19	5	28
	Mislabelled	2	7	4	13
	Missing	2	9	5	16
	Merged	0	2	0	2
	Split	0	0	1	1
	Path-Error	1	1	0	2
	Other	0	0	0	0
	Totals	9	38	15	62

Utterance	Category	Initial	Mid	Final	Totals
evalb05					
	Correct	5	14	8	27
	Mislabelled	2	4	10	16
	Missing	0	2	3	5
	Merged	0	2	2	4
	Split	0	0	0	0
	Path-Error	1	1	0	2
	Other	0	0	0	0
	Totals	8	23	23	54
evalb06					
	Correct	4	9	8	21
	Mislabelled	3	11	8	22
	Missing	1	3	1	5
	Merged	1	2	0	3
	Split	0	0	0	0
	Path-Error	0	0	1	1
	Other	0	0	0	0
	Totals	9	25	18	52
evalb07					
	Correct	2	7	8	17
	Mislabelled	3	7	4	14
	Missing	1	1	3	5
	Merged	0	3	0	3
	Split	0	0	0	0
	Path-Error	1	0	0	1
	Other	0	0	0	0
	Totals	7	18	15	40
evalb08					
	Correct	4	9	5	18
	Mislabelled	1	11	3	15
	Missing	3	4	2	9
	Merged	0	2	1	3
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	8	26	11	45

Utterance	Category	Initial	Mid	Final	Totals
evalc01					
	Correct	6	11	4	21
	Mislabelled	3	1	7	11
	Missing	2	2	2	6
	Merged	0	0	2	2
	Split	0	1	0	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	11	15	15	41
evalc02					
	Correct	3	2	2	7
	Mislabelled	1	1	2	4
	Missing	0	1	1	2
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	4	4	5	13
evalc03					
	Correct	7	8	13	28
	Mislabelled	4	3	9	16
	Missing	4	9	2	15
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	15	20	24	59
evalc04					
	Correct	5	7	10	22
	Mislabelled	2	4	8	14
	Missing	5	8	3	16
	Merged	0	0	2	2
	Split	0	1	0	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	12	20	23	55

Utterance	Category	Initial	Mid	Final	Totals
evalc05					
	Correct	3	5	4	12
	Mislabelled	3	1	5	9
	Missing	1	1	1	3
	Merged	0	0	1	1
	Split	0	0	0	0
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	7	8	11	26
evalc06					
	Correct	3	4	2	9
	Mislabelled	0	0	2	2
	Missing	1	1	1	3
	Merged	0	0	0	0
	Split	1	0	0	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	5	5	5	15
evalc07					
	Correct	4	6	10	20
	Mislabelled	6	1	2	9
	Missing	2	1	2	5
	Merged	1	0	3	4
	Split	0	0	0	0
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	13	9	17	39
evalc08					
	Correct	4	3	2	9
	Mislabelled	1	1	4	6
	Missing	0	0	1	1
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	1	1	0	2
	Other	0	0	0	0
	Totals	6	5	7	18

Utterance	Category	Initial	Mid	Final	Totals
evald01					
	Correct	3	6	10	19
	Mislabelled	8	7	6	21
	Missing	3	4	6	13
	Merged	1	0	4	5
	Split	1	0	0	1
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	16	18	26	60
evald02					
	Correct	7	8	10	25
	Mislabelled	2	3	8	13
	Missing	4	3	0	7
	Merged	0	0	2	2
	Split	1	0	1	2
	Path-Error	0	0	2	2
	Other	0	0	0	0
	Totals	14	14	23	51
evald03					
	Correct	10	17	16	43
	Mislabelled	4	13	7	24
	Missing	1	3	2	6
	Merged	0	0	0	0
	Split	0	0	0	0
	Path-Error	0	1	2	3
	Other	0	0	0	0
	Totals	15	34	27	76
evald04					
	Correct	3	4	7	14
	Mislabelled	6	4	6	16
	Missing	2	1	0	3
	Merged	1	0	0	1
	Split	0	0	1	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	12	9	14	35

Utterance	Category	Initial	Mid	Final	Totals
eval05					
	Correct	6	5	7	18
	Mislabelled	2	6	8	16
	Missing	1	2	0	3
	Merged	2	0	1	3
	Split	1	0	0	1
	Path-Error	0	1	2	3
	Other	0	0	0	0
	Totals	12	14	18	44
eval06					
	Correct	4	7	9	20
	Mislabelled	11	9	14	34
	Missing	2	4	4	10
	Merged	3	2	2	7
	Split	0	0	0	0
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	20	23	29	72
eval07					
	Correct	5	7	7	19
	Mislabelled	11	5	15	31
	Missing	3	2	3	8
	Merged	8	2	5	15
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	27	16	30	73
eval08					
	Correct	11	4	10	25
	Mislabelled	13	8	14	35
	Missing	3	10	7	20
	Merged	8	6	4	18
	Split	0	0	1	1
	Path-Error	0	0	1	1
	Other	0	0	0	0
	Totals	35	28	37	100

