15.

THE HARPY SPEECH UNDERSTANDING SYSTEM

Bruce Lowerre
Raj Reddy
Carnegie-Mellon University
Pittsburgh, PA 15213

15-1. ABSTRACT

Harpy is one of the first systems to demonstrate that high performance, large vocabulary connected speech recognition systems can in fact be realized economically for task-oriented (restricted) languages. In this chapter we present, using simple examples, the principles of organization of the Harpy system. We will illustrate how knowledge sources (KSs) are specified, how the knowledge compiler integrates the KSs into a unified directional graph representation, and how this knowledge is utilized. In conclusion, we will discuss many of the limitations of the present system and how these can be eliminated or reduced in future systems.

15-2. INTRODUCTION

Harpy is one of the systems developed as part of the five year ARPA speech understanding research effort. A study group headed by Allen Newell proposed a set of specific performance goals in 1971 to be achieved within a five year period (Newell, et al., 1971). Figure 15-1 presents the original stated goals and the performance of the Harpy system. It is interesting to note that Harpy not only met all the specifications but exceeded several of the stated objectives. In particular, the system ran an order of magnitude faster with only about half the error rate, in a noisy environment using a poor frequency-response close speaking microphone.

A comprehensive review of the recent speech recognition research including the ARPA speech program is given in Reddy (1976), Klatt (1977), and Lea (this volume). Here, we will briefly mention some of the prior research which directly contributed to the success of the Harpy System.

The parametric representation and the distance metric used in the Harpy system are based on LPC coefficients (Atal, 1971; Itakura, 1968; and Markel, 1972) using minimum distance residual metric (Itakura, 1975). The segmentation and labeling are extensions of techniques used in Hearsay II (Erman, this volume), Hearsay I (Reddy, et al., 1973), and earlier work of our group (Reddy, 1967). Other significant early work in this area is by Tappert et al. (1970) using the transeme approach. The juncture rules were all empirically derived but were influenced by the work of Oshika, et al., (1975) and Cohen & Mercer, (1975). The integrated network representation of knowledge is based on the Dragon system developed by Jim Baker at Carnegie-Mellon University. The best-few beam search technique is an extension of the best-first technique used in the Hearsay I system. Of all these, the single most important intellectual legacy upon which Harpy is based is the representation and delayed-decision techniques first used effectively in the Dragon system (Baker, 1975). It is also important to note the intellectual ferment created

Targets (from 1971)	HARPY Performance (1976)
Accept connected speech	Yes
from many	5 (3 male, 2 female)
cooperative speakers of the General American Dialect	Yes
in a quiet room	Computer terminal room
using a good quality microphone	Ordinary microphone
with slight tuning/speaker	Substantial tuning (20-30 utterances/speaker)
requiring only natural adaptation by the user	No adaptation required
permitting a slightly selected vocabulary of 1000 words	1011 words, no post-selection
with a highly artificial syntax and highly constrained task	Combined syntactic and semantic constraints → Avg. branching factor of 10
providing graceful interaction	modest interaction capabilities
tolerating < 10% semantic error	9% sentence error -- 5% semantic
in a few times real-time* [on a 100 MIPS machine]	80 times real-time on a* .35 MIPS PDP-KA10 Using 256K of 36-bit words With a simple program organization Costing about $5 per sentence
and be demonstrable in 1976 with a moderate chance of success.	Operational 13 August 1976

* A few times real-time on 100 MIPS processor is anywhere from 200 to 500 MIPSS (Millions of Instructions executed Per Second of Speech). The actual performance of Harpy was about 28 MIPSS, i.e., 80 times real-time on a .35 MIPS processor.

Figure 15-1 Performance of the Harpy System

by the large ARPA effort and continued interactions with several other groups actively pursuing similar research objectives.

15-3. KNOWLEDGE SOURCES

An important tenet that distinguishes speech recognition systems from understanding systems is the assumption that the speech signal does not have all the necessary information to uniquely decode the message and that one must use linguistic and context dependent knowledge sources to infer (or deduce) the intent of the message. However, this distinction appears to be getting fuzzy as time progresses and systems use not all the available sources of knowledge, but only a few of the knowledge sources, e.g., phonological, prosodic, lexical, and syntactic knowledge. Harpy belongs to this class of systems. Although we have implemented two tasks (chess and abstract retrieval) for which we have included task and context specific knowledge, most tasks implemented on Harpy do not include this knowledge. However, the structure and implementation of the Harpy system do not preclude the use of such knowledge.

In this section we will illustrate, by means of a simple example, typical knowledge sources used in a task and their specification and modification by the user. Figure 15-2 contains all the knowledge sources for a Mini-Query-Language (MQL). This is the usual form in which a user specifies various knowledge sources. Figure 15-3 illustrates some of these knowledge sources as directed graph representations. If an interactive graphics display terminal is available, knowledge sources can be specified directly in this network form by the user. This is the more natural form of specification as the knowledge compiler (Sec. 15-4) operates on the directed graph representation.

15-3.1 Syntactic Knowledge.

The syntax of the language is given in Fig. 15-2a as a set of four syntax equations. This is a special form for representing a class of phrase structure languages and is often called in computer science literature Backus Normal Form (BNF). Sentences "Please help me." or "Please show us everything." are legal in this language. The first equation states that a sentence in this language consists of a begin symbol ([) followed by phrase <SS> (in the second equation) followed by an ending symbol (]). The second equation states that the symbol <SS> may be defined in terms of two alternative phrase structures: please followed by help followed by any phrase defined by symbol <M> or please followed by show followed by <M> followed by phrase <Q>. Equation 3 states the phrase <Q> can be either everything or something. Equation 4 states that the phrase <M> can be either me or us. Figure 15-3a shows how these equations would be specified in directed graph representation.

Specifying the grammar for a new task is the role of a language design expert. He needs to have a deep understanding of the implications of each of the design decisions. The language must permit as much flexibility and graceful interaction as possible without sacrificing the recognizability and accuracy. Controlling the branching factor and the vocabulary ambiguity are two basic techniques available to the language designer. Goodman (1976) provides a formal model combining both of these aspects. Some of the language design tools developed by Goodman were instrumental in the development of a family of languages for the Harpy system which led to the successful demonstration in 1976. Although time pressures made it difficult to fine-tune these languages, it is clear that constrained and yet habitable languages can

```
<SENT> ::=    [ <SS> ]
<SS> ::=      please help <M>
              please show <M> <Q>
<Q> ::=       everything
              something
<M> ::=       me
              us
```

Figiure 15-2a Grammar of MQL

```
everything    (-,0) (EH,EH2) V R IY2 TH IH3 NX
help          (-,0) HH AA3 EL3 (← (-,0), -) P
me            (-,0) M IY
please        (← (-,0), -) (P (L,L2), PL (L,0)) IY (Z{4}, (Z,0) S)
show          (-,0) SH AA5 (OW,0)
something     (-,0) S AA M TH IH3 NX
us            (-,0) IH6 S (HH,0)
[               -
]               -
```

Figure 15-2b Pronunciation Dictionary for MQL

```
#,-↑-{,100}
-,#↑.-{,100}
-,-↑-{1,40}
0,-↑.< -{1,40}
Z,HH}
[Z,S],SH{
P,M{
OW,IH6↑(.< (>,0), >)
IY,[EH,EH2]↑(.< (>,0), >)
0,<{
```

Figure 15-2c Word Juncture Rules for MQL

-	1	5	HH	3	6	AA	4	10
←	1	8	M	3	6	AA3	6	12
P	3	8	NX	5	12	AA5	3	8
PL	4	10	R	2	5	OW	2	20
V	3	6	L	2	4	IH3	3	7
Z	2	10	L2	2	5	IH6	2	6
TH	3	8	EL3	8	14	IY	4	20
S	4	12	EH	6	14	IY2	4	8
SH	4	12	EH2	2	7			