Utterance	Category	Initial	Mid	Final	Totals
evale01					
	Correct	4	6	2	12
	Mislabelled	4	6	6	16
	Missing	4	4	3	11
	Merged	0	2	2	4
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	12	18	13	43
evale02					
	Correct	5	9	11	25
	Mislabelled	6	1	3	10
	Missing	2	2	4	8
	Merged	0	1	1	2
	Split	0	1	0	1
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	13	14	19	46
evale03					
	Correct	6	6	8	20
	Mislabelled	2	3	5	10
	Missing	2	2	2	6
	Merged	0	2	2	4
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	10	13	17	40
evale04					
	Correct	8	8	6	22
	Mislabelled	4	2	6	12
	Missing	4	2	4	10
	Merged	1	0	2	3
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	17	12	18	47

Utterance	Category	Initial	Mid	Final	Totals
evale05					
	Correct	4	10	10	24
	Mislabelled	7	4	5	16
	Missing	3	2	1	6
	Merged	4	0	1	5
	Split	0	0	0	0
	Path-Error	1	0	0	1
	Other	0	0	0	0
	Totals	19	16	17	52
evale06					
	Correct	7	6	8	21
	Mislabelled	2	1	3	6
	Missing	1	1	4	6
	Merged	0	0	1	1
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	10	8	16	34
evale07					
	Correct	6	5	6	17
	Mislabelled	7	3	7	17
	Missing	2	0	2	4
	Merged	1	0	2	3
	Split	0	0	0	0
	Path-Error	0	1	1	2
	Other	0	0	0	0
	Totals	16	9	18	43
evale08					
	Correct	5	9	7	21
	Mislabelled	9	9	15	33
	Missing	3	1	1	5
	Merged	4	3	3	10
	Split	0	1	0	1
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	21	24	26	71

Utterance	Category	Initial	Mid	Final	Totals
evale05					
	Correct	4	10	10	24
	Mislabelled	7	4	5	16
	Missing	3	2	1	6
	Merged	4	0	1	5
	Split	0	0	0	0
	Path-Error	1	0	0	1
	Other	0	0	0	0
	Totals	19	16	17	52
evale06					
	Correct	7	6	8	21
	Mislabelled	2	1	3	6
	Missing	1	1	4	6
	Merged	0	0	1	1
	Split	0	0	0	0
	Path-Error	0	0	0	0
	Other	0	0	0	0
	Totals	10	8	16	34
evale07					
	Correct	6	5	6	17
	Mislabelled	7	3	7	17
	Missing	2	0	2	4
	Merged	1	0	2	3
	Split	0	0	0	0
	Path-Error	0	1	1	2
	Other	0	0	0	0
	Totals	16	9	18	43
evale08					
	Correct	5	9	7	21
	Mislabelled	9	9	15	33
	Missing	3	1	1	5
	Merged	4	3	3	10
	Split	0	1	0	1
	Path-Error	0	1	0	1
	Other	0	0	0	0
	Totals	21	24	26	71

Utterance	Category	Initial	Mid	Final	Totals
TOTALS					
	Correct	216	318	279	813
	Mislabelled	170	186	243	599
	Missing	83	117	87	287
	Merged	38	31	45	114
	Split	5	9	4	18
	Path-Error	8	14	10	32
	Other	0	0	0	0
	Totals	520	675	668	1863

Appendix 4

Description of LA Errors

EVALA01 Our lawyer will allow your rule

Our	Correct and accessed
lawyer	Failed on missing Glide and final Cvowel
will	No segments found
allow	Initial two phonemes not found
your	No segments found
rule	Vowel found, but initial and final Liquids missing.

EVALA02 I'm naming one man among many

I	Mislabelled Dipthong
am	Mislabelled Nasal
naming	Missing initial and mid Nasals
one	Missing initial Glide only
man	Missing final Nasal only
among	Found first CVowel only
many	Vowels found. Nasals mislabelled as B's.