Figure 15-2d Phone Templates for MQL

Figure 15-2 Knowledge Source for a Mini-Query-Language (MQL)

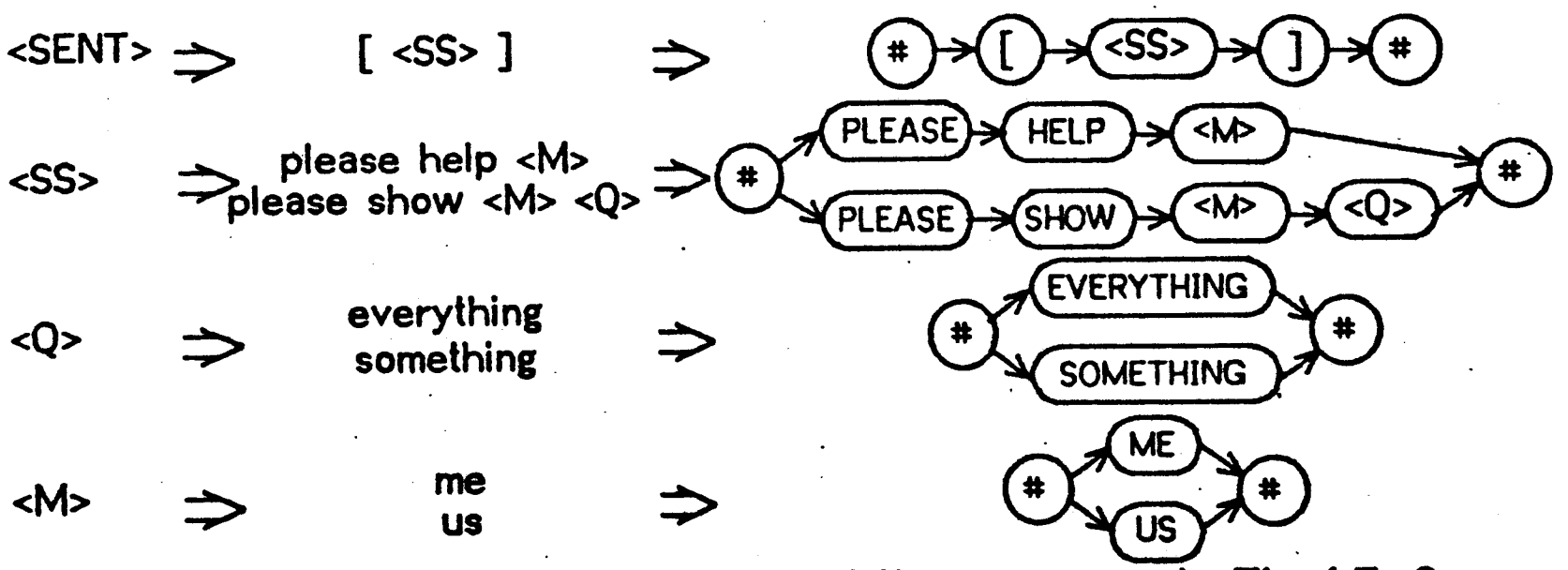


Figure 15-3a Representation of the grammar in Fig 15-2a

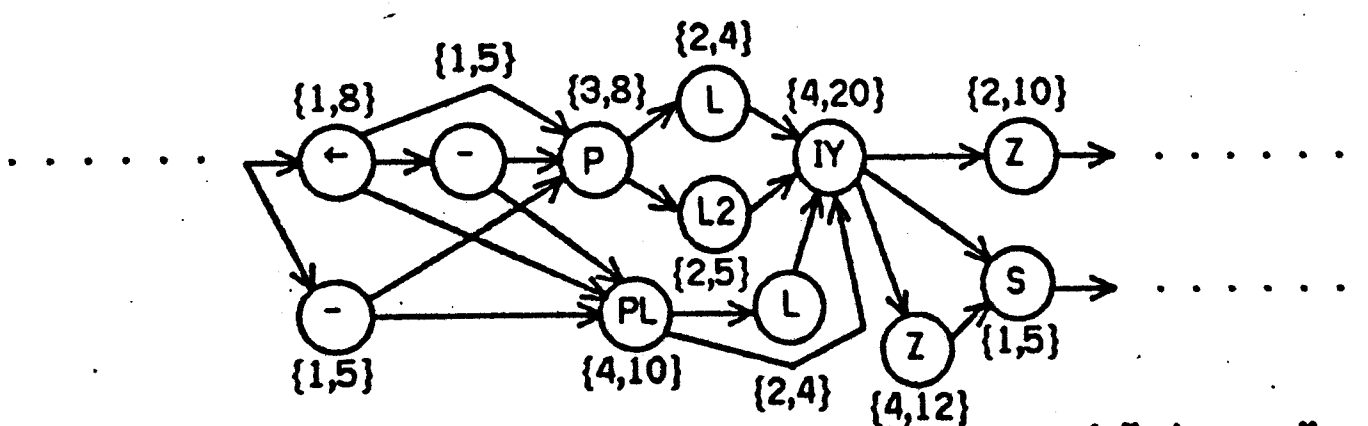


Figure 15-3b Representation of the word "please"

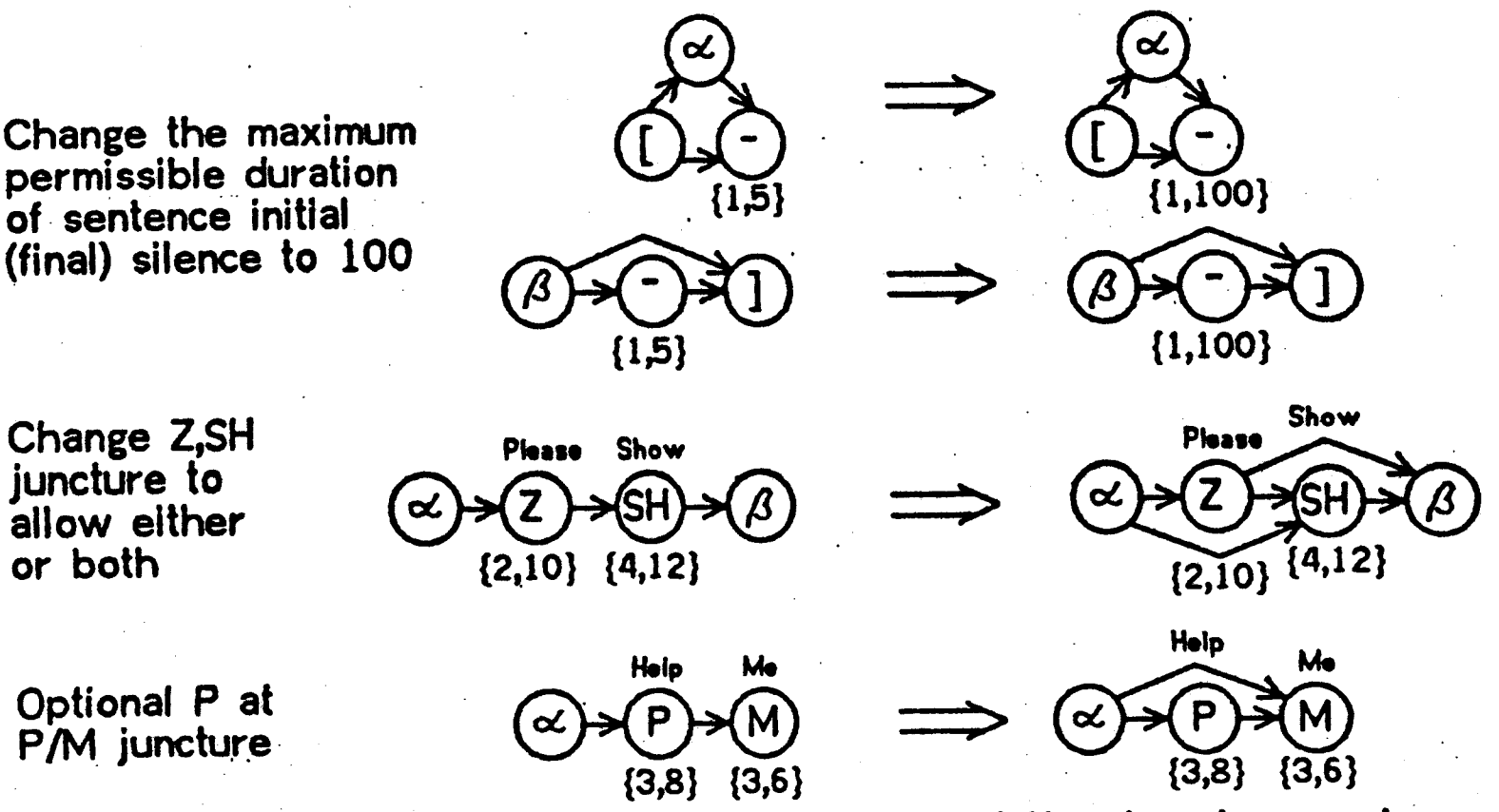


Figure 15-3c Representation of some of the juncture rules

Figure 15-3 Directed graph representation of some of the knowledge given in Figure 15-2

be designed for use in specific task domains.

15-3.2 Lexical Knowledge.

Figure 15-2b gives the pronunciation dictionary for the MQL. Alternative pronunciations of each word are represented in a special notation (Lowerre, 1976, pg. 41). This is a string representation of the pronunciation graph. Figure 15-3b gives the graph representation for the word please. In the string notation, parentheses are used to indicate alternative choices and commas separate the alternatives. "0" is used to indicate null choice. Curly brackets are used to specify duration in centi-seconds, if the expected duration differs from the normal range of durations given as part of phone specification.

Note that the pronunciation graph in Fig. 15-3b is significantly more complex than what would normally appear as a baseform in a pronunciation dictionary. This is because the Harpy dictionary attempts to capture all the intra-word phonological phenomena as part of pronunciation. In the example given in Fig. 15-3b we see several such phenomena: /l/ in please may be partly or wholly devoiced and /z/ may be voiced, devoiced, or mixed. For each of these alternative phones, spectral patterns expected may be different for different allophones and require distinct symbolic notation.

Making up pronunciation dictionaries for a new vocabulary is perhaps the most time consuming task at present. Our initial attempts to derive all the intra-word phenomena from a set of pre-defined phonological rules operating on a baseform proved to be a failure. It appears that numerous subtle and complex processes have to be modeled very carefully before we can derive the pronunciation graph at the level of detail required by the Harpy system. Our present plan is to develop automatic and/or interactive knowledge acquisition techniques for learning the word structure directly from examples without the use of any predefined model.

15-3.3 Juncture Rules.

Another aspect of speech knowledge that is important for the successful operation of Harpy relates to the phenomena that occur at word boundaries. Unlike written text, where word boundaries are clearly defined, in spoken language word boundaries tend to overlap making it difficult to detect the end of one word and the beginning of the next. Knowledge about such phenomena is represented in Harpy in the form of Juncture Rules.

Figure 15-2c gives a set of word juncture rules for MQL. In general juncture rules contain examples of insertion, deletion, and change of phones occuring at word junctures. For this simple task, only a few juncture rules are required. Lowerre (1976, pg. 42) gives the notation used for specifying juncture rules. The exact details are unimportant here. What is interesting to observe is that many of these rules can be rewritten in the form of graph-rewriting rules (as illustrated in Fig. 15-3c) and many of the known phonological phenomena can be represented in this form. Figure 15-3c contains an example of duration change at sentence initial and final positions, an example of feature assimilation, and an example of phone deletion at word boundaries.

The approach taken in the design of Harpy is to manually tailor the juncture rules to the task at hand. This is a time consuming process. In the long run a complete set of rules capturing a wide variety of juncture phenoma has to be collected, revised, and refined. Phonological rules available in the literature do not provide the necessary level of detail to be used directly in Harpy-like systems. This is a case where automatic knowledge acquisition from examples is likely to be very helpful.

15-3.4 Phonemic Knowledge.

Figure 15-2d shows the list of phones used in MQL and the range of durations permissible for each phone, in centi-seconds. The durations are determined empirically. Missing in the figure are the phone templates which are represented as a transformation of linear prediction coefficients.

In general, the phone list for a given task consists of all the allophones (with distinct spectral characteristics) necessary to uniquely represent the expected spectral sequences in the word list. Each distinct allophone of a given phoneme is identified by adding a digit at the end of the phoneme symbol.

The generation of the spectral templates for various allophones is one of the significant innovations within the Harpy system. Lowerre (1977) describes this process in detail. There are three sources of variability that affect the phone template characteristics: environmental noise, transducer characteristics, and speaker characteristics.

In the Harpy system a single composite template is generated automatically for each allophone, capturing all three sources of variability. A speaker is asked to repeat 20 or so predefined sentences in the environment using the microphone. The Harpy system analyzes these sentences using a speaker independent set of templates to identify the location of each word and each phone within the word (more on the recognition process in Sec. 15-5). All instances of a given allophone in the 20 or so training sentences are averaged to generate a speaker specific, microphone specific, and environment specific template. Note that the template generation process is entirely automatic. Since any number of instances of an allophone can be averaged to generate the template, the same technique can be used to generate speaker independent or microphone independent templates by averaging over an appropriate set.

15-4. KNOWLEDGE COMPILER

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. The Hearsay II System (Erman, in this volume) provides one interesting solution to the problem. The so-called "blackboard" model is used to effect communication and cooperation. The Harpy system attempts to structure all knowledge into a unified directed graph representation. Some KSs syntax are easily represented as a graph. But many other aspects: juncture rules, pronunciation variability, and duration effects, are not as easily represented as a graph and often require considerable ingenuity. Much of the Harpy's success is the result of solving the difficult technical problems associated with forcing all the diverse KSs into a unified framework.

As we saw in Sec. 15-3, the KSs are specified by the user as independent units, each with its own notation (Fig. 15-2). Translating KSs represented in different notations into a single unified graph requires using a knowledge compiler. This process is not unlike translating programs specified in a higher level language into machine code using a language compiler. Figure 15-4 gives a flowchart of knowledge compiling process in the Harpy system. This is by far the most computer-intensive process of the Harpy system. It took over 13 hours of DEC System/10 (KL) time to compile the 1011 word vocabulary document retrieval task into a 15,000 state network.