EVALA03 I'm well known among men

I	Mislabelled
am	Correct, but as no match to initial word, no words found
well	All Mislabelled
known	Final Nasal Mislabelled
among	Only first CVowel Correct
men	Final Nasal Mislabelled

EVALA04 When will our yellow lion roar

When	First Glide mislabelled
will	All mislabelled
our	Correct and accessed
yellow	Vowels correct
lion	Last syllable correct
roar	Correct but initial Liquid overlaps final Nasal of lion

EVALA05 Bobby did a good deed

Bobby	Both B's missing. Vowels correct
did	Initial B correct.
a	Mislabelled
good	Correct and accessed
deed	Initial B merged. Final B mislabelled

EVALA06 Did George do a good job

Did	None correct
George	BVowel split. Z's as F and S
do	B as P. Vowel correct

a	Merged with a previous CV labelling
good	Correct and accessed
job	Correct apart from first phoneme

EVALA07 Patty cut up a potato cake

Patty	Correct and accessed
cut	Correct and accessed
up	P Mislabelled as B
a	Correct and accessed
potato	CVowel missing and BVowel mislabelled, otherwise correct
cake	Correct and accessed

EVALA08 Katie tacked up a cute picture

Katie	Correct and accessed
tacked	Correct but final stops merged
up	Correct and accessed
a	Missing
cute	D and BV mislabelled
picture	Both CVowels mislabelled

EVALA09 Which tea party did judge Baker go to

Which	Only final S correct
tea	Correct and accessed
party	Correct, but first vowel split
did	Only initial B correct
judge	Only CVowel correct
Baker	Initial and final phonemes incorrect
go	Correct
to	Vowel incorrect

EVALA10 They use our azure vials

They	V missing, FVowel correct
use	Only D missing
our	L missing
azure	S for Z. Path error on too small initial vowel
vials	Only CVowel found

EVALA11 Three chefs face a thief

Three	Only the L was missing, but because the initial F was split in such a way that it formed an FF path, no words were accessed at all.
chefs	Correct
face	Correct
a	Correct, but not long enough to form a path
thief	Correct

EVALA12 His vicious father has seizures

His	CVowel mislabelled.
vicious	Second syllable correct
father	First syllable correct
has	Vowel approximately in right place. Z labelled as S and merged with following S.
seizures	First syllable correct

EVALA13 Weave me a web above a poppy

Weave	Only FVowel correct
me	Correct and accessed
a	Given as BVowel
web	Only FVowel correct
above	Initial and final phoneme incorrect
a	Given as BVowel
poppy	Correct and accessed

EVALA14 The judge's short decision really touched the youth

The	Fricative missing and CVowel mislabelled
judge's	Z's given as B's. Final Z merged with following S
short	Correct and accessed despite previous errors
decision	Initial B merged with previous P.
really	CVowels given as Fvowels
touched	Correct but not accessed because of previous errors
the	Covered by FVowels
youth	FVowel continues upto the F, which is correct

EVALA15 A thick-set officer pitched out her hash

A	Correct but too far into utterance
thick-set	Only the first CVowel mislabelled
officer	Second and third vowels mislabelled otherwise correct
pitched	Only CVowel incorrect
out	P given as B
her	Correct and accessed
hash	Both fricatives mislabelled

EVALA16 Does John believe you were measuring the gun

Does	Only CVowel correct. Final Z merged with following and labelled P.
John	Only BVowel correct
believe	Vowels correct
you	Vowel correct
were	Correct
measuring	FV and final N correct only
the	Covered only by previous N
gun	Final N given as B otherwise correct

splendid	First syllable correct apart from missing L
newspaper	First syllable covered by FV.
reports	First syllable covered by FV Final P and S correct

EVALB05 Especially complicated programs often involved several tests

Especially	Initial CV given as FV. Remainder correct apart from mislabelled CV and L in 3rd syllable
complicated	Correct up to missing N. Remainder correct apart from missing vowels and L
programs	Inital P merged with hypothesis for previous phoneme causing path error. Missing L's amongst other problems
often	Correct but word beginning not hypothesized
involved	Initial CV given as BV. N correct but also cover V
several	First 2 phonemes correct, the rest covered by N
tests	All correct but gap between FV and S so not accessed

EVALB06 Perhaps additional laboratory research wasn't altogether necessary

Perhaps	All correct apart from F mislabelled as S
additional	First 2 phonemes correct; mislabelled CV then correct S. Last syllable given as BV
laboratory	Missing L and CV.
research	Initial L and CV mislabelled Mid S and CV correct but a gap betweeen them
wasn't	Only N correct
altogether	Missing L, amongst others
necessary	First 2 syllables correct apart from missing CV. Last syllable covered by S

EVALB07 Monkeys enjoy apples, oranges, bananas, etc.

Monkeys	Initial N missing. First syllable covered by BV, but second is correct
enjoy	Initial CV mislabelled. Z covered by previous N
apples	First 2 phonemes correct but large gap filled with Z's
oranges	All mislabelled
bananas	Initial P given as B. Of remaining 6, 4 correct
etc	Initial syllable correct, then mislabelled FV, correct P, last 2 phonemes missing

EVALB08 Staff with serious difficulties sometimes leave rather suddenly

Staff	PS sequence at utterance beginning stopped all access otherwise **staff** would have been correct
with	Missing G and mislabelled F
serious	First 2 phonemes correct. L missing. Both CV's covered by previous FV
difficulties	First 2 phonemes mislabelled
sometimes	Initial S correct but also covers previous Z.. Following CV and N mislabelled
leave	Whole word covered by (correct) FV hypothesis
rather	Missing L and V, but vowels correct
suddenly	Only initial S correct

EVALC01 Those species which make trees are trained to a central leader

Those	Initial V mislabelled
species	Correct apart from final Z
which	Initial 2 phonemes labelled L
make	Initial N mislabelled
trees	Initial P merged with previous stop, L missing, FV appears to be split, Z given as S
are	Correct
trained	Correct apart from mislabelled L
to	BV mislabelled
a	Correct
central	Initial S missing. Only FV and N correct
leader	Initial L missing. FV and B correct, but final CV mislabelled

EVALC02 There are many kinds

There	Initial V mislabelled
are	Correct
many	Correct apart from final FV given as BV
kinds	Only first and last phonemes correct

EVALC03 Remove all the branches growing between these as well as any growing in other directions

Remove	Initial L and CV missing
all	Both phonemes mislabelled
the	Both labelled as BV
branches	B and L missing
growing	Initial B labelled P, L missing, otherwise correct
between	CV and G misssing otherwise correct
these	Initial V missing
as	CV correct, Z given as S
well	All incorrect
as	Correct and accessed
any	Missing N otherwise correct
growing	Initial B correct but most of remainder mislabelled
in	CV missing
other	Missing V only
directions	First 2 syllables correct apart from mislabelled CV and L.

EVALC04 Spread the remaining branches out and tie them to horizontal training wires

Spread	Initial 2 phonemes correct. L and final B missing
the	V missing
remaining	Initial L and CV mislabelled. Remainder correct but FV split
branches	Initial B correct. L missing. Last syllable

	covered by S
out	Correct and accessed
and	Covered by B
tie	Inital P given as B. D correct
them	All incorrect
to	CV given as BV
horizontal	2 BV's and P correct only
training	Missing L between correct P and FV. Last syllable mislabelled
wires	Final S only correct

EVALC05 Their forms are also grown against walls

Their	V mislabelled
forms	F and BV correct but a gap between, N and Z covered by B
are	CV covered by previous B, L mislabelled
also	Correct
grown	Missing L
against	Correct apart from first CV
walls	Mislabelled

EVALC06 Support the main stem

Support	Missing CV between correct S and P. Mislabelled BV before correct P
the	Missing
main	Correct and accessed
stem	Correct apart from last N given as B

EVALC07 If the main stem is not long enough, train a leader

If	Missing initial CV. F correct. P F sequence means no words accessed
the	V labelled as F, merged with previous phoneme
main	correct but FV overlaps with first N causing path error
stem	Correct apart from final N. FV continues on over next word
is	Covered by previous FV
not	Correct but not accessed because of utterance initial error
long	Only BV correct
enough	Initial CV, N covered by FV, rest correct
train	P covered by previous F, and L missing
a	Mislabelled as FV
leader	Mislabelled L and final CV

EVALC08 Keep to a single leader

Keep	Correct, but gap between initial P and FV
to	Correct and accessed
a	Correct but overlaps last BV
single	Mislabelled CV between correct S and N. 2nd syllable mislabelled

leader	Only first FV correct. L mislabelled

girls	B, CV, L covered by B, Z as F
magazine	Previous F continues upto B, which is correct. Rest mislabelled

only	Only N correct
hope	F mislabelled. BV and P correct but separated by gap
that	V and P mislabelled
the	Missing
books	Correct and accessed
will	Mislabelled
turn	Correct but gap between FV and N
out	Correct and accessed
to	P merged with previous P, FV for CV
be	B correct, FV for CV
of	Mislabelled
more	Correct but gap between N and BV
use	Mislabelled apart from S

EVALE08 **Although I like the approach very much myself, I have had no experience of teaching with it**

Although	Mislabelled
I	Mislabelled
like	Mislabelled apart from P
the	V mislabelled, FV correct
approach	Only P correct
very	Only FV correct
much	Correct apart from A
myself	Only S and F correct
I	Covered by previous F
have	Mislabelled
had	Correct but FV split
no	N correct, FV for BV
experience	S,P correct, rest mislabelled
of	Mislabelled
teaching	First 3 phonemes correct, but gap between FV and S. Rest mislabelled
with	Mislabelled
it	P correct