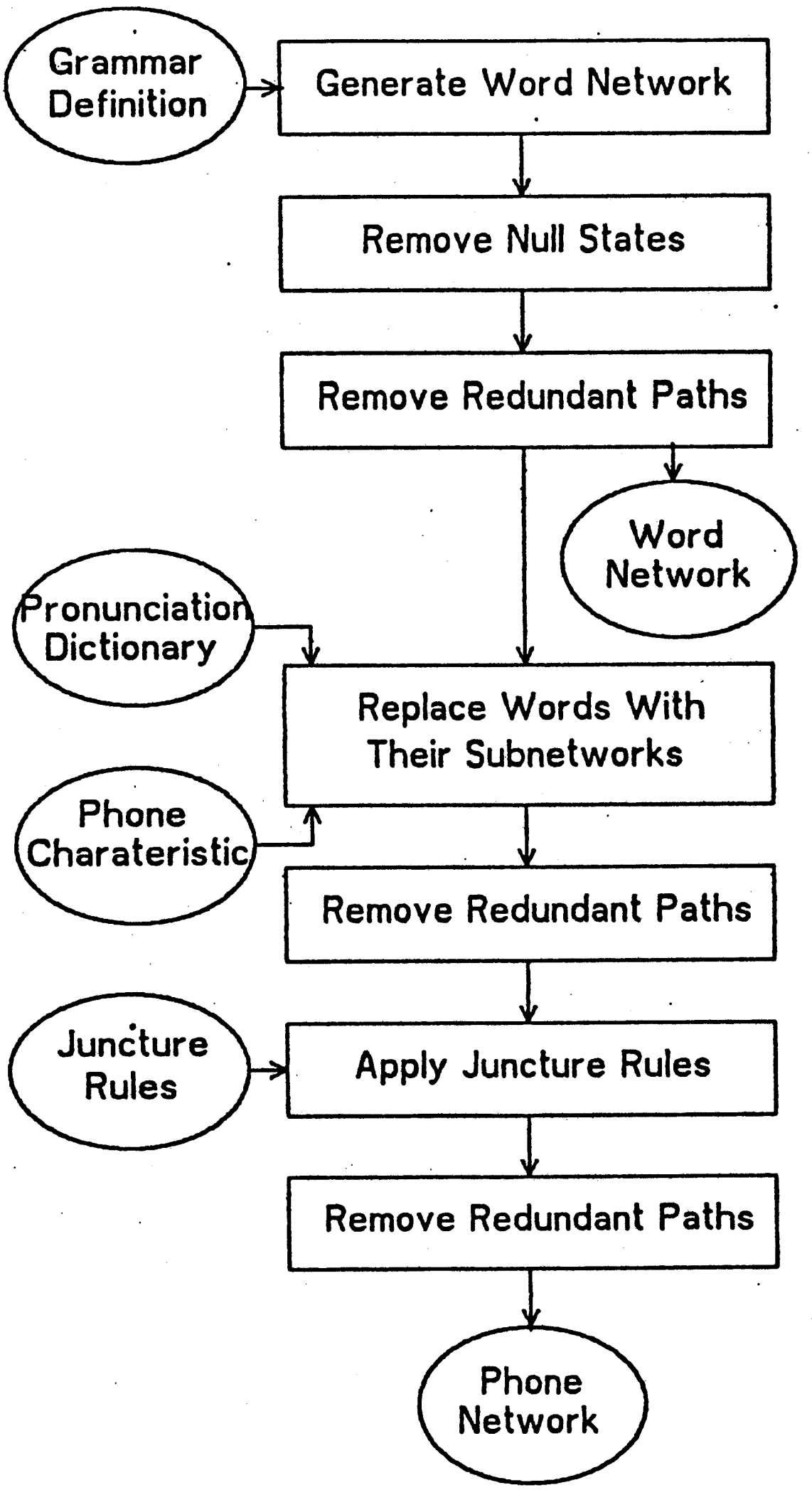


Figure 15-4 Flowchart of the Knowledge Compiler

The first step in the compiling process is to generate a word network from the BNF specification of the type given in Fig. 15-2a. The goal is to replace all occurrences of non-terminal symbols (ones that require further specification; enclosed within <...>) by terminal symbols (words of the language). In the top equation <SENT> is redefined repetitively until all the paths in the directed graph contain only terminal symbols. The Harpy system uses two different techniques to achieve this transformation: substitution and pointer replacement. In the substitution technique every occurrence of a non-terminal is replaced by its definition. In the pointer replacement technique all occurrences of a non-terminal are changed so that they point to the definition of that non-terminal. In cases where each non-terminal occurs only once on the righthand side of the equation, both these techniques yield identical graphs. Otherwise the pointer replacement technique results in a more compact but less constrained grammar than the original grammar. If one wants an exactly equivalent syntax then one uses the substitution rather than the pointer replacement. We do not illustrate it here because of lack of space.

Figure 15-5 illustrates this process along with null state elimination and redundant path removal. We start with the basic definition of <SENT> as shown at the top of Fig. 15-3a. We modify the pointers so that pointers to <SS> are altered to point to the definition of <SS>. This process is repeated for each non-terminal encountered in the equations as seen in Fig. 15-5a. Figure 15-5b shows a topological equivalent graph to Fig. 15-5a resulting from pointer replacement technique.

Two important facts are worth noting in the graph in Fig. 15-5b. Note that it permits some sentences that were not legal in the original grammar, e.g., "please show me" and "please help me everything". Secondly the graph has a large number of null nodes (#) which serve no function at this stage.

Lowerre (1976, pp. 44-51) gives details of techniques for state space reduction by removing null states and redundant states. Figure 15-5c shows the graph after the null states are removed. The null state removal reduces the number of states but usually increases the number of pointers. Figure 15-5d shows the word network after the removing of redundant states. Two states A and B in the graph are redundant if: 1) each has the same terminal lexical symbol and the same set of prior states or following states. Using this definition we find that the two occurrences of the word please in Fig. 15-5c can be replaced by a single state as shown in Fig. 15-5d. The pronunciation network please shown in Fig. 15-3b is a further specification of please appearing in Fig. 15-5d. Thus, we can replace the word please in Fig.15-5d with its pronunciation network. Likewise, replacing every node in the word network in Fig. 15-5d with its pronunciation network generates a new finite state graph, where each path is a pronunciation of an acceptable sentence. Figure 15-6 shows part of the phone network for our MQL example. We remove redundant states in the phone network using the same techniques as in the word network. Applying the graph rewriting juncture rules,the examples given in Fig. 15-3c, adding an optional silence before the final state and again removing redundant states we get part of the final compiled knowledge network for our Mini-Query-Language as shown in Fig. 15-7.

In its final compiled form each state in the network contains the following information: word lexicon number, phone lexicon number, word id number, minimum phone duration, maximum phone duration, transition count, intersection state marker, and a list of its following states. The word lexicon number is calculated from a sequential numbering of the dictionary words. This serves as an internal lexical pointer to the actual word name (string) for recognition. The phone number is likewise calculated from a sequential numbering of the phones and serves as a pointer to the phone name.