Appendix 5

Paths through Word Lattice

Utterance	FINE-CLASS	MID-CLASS	RM1-DATA
EVALA01		1.971E08	1.814E09
EVALA02	3.413E04	3.471E14	7.680E11
EVALA03	1.092E03	1.666E10	LA FAILURE
EVALA04	1.064E03	8.771E08	9.601E15
EVALA05	4.000E00	1.964E05	1.056E06
EVALA06	2.000E00	5.512E03	1.151E06
EVALA07	3.000E00	1.773E09	5.386E09
EVALA08	3.000E00	1.471E07	4.009E08
EVALA09	4.000E01	1.178E09	1.329E13
EVALA10	8.000E00	5.630E03	1.401E11
EVALA11	2.000E00	5.600E01	LA FAILURE
EVALA12	4.000E00	2.898E04	1.410E09
EVALA13	3.200E01	2.821E07	2.462E12
EVALA14	6.000E00	1.986E07	3.891E07
EVALA15	2.000E01	5.171E08	5.974E10
EVALA16	1.000E00	1.586E10	2.004E10
EVALB01	1.200E01	2.612E13	5.808E04
EVALB02	3.900E01	1.026E12	2.253E05
EVALB03	2.000E00	3.399E10	1.097E15
EVALB04	1.080E02	5.146E16	6.441E13
EVALB05	4.000E00	3.651E12	6.707E18
EVALB06	2.340E02	2.835E19	6.687E18
EVALB07	2.100E01	3.378E15	2.125E05
EVALB08	1.200E01	5.424E09	LA FAILURE
EVALB09	6.000E00	4.735E15	1.183E21
EVALB10	2.400E01	6.010E15	2.640E04
EVALB11	1.000E00	1.593E10	1.139E11
EVALB12	1.980E02	1.003E19	2.525E05
EVALB13	3.000E00	2.180E04	5.426E06
EVALB14	1.080E02	1.597E12	1.661E14
EVALB15	1.800E01	1.882E07	5.196E08
EVALB16	2.400E01	3.267E11	4.017E10
EVALC01	1.440E02	9.962E12	3.608E13
EVALC02	1.900E01	1.203E04	6.722E05
EVALC03	2.700E04	2.749E22	6.794E24
EVALC04	4.082E04	2.547E19	1.871E20
EVALC05	3.000E01	1.350E06	1.257E11
EVALC06	9.000E00	7.712E05	2.093E04
EVALC07	2.376E03	3.393E16	LA FAILURE
EVALC08	2.400E01	4.164E07	8.131E07
EVALC09	6.000E00	5.600E06	8.473E05
EVALC10	4.752E03	3.651E07	2.474E13
EVALC11	4.800E02	2.458E12	9.518E15
EVALC12	1.050E02	1.624E05	1.879E13
EVALC13	5.000E00	8.122E05	1.216E08
EVALC14	1.440E03	1.243E08	1.029E10
EVALC15	1.800E03	7.682E11	7.761E10
EVALC16	3.534E06	1.060E23	3.145E19

EVALD01	1.536E03		4.840E16
EVALD02	6.336E03	1.229E16	3.435E20
EVALD03	8.064E04	2.329E15	7.176E27
EVALD04	8.000E00	9.737E08	3.600E17
EVALD05	3.200E01	5.463E07	LA FAILURE
EVALD06	3.552E05	1.102E22	1.592E25
EVALD07	8.160E02	7.372E17	3.591E09
EVALD08	6.370E07	1.589E32	3.136E03
EVALD09	1.152E03	1.334E15	5.750E19
EVALD10	4.000E01	5.077E16	2.927E11
EVALD11	7.782E04	5.027E31	5.650E16
EVALD12	2.400E02	7.482E18	1.921E25
EVALD13	1.800E01	3.853E12	7.750E15
EVALD14	1.728E03	8.753E13	5.774E05
EVALD15	9.600E02	2.094E13	1.472E14
EVALD16	1.992E03	7.396E14	3.576E12
EVALE01	3.600E01	5.459E12	5.416E08
EVALE02	2.000E02	1.241E14	3.590E15
EVALE03	5.760E02	3.765E16	9.591E13
EVALE04	1.000E02	8.153E13	6.581E18
EVALE05	4.480E02	1.129E17	1.063E06
EVALE06	8.640E02	5.365E14	8.980E15
EVALE07	6.720E02	2.146E15	2.136E19
EVALE08	1.792E03	1.012E19	2.500E13
EVALE09	3.000E03	5.402E10	4.554E07
EVALE10	6.336E04	1.122E28	8.095E19
EVALE11	3.000E01	5.959E09	1.205E06
EVALE12	1.600E02	2.155E14	LA FAILURE
EVALE13	1.800E01	1.396E06	3.107E10
EVALE14	1.659E05	8.449E20	1.043E12
EVALE15	2.400E01	3.326E09	1.128E14
EVALE16	9.600E02	4.807E26	3.983E30
AVERACE PATHS:	8.623E05	2.648E30	5.393E28

Bibliography

Abercrombie, D. 1967
Elements of General Phonetics
Edinburgh University Press, Edinburgh.

Bard, E.G., Blockland, A., Cooper, M., Dalby, J., Duncan, G., Filz, G., Foster, J.C., Hurford, J.R., Johnstone, A., Kupiec, J., Matheson, C., Rohwer, R., Watson, G.S. 1987
An Evaluation of RM1
Centre for Speech Technology Research, Edinburgh University

Bladon, R.A.W., & Al-Bamerni, A. 1976
Coarticulation resistance in English /l/
Journal of Phonetics, 4, 137-150

Bobrow, D.G., Kaplan, R.M., Kay, M., Norman, D.A., Thompson, H.S. & Winograd, T. 1977.
GUS, A Frame-Driven Dialog System.
Artificial Intelligence 8, 155-173.

Carroll, J., Davies, P. & Richman, B. 1971
The American Heritage Word Frequency Book.
Houghton-Mifflin. New York.

Church, K. 1983
Phrase Structure Parsing: a method for taking advantage of allophonic constraints
PhD. MIT

Cohen, P.R. 1985.
Heuristic Reasoning about Uncertainty: An Artificial Intelligence Approach.
Pitman.

Cole, R.A. & Jakimik, J. 1980
A model of speech perception.
In R. Cole (ed) Perception and Production of Fluent Speech. Hillsdale, NJ.

Cole, R.A., Stern, R.M. & Lasry, M.J. 1983
Performing Fine Phonetic Distinctions: Templates vs Features
Paper given to a conference on invariability in speech. MIT.

Dalby, J,. Laver, J. & Hiller, S.M. 1986
Mid-class phoneitc analysis for a continuous speech recognition system.
In Proceedings of the Institute of Acoustics, 8.7, 347-54. Institute of Acoustics: Edinburgh.

Elman, J.L. & McClelland, J.L. 1984.
The Interactive Activation Model of Speech Perception.
In Norman Lass. Speech and Language. New York. Academic Press.

Erman, L.D. & Lesser, U.R. 1980
The Hearsay-II speech understanding system: A tutorial.
In Wayne A. Lea (op cit).

Feldman, J.A. & Ballard, D.H. 1982
Connectionist models and their properties.
Cognitive Science, 6, 205-254.

Forster, K.I. 1976
Accessing the mental lexicon.
In R.J. Wales & E. Walker (eds) New Approaches to language mechanisms. Amsterdam.

Foss, D.J. & Blank, M.A. 1980
Identifying the speech codes.
Cognitive Psychology, 12, 1-31.

Ganong, W.F. 1980
Phonetic categorisation in auditory word perception.
Journal of Experimental Psychology: Human Perception and Performance, 6, 110-125.

Gee, J.P. & Grosjean, F. 1983
Performance structures: a psycholinguistic and linguistic appraisal.
Cognitive Psychology, 15, 411-458.

Goodman, G. & Reddy, R. 1980
Alternative Control Structures for Speech Understanding.
In Wayne A. Lea (op cit).

Gordon, J. & Shortliffe, E.H. 1983.
The Dempster-Schafer theory of evidence.
In Buchanan & Shortliffe 1984. Rule-based expert systems. Addison-Wesley.

Grosjean, F. 1980.
Spoken word recognition processes and the gating paradigm.
Perception and Psychophysics, 24, 267-283.

Grosjean F. 1985
The recognition of a word after its acoustic offset: Evidence and implications.
Working paper. Northeastern University, Boston.

Harrington, J.M., Laver, J., & Cutting, D. 1986
Word-structure reduction rules in automatic, continuous speech recognition.
In Proceedings of the Institute of Acoustics, 8, 451-60. Institute of Acoustics: Edinburgh

Harrington, J.M. & Johnstone, A.M. 1987
The effects of word boundary ambiguity in continuous speech recognition.
Proceedings of the 11th International Conference of Phonetic Sciences, Tallinn, Estonia, USSR.