The word id is used to uniquely identify every occurrence of a terminal

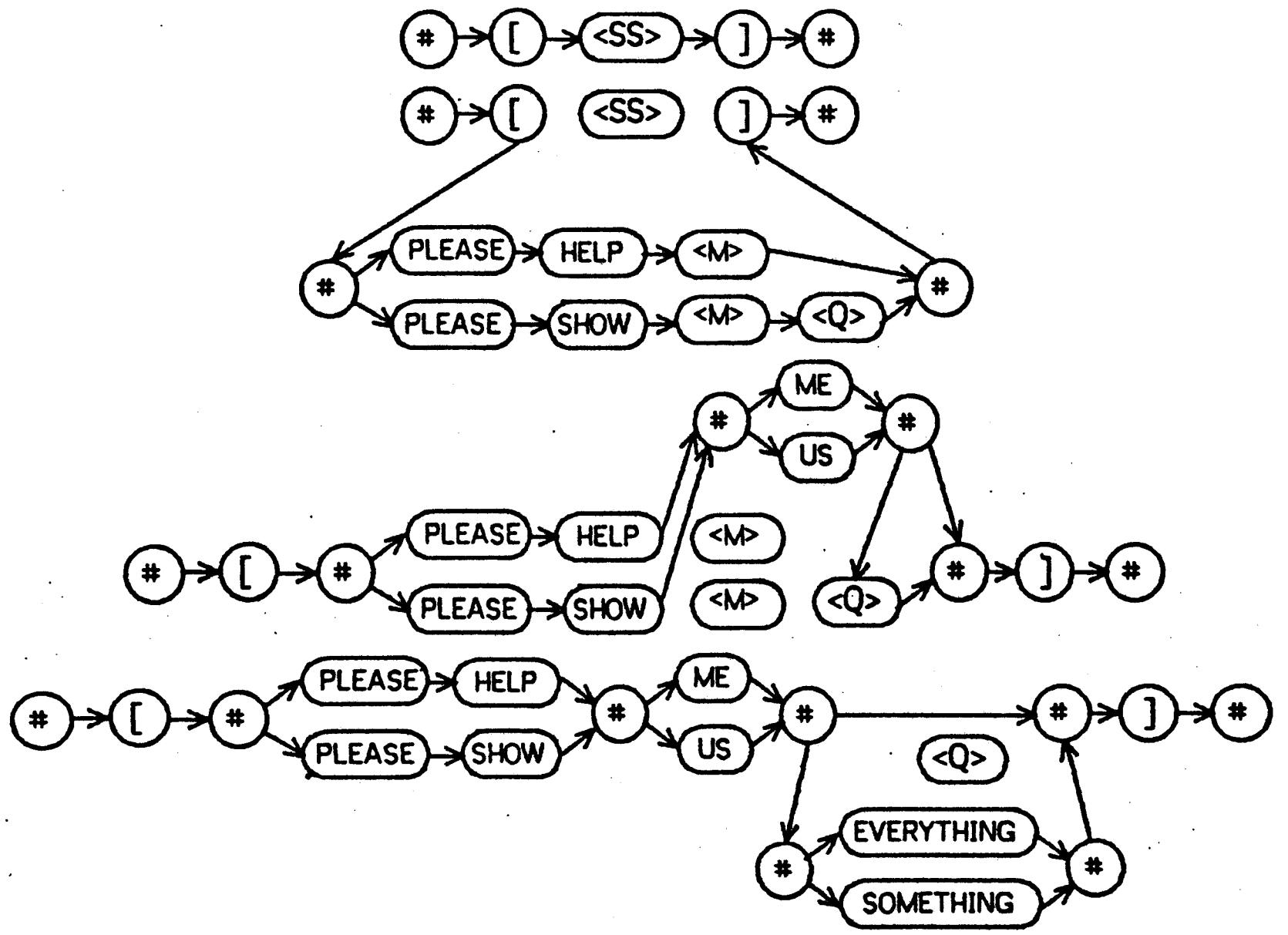


Figure 15-5a Generation of the Word Network From the Grammar Specification

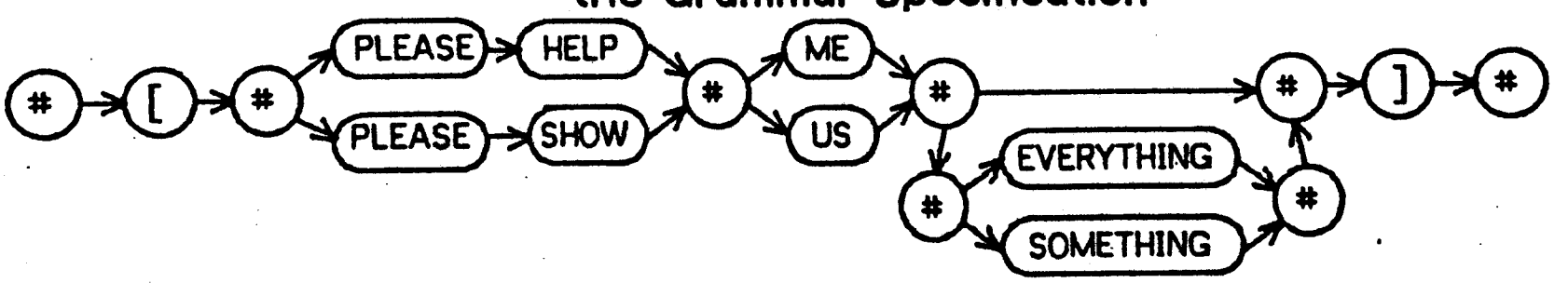


Figure 15-5b Word Network After Expansion

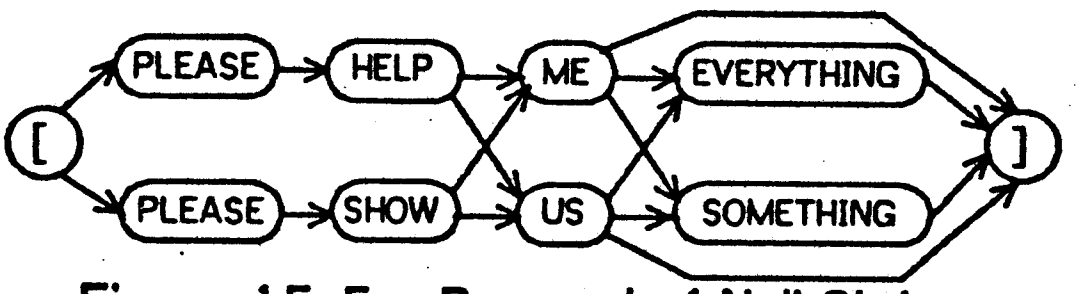


Figure 15-5c Removal of Null States

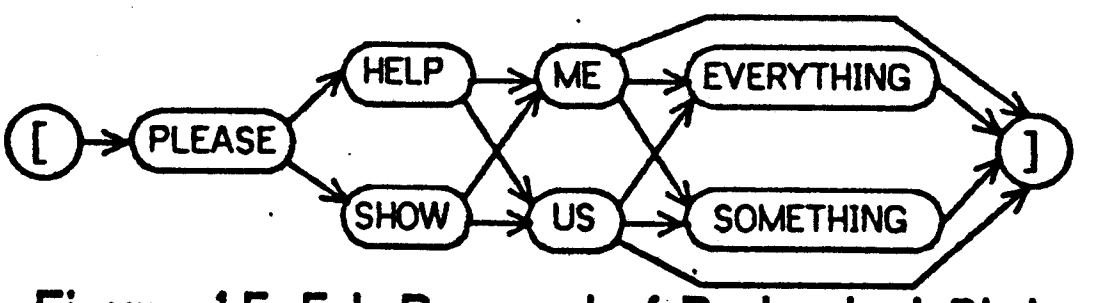


Figure 15-5d Removal of Redundant States

Figure 15-5 Generation of Word Network

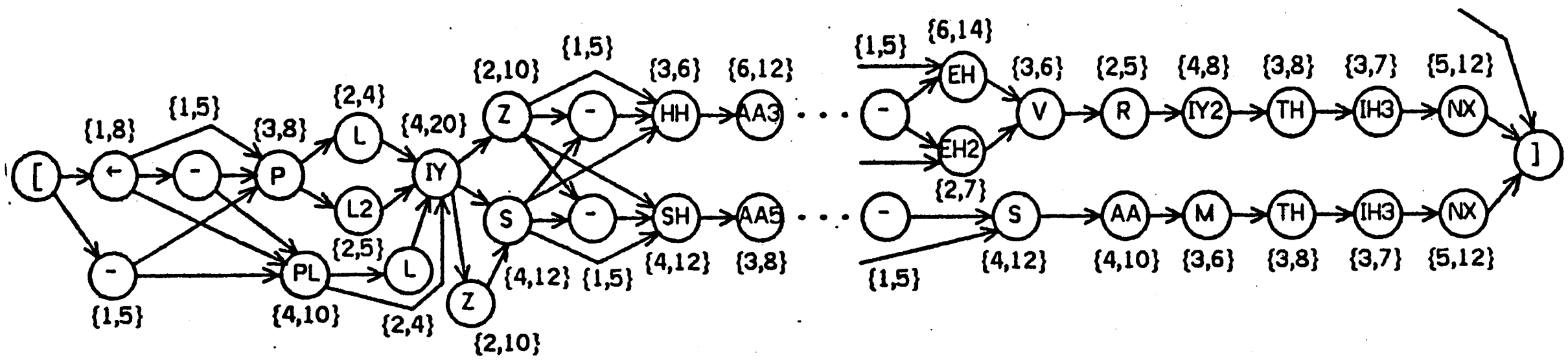


Figure 15-6 Network After Replacing all Words With Phone Subnetworks (Part of network not shown)

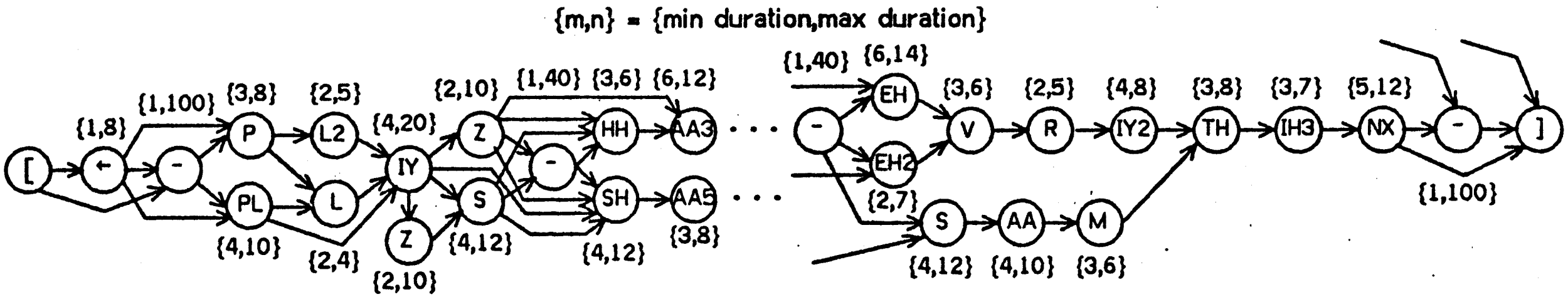


Figure 15-7 Network After Removal of Redundant Paths, Application of Juncture Rules and Addition of Optional Silence Before Final State (Part of network not shown)

symbol (word) in the BNF grammar. Some words may occur more than once in the grammar. However, each will be given a unique id number. This is necessary for the recognition process to recognize that a transition has been made from one word to the next. This problem is especially prevalent in cases where the same word can occur two or more times in a row, such as in a series of digits. In this case, the BNF grammar must be constructed so that there are two lists of digits where the first list is linked to the second and the second is linked back to the first. This will generate connected network states with the same word lexical number but with different word id numbers.

The minimum duration is the minimum expected phone duration in centi-seconds. This is used by the recognition process in calculating transition probabilities. An inter-state transition which does not allow minimum duration in the exited state will be penalized. Similarly an intra-state transition which causes the maximum duration to be exceeded will be penalized.

The transition count is the minimum number of inter-state transitions needed to reach the final state. This is used by the recognition process to prune "dead-end" paths. For example, if state J has a transition count of N, and at a point in the recognition the number of segments left to go is less than N, then state J is pruned from consideration since its path cannot reach the final state before the end of the utterance. A side effect of this pruning is that on the last segment, the only state being considered is the final state (with transition count of 0).

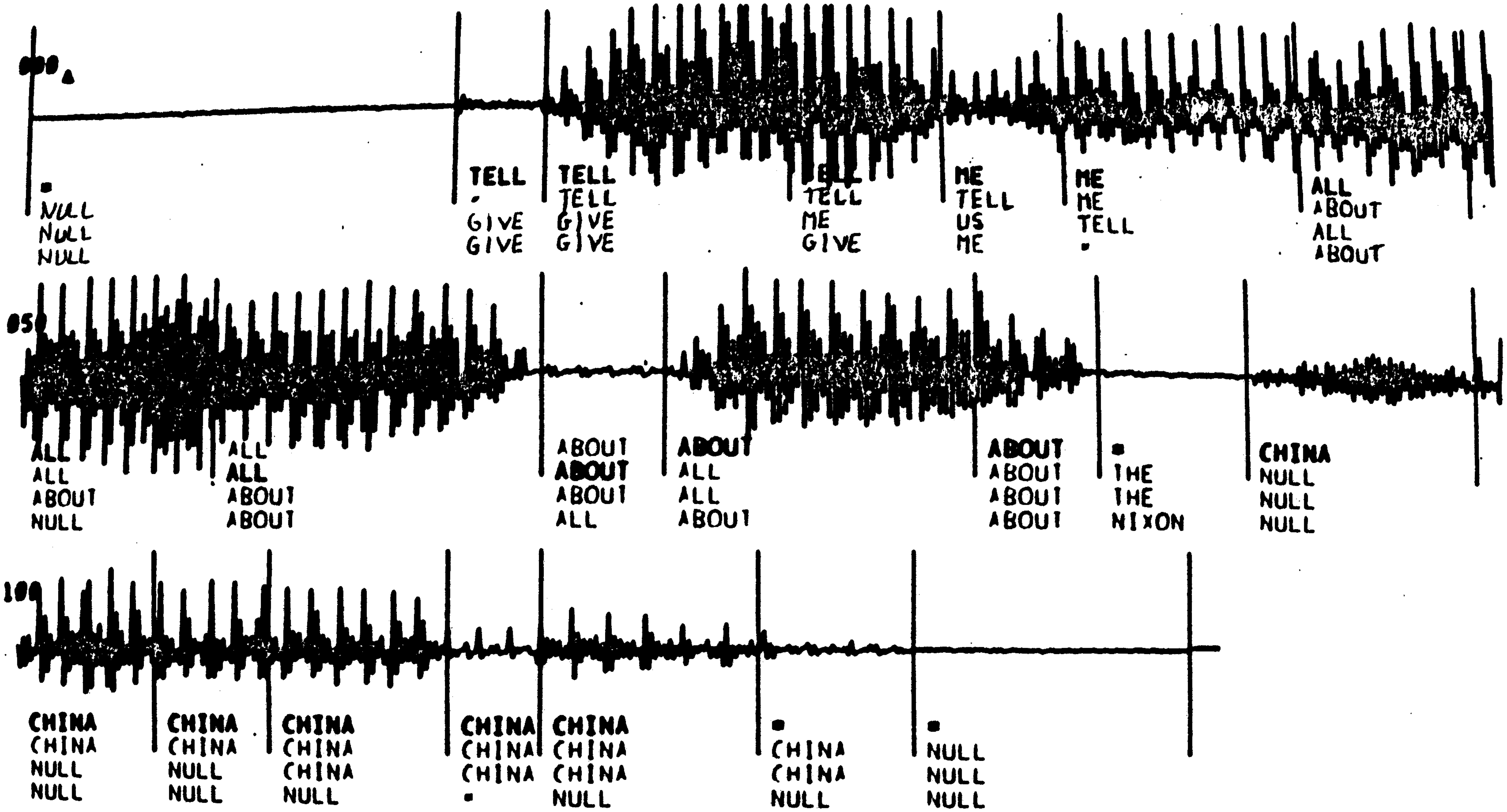
Discussion. There are several other technical aspects of the knowledge compiler such as subsumption, intersection states, determination of transition count, and other details that are omitted for lack of space. Many of these tend to be efficiency issues in space and time. We have omitted discussion on how semantic, pragmatic, and prosodic knowledge would be incorporated into the network. We have been able to formulate solutions to these problems in several cases and do not anticipate any major difficulties in using these classes of knowledge within the Harpy framework.

15-5. THE RECOGNITION PROCESS

In this section we will describe how knowledge is used in the recognition of an unknown utterance. Figure 15-8 gives the complete graphical trace of an utterance recognition. The unknown sentence is digitized, segmented, matched with phone templates, and compared with the sentences in the language using the knowledge network. The utterance with the best score is chosen as the sentence and displayed at the bottom of the waveform. In the rest of this section we will briefly described each of these steps in the recognition process using the example in Fig. 15-8.

15-5.1 Digitization

All the experiments using the Harpy system were conducted in a computer terminal room environment (approx. 65 dbA) using an Electrovoice head-mounted close speaking microphone (model no. RE51). The signal was low pass filtered at 4.5 KHz using a Kronhite (model no. 3750R) and sampled at 10 KHz sampling rate. The time sharing operating system was modified to accept high data rate signal without any data loss. Utterance begining and end detection was done using amplitude and zero-crossing measurements of the signal (Gill, et al. 1978). The digitized waveform is displayed as in Fig. 15-8. Note that the second line of the waveform is a continuation of the first and the third is a continuation of the second.