Harrington, J.M. & Johnstone, A.M. 1987
The effects of equivalence classes on parsing phonemes into words in continuous speech recognition.
Computer Speech and Language 22, 273-288.

Hayes-Roth, F. 1978
The role of partial and best matches in knowledge systems.
In D.P. Waterman & F. Hayes-Roth. Pattern Directed Inference Systems. Academic Press

Hayes-Roth, F & Lesser, V.R. 1977.
Focus of Attention in the Hearsay-II Speech Understanding System.
IJCAI-77, 27-35.

Huttenlocher, D.O. & Zue, V.W. 1984
A model of lexical access based on partial phonetic information.
Proceedings ICASSP, 26.4, 1-26.4.4.

Jarvella, R.J. & Meijers, G. 1983
Recognising morphemes in spoken words: some evidence for a stem-organised mental lexicon.
In Flores D'Arcais & R.J. Jarvella (eds) The Process of Language Understanding. Wiley.

Johnstone A. & Altmann G. 1984.
Automated speech recognition: a framework for research.
D.A.I Research Paper No. 233. Edinburgh.

Johannson, S., Leech, G.N. & Goodluck, H. 1978
The Lancaster-Oslo/Bergen Corpus of British English.
Department of English, Oslo University.

Karttunen, L. 1983.
KIMMO: A General Morphological Processor.
Texas Linguistic Forum 22.

Kay, M. 1977
Morphological & syntactic analysis.
In Zampolli, A. (ed.) Linguisitc Structures Processing
North Holland.

Klatt, D.H. 1977.
Review of the ARPA Speech Understanding Project.
JASA, 62, 1345-1366.

Klatt, D.H. 1979
Speech perception: a model of acoustic-phoneitc analysis and lexical access
Journal of Phoneitc, 7 279-312

Klatt, D. H. 1980
Speech perception: A model of acoustic-phonetic analysis and lexical access.
In R. Cole (op cit).

Klovstad, J.W. 1976.
Probabilistic Lexical Retrieval with Embedded Phonological Word Boundary Rules.
In Woods et al (op cit).

Lamel, L. & Zue, V.W. 1984
Properties of consonant sequences within words and across word boundaries.
Proceedings ICASSP 42.3. 1-42.3.4.

Lea, Wayne, A. 1980
Trends in Speech Recognition.
EngleWood Cliffs: New Jersey.

Lee, K.F. 1988
Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX system.
CMU-CS-88-148

Lieberman, P. 1963.
Some effects of semantic and grammatical context on the production and perception of speech.
Language and Speech, 6, 172.

Lowerre, BT. & Reddy, D.R. 1980
The HARPY Speech Understanding System.
In Wayne A. Lea (op cit).

Luger, G.F. & Stubblefield, W.A. 1989
Artificial intelligence and the design of expert systems.
Benjamin/Cummings CA.

Marr, D. 1982.
Vision.
Freeman. New York.

Marslen-Wilson, W.D. 1986.
Parallel processing in spoken word recognition.
Manuscript.

Marslen-Wilson, W.D. 1980
Speech Understanding as a Psychological Process.
In J.C. Simon (ed) Spoken language generation and understanding. New York Reidel.

Marslen-Wilson, W.D. & Tyler, L.K. 1980.
The Temporal Structure of Spoken Language Understanding.
Cognition, 8, 1-71.

Marslen-Wilson, W.D. & Welsh, A. 1978.
Processing Interactions and Lexical Access during Word Recognition in Continuous Speech.
Cognitive Psychology, 10, 29-63.

McClelland, J.L. & Rumelhardt, D.E. 1981
An interactive activation model of context effects in letter perception; Part I An account of basic findings.
Psychological Review, 88, 375-407.

McClelland, J.L. 1985 *Putting knowledge in its place: A scheme for programming parallel processing structures on the fly.*
Cognitive Science, 9 113-146.

McClelland, J.L. & Elman, J.L. 1986
The TRACE model of speech perception.
Cognitive Psychology, 18, 1-186.

Miller, G., Heise, G. & Lichten W. 1951
The intelligibility of speech as a function of the context of the test materials.
Journal of Experimental Psychology, 41, 329-335.

Miller, G. & Nicely, P.E. 1955
An analysis of perceptual confusions among some English consonants.
Journal of the Acoustical Society of America, 27, 338-352.

Morton, J. 1969
Interaction of information in word recognition.
Psychological Review, 76, 165-178.

Morton, J. & Long, J. 1976
Effect of word transitional probability on phoneme identification.
JVLVB, 15, 43-51.