I heard "TELL ME ALL ABOUT CHINA"

Figure 15-8 A Graphical Trace of the Recognition Process

15-5.2 Segmentation

The continuous speech signal is segmented into discrete components using ZAPDASH (zerocrossings and peaks in smoothed and differenced waveforms) parameters. The segmentation process is described in detail by Gill et al., (1978). A recursive top down segmentation procedure is used to identify segments based on features such as silence, voicing, frication, peak detection and dip detection. Figure 15-8 shows a typical segmentation achieved by the system. Note that segmentation boundaries are marked by vertical bars. Note that vowels are usually divided into two or more segments. There is one missing segment boundary at the juncture of the words all and about. The l-schwa boundary is missing. But given the constraints of the language the system recovers from this error.

15-5.3 Phone Template Matching

Spectral characteristics of each segment are determined by using LPC analysis at the midpoint of the segment. The average segment length is about 5 centi-seconds, resulting in a factor 5 improvement in signal analysis over systems which perform LPC analysis each centi-second. For each segment the LPC coefficients of the segments are matched with speaker dependent templates generated using the procedure described in Sec. 15-4.4. The LPC minimum distance residual metric is based on Itakura (1975). In certain versions of the Harpy system only templates which are permissable in that context are computed. Note that implicit within this discussion is the assumption that given enough allophone templates it is reasonable to attempt labeling of segments using pattern matching techniques. This was by no means an accepted approach to phone labeling until recently.

15-5.4 Recognition and Match

We will illustrate the steps in matching and search process using Fig. 15-9. Figure 15-9a is part of a knowledge network in which the sentence "Tell me all about China." is legal. Only the first few states of the knowledge network are shown. Note that given this network, only four phone labels are permissible in the sentence initial position, namely IH2, G, -, and T. Given this information, the phone template matching procedure only performs these matches. The values of the acoustic match distance are given on the right side of the boxes in Fig. 15-9b. Note that the silence phone /-/ has the smallest distance of 0.23 for the first segment, /T/ has distance of 1.46, /g/ a distance of 1.57, and /IH2/ has a distance of 2.26. Since these are the only legal phones for this position, none of the other phone distances are evaluated.

As the acoustic matches are generated, Harpy begins the recognition process. The goal of the recognition task is to find an optimal sequence of phones satisfying two criteria. The sequence must represent a legal path through the knowledge network and should consist of phones with high acoustic match probabilities (the actual calculation produces "- log probability").

The beam search technique used by the Harpy system is a heuristic search technique which locates a near-optimal sequence of phones that is consistent with the network. Beam search is a technique in which a group of near-miss alternatives around the best path are examined. By searching many alternatives simultaneously, this method avoids the need for backtracking. The search is executed by creating and examining a tree structure of phones whose connections are consistent with transitions in the knowledge net. Each ply (or column) in the recognition tree, given in Fig. 15-9b, represents the matching associated with one segment of the digitized utterance.

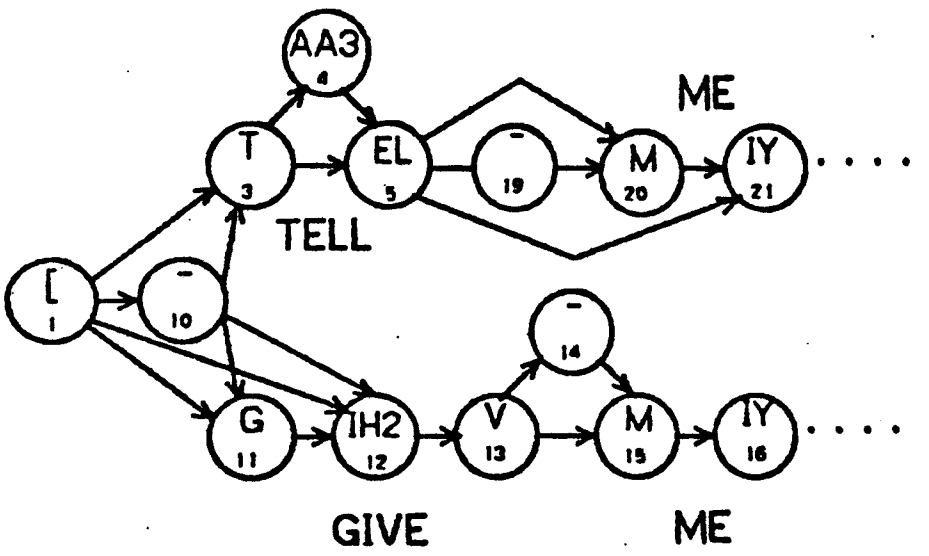


Figure 15-9a Partial Knowledge Network

Figure 15-9b Partial Recognition Tree

Figure 15-9 Recognition Process Using Beam Search

The root node of the tree is the sentence beginning state "[" of the network in Fig. 15-9a. Harpy begins the search by taking all the legal phones that can follow from the sentence initial state and entering them in the recognition tree. Refering to Fig. 15-9b, each box in the second ply (column) gives phone and the word associated with candidate state. Next, a path probability is calculated for each candidate. This is a cumulative value based on the path probability of the previous node and the acoustic match probability of the current node. The path probability value is indicated on the left side of each box. The path with the best probability (boxes with double lines in Fig. 15-9b) is determined. In this case the "-" state has the best value. All the remaining candidates are then compared with it. Those candidates that fall below a threshold of acceptability are pruned from further searching. In Fig. 15-9b the pruned states are indicated by dotted boxes. The successors of each surviving candidate are expanded, based on the information in the knowledge network.

As can be seen from Fig. 15-9a, there are four successors to the /-/ state and three successors to the /t/ state. Note that, although not shown in Fig. 15-9a as a specific pointer, each state can transition to itself. This permits two or more segments to be matched to the same state. Thus, the segmentation can introduce extra segment markers without significant penalty while a missing segment causes problems. Therefore, potential missing states must be created as optional states during the network creation time.

When the successor states are copied onto the recognition tree as candidates for match with the next segment in the segment list, two or more states may generate the same successor. Instead of retaining two independent paths through the same node, we can collapse them into a common path, avoiding redundant computation. In Fig. 15-9b both the /-/ and the /t/ states generate /t/ as a successor. Only the path with the highest prior value is relevant at this point. The deleted path is indicated by a double broken-line box. Thus, lesser-valued paths can be discarded because their path probabilities can never exceed the one with the highest value. The path probabilities are calculated as before, the best path is established and unpromising alternatives are pruned.

The forward search continues, expanding the recognition tree and saving those connections that satisfy the threshold, until we reach the end of the utterance. It was not possible to show the entire tree. Figure 15-9b shows only a portion of the recognition tree, i.e., up to the fourth segment of the waveform shown in Fig. 15-8. Returning to Figure 15-8, we see under each segment the words associated with the best four states in the candidate list for that segment.

So far we have explained the beam search technique informally by means of an example. The following algorithmic description of the search process provides a more formal and precise definition of the search and matching process.

```
Start:
Put the initial state in the list of previous states:
Repeat the following for each segment:
  For each state A in the list of previous states:
      Take A out of the list
      For each successor B of state A:
          Compute the cumulative loglikelihood P of
              transitioning from A to B as:
          P(A→ B) = P associated with A + acoustic match between the
               phoneme associated with B and the current segment
          If P(A→ B) is better than the loglikelihood of transitioning
             to B from any other state:
                put the path A→ B and its loglikelihood in the list of
                    candidate paths
          If P(A→ B) is better than any other loglikelihood computed
              so far for this segment:
                  P(A→ B) becomes the best loglikelihood P(best)

  Compute the beamwidth E as function of segment duration
  For each path in the list of candidate paths:
    If the distance between the loglikelihood associated with
      the path and P(best) is less than E:
        put the "to" state of the path (B) in the list of previous states.
```

Of all the paths that survive at the end of the utterance, the one with the best path probability is the solution we are seeking. This is the only path that satisfies the two criteria of the recognition process. It provides the best interpretation of the acoustic matches, while satisfying the constraint of the knowledge network.

A backtrace through the recognition tree reveals the desired solution indicating the phone and word assignments associated with the best path. Since the pointers for each surviving path in the beam of the recognition tree are retained till the end, this turns out to be a straight forward look-up operation and does not involve any search. Note that what appears to be the best choice for each segment in the forward search may not, in fact, be part of the globally best sequence discovered by delaying the decision till the end. Thus, local errors introduced by segmentation and acoustic matches are recovered by delaying commitment to a particular path until the forward search is completed. Therefore the forward search may be errorful without affecting the final solution. Refering to Fig. 15-8 again, the final word choice selected by the backtrace is indicated brighter than the rest of the candidate choices. Note that on line two of the waveform the assignment of <u>all</u> and <u>about</u> involves the second best choices.

Occasionally, the heuristics associated with the beam search miss the optimal path, but because the acoustic matches are less than perfect, attempting to find the optimal path at great cost and effort leads to little or no improvement in the overall performance.

15-6. PERFORMANCE

In this section we will present a summary of the Harpy system performance for several tasks. A more detailed description of these results is given in Reddy et al., (1976). Table 15-1 summarizes some of the interesting results from these experiments. The first four tasks essentially have no syntactic or other higher level knowledge. The rest of the tasks use some form of syntactic and other task-dependent information.

Table 15-1 contains several relevant dimensions for each task. Each task was tested with several speakers, as shown in column 2. The pair of numbers in parentheses indicates the number of male and female speakers in each group. Column 3, 4, and 5 are indicative of the measure of complexity of each task. Size of the vocabulary is usually quoted as a measure of complexity. This is usually acceptable for languages which do not use syntactic or other task dependent constraints. The average branching factor is a measure of lexical fanout (Reddy, et al., 1977 pg. 41) allowed by the grammar. The branching factor based on entropy (2^{entropy}) of the language has been used by Cohen and Mercer (1975) and Goodman (1976). The IBM group uses the term "perplexity" to refer to this measure. The last two columns give performance of the Harpy system on these tasks: word error rate and millions of instructions executed per second of speech.

Effect of telephone: We conducted an experiment to evaluate the effect of telephone input on Harpy performance. We note from Table 15-1 (task 2) that using telephone speech increases the word error rate by a factor of 3 to 4.

Effect of speaker: Task 3 in Table 15-1 shows the effect of using speaker independent phone templates. Again, we see that there is a significant increase in error rate. Rabiner and Sambur (Rabiner, 1976) state that they get about 4% word error using a speaker independent recognition system. The IBM group (Bahl, 1978) reports less than 1% word error rate using speaker dependent training in a quiet environment.

Effect of vocabulary: The principal effect of the vocabulary size appears to be to increase the space required. It appears to have very little effect on the accuracy or speed of the system.

Effect of branching factor: Clearly, increase in branching factor increases the error rate and processing required. Different groups seem to prefer different measures. The IBM group (Bahl, 1978) claims that the performance of their system correlates highly with the entropy based perplexity measure. The Harpy system does not exhibit any such clear correlation. Both the digits task (perplexity 10) and the 1011 word abstract retrieval task (perplexity 5) both have the same error rate. Further, the retrieval task with a perplexity of 5, is 8 times slower than the digits task with higher "perplexity". Further, Harpy has a better error rate of only 12% with the spelling task (perplexity 26) while the IBM group reports an error rate of 30% using a task with a perplexity of 20. Independent of what measure is best to use, Harpy clearly demonstrates that it is possible to have high performance connected speech input for task-oriented constrained languages.

	No of Speakers	Vocab. Size	Average Branching Factor	Entropy based B.F., "Perplexity"	Word Error rate (%)	MIPS reqired to process a second of speech
Connected digits -Speaker dependent	10 (7+3)	10	10	10	2	3.5
Connected digits -Telephone	4 (3+1)	10	10	10	7	3.5
Connected digits -Speaker independent templates	20 (14+6)	10	10	10	7	3.5
Alphabet -Spelling task	2 (2+0)	26	26	26	12	5.2
Abstract retrieval	5 (3+2)	1001	33	5	2	28

Table 15-1

15-7. DISCUSSION

There are several important factors that contributed to the success the Harpy system. Foremost among them are the representation of knowledge and the beam search technique. Several speech related decisions that led to improved performance were: use of large number of allophone templates, dynamic adaptation of template characteristics for a new speaker, the decision to encode intra-word phonological phenomeme into the pronunciation dictionary, lpc analysis, Itakura metric, and so on. Several questions arise about the generality and extendability of Harpy-like systems. We will raise some of these and present our current views on these topics.

1. Doesn't finite state grammar restriction make Harpy useless for use with natural language? From a computational point of view the answer appears to be "No". If one is willing to place a restriction on the length of sentence then the language can be modeled using finite state graph (FSG) representation. A simple FSG where any word can follow any other word, would accept all legal sentences but would permit many more illegal sentences as well. However, one need not take such a drastic step. As we observed in Sec. 15-5, if one is willing to accept some loss of constraints, the language can be represented by a suitable FSG which covers the language.

2. Doesn't FSG representation require a large amount of memory for complex tasks? The answer is "Yes" but given the advances in computer technology it doesn't matter. A graph structure that attempts to capture every possible variation requires a great deal of memory. Our current estimate is that a complex language may require a few million states (the 1011 word task has about 15,000) and that semiconductor memories capable of holding such state information would be available for a few hundred dollars. We have

also been studying issues of paging knowledge networks from secondary memory and the possibility of using multi-level networks rather than a single integrated network. Multi-level networks require a greater degree of dynamic interpretation during execution. Both of these appear technically feasible. It will be purely a question of space-time-cost trade-off in system organization.

3. How can Harpy handle sentences that are not part of its grammar? In general it cannot, but neither can any other system or human. To be able to handle new words and new sentence constructs that are not part of one's vocabulary and language one needs a knowledge acquisition facility. We have been developing concepts that would permit Harpy-like systems to acquire new words and new constructs. This requires an ability for the system to recognize that the unknown utterance is inconsistent with its internal knowledge and activate partial matching and word spotting type networks which are substantially less constrained. If all the words are known but they are ungrammtical (as in "sleep roses dangerously young colorless"), the sentence construct, if desired, can be assimilated into the word network. If one or more words are unknown then one needs a "speak and spell" program to learn the new words or variations of existing words.

In conclusion, we see that there are many possible avenues for evolving Harpy-like systems. It appears that many of the present limitations of the system can be removed without losing the high performance aspects of the system. However, these changes are likely to take many years to come given the limited research activity in the area.

15-8. ACKNOWLEDGEMENTS

The research reported here was supported by the Defense Advanced Research Projects Agency and monitored by the Airforce Office of Scientific Research under contract number F44620-73-C-0074.

We would like to thank Gary Goodman and Ron Cole for their helpful comments on this manuscript and John Zsarnay for his help in generating the illustrations.

15-9. REFERENCES

Atal, B.S., & S.L. Hanauer, "Speech analysis & synthesis by linear prediction of the speech wave", J. Acoust. Soc. Amer. vol 50, no 2, 1971.

Bahl, et al., "Automatic recognition of continuously spoken sentences from a finite state grammar", in Proc. IEEE-ICASSP Conf. 1978, Tulsa, Okla.

Baker, J.K., "The DRAGON system - An overview", IEEE Trans. ASSP, vol 23, 1975.

Baker, J.K., "Stochastic modeling for automatic speech understanding", in Speech Recognition: Invited Papers of the IEEE Symp., D.R. Reddy (ed