Mostow, D.J. & Hayes-Roth, F. 1978.
A Production System for Speech Understanding.
In Waterman & Hayes-Roth. Pattern Directed Inference Systems.

Nakatani, L. & Dukes, K. 1977
Locus of segmental cues for word juncture.
Journal of the Acoustical Society of America, 62, 714-719.

Nakatani, L. & Schaffer, J.A. 1978
Hearing "words" without words: prosodic cues for word perception.
Journal of the Acoustical Society of America, 63, 234-245.

Nilsson, N.J. 1980
Principles of Artificial Intelligence.
Tioga Publishing Co., Palo Alto, CA.

Nilsson, N.J. 1971
Problem Solving Methods in A.I.
McGraw-Hill.

Nusbaum, H.C. & Pisoni, D.B. 1986
The Role of structural constraints in auditory word recognition
Proceedings of the Montreal Symposium on Speech Recognition,
McGill University.

Paxton, W.H. 1977
A framework for speech understanding.
Technical note 142. Stanford Research Institute.

Pearl, J. 1984
Heuristics
Addison-Wesley.

Pisoni, D.B., Nusbaum, H.C., Luce, P.A. & Slowiaczek, L. M. 1985
Speech Perception, word recognition and the structure of the lexicon.
Speech Communication, 4, 75-95.

Pisoni, D.B., Luce, P.A. & Nusbaum, H. C. 1986
The role of the lexicon in speech perception.
Draft Paper. Speech Research Laboratory, Indiana University.

Pollack, I. & Pickett, J.M. 1963.
The intelligibility of excerpts from conversation.
Language and Speech, 6, 165-171.

Reddy, D.R. 1976
Speech Recognition by Machine:A review.
Proceedings of the IEEE, 64, 501-531.

Reddy, R.D. 1980
Machine models of speech perception.
In R.Cole (op cit)

Reddy, D.R. & Erman, L.D. 1975.
Tutorial on System Organisation for Speech Understanding.
In D.R. Reddy (ed) Speech Recognition. Academic Press.

Shockey, L. 1973
Phonetic and phonological properties of connected speech.
Working paper in phonetics 13. Ohio State University.

Samuel, A.G. 1981
Phonemic Restoration: Insights from a new methodology.
Journal of Experimental Psychology, 10, 474-494

Samuel, A.G. 1981
The role of bottom-up confirmation in the phonemic restoration illusion.
Journal of Experimental Psychology, HPP 7, 1124-1131

Shillcock, R.C., Altmann, G.T.M., & Bard, E.G. 1987
The role of subsequent context in word recogntion in spontaneous speech
Working paper. CSTR, University of Edinburgh.

Smith, A.R. & Sambur, M.R. 1980
Hypothesizing and Verifying Words for Speech Recognition.
In Wayne A. Lea. Trends in Speech Recognition. Prentice-Hall.

Swinney, D.A. 1979
Lexical Access during sentence comprehension.
JVLVB, 18, 645-659.

Thompson, H.S. 1984
Word Recognition: A paradigm case in computational psycho-linguistics.
Proceedings of the Sixth Annual Meeting of the Cognitive Science Society.

Thompson, H.S. & Ritchie, G.D. 1984
Techniques for Parsing Natural Language: Two Examples.
In M. Eisenstadt & T. O'Shea (eds.) Artificial Intelligence Skills. Harper & Row.

Viterbi, A.J. 1967
Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm.
IEEE Transactions on Information Theory IT-13(2):260-269.

Waltz, D.L. & Pollack, J.B. 1985
Massively parallel parsing.
Cognitive Science 9 51-74

Warren, R.M. & Sherman, G. 1974
Phonemic restorations based on subsequent context.
Perception & Psychophysics, 16, 150-156

Warren, R.M. & Warren, R. P. 1970
Auditory confusions and illusions
Scientific American 233 30-36

Wolf, J. J. & Woods, W.A. 1978
The HWIM speech understanding system.
In Wayne A. Lea (op cit).

Woods, W.A. 1982.
Optimal Search Strategies for Speech Understanding Control.
Artificial Intelligence, 18, 295-326.

Woods, W.A., Bates, M., Bruce, B., Cook, C., Klovstad, J., Nash-Webber, B., Schwartz, R., Wolf, J., Zue, V. 1976.
Speech Understanding Systems - Final Technical Progress Report.
Report No 3438 Vols. I-V, BBN, Camb. Ma.

Zue, V.W. 1986
The role of analysis by synthesis in phonetic recognition
Proceedings of the Montreal Symposium on Speech Recognition,
McGill University.

Zue, V.W. 1985
The Use of speech knowledge in automatic speech recognition
Proc. IEEE 11 1602,1615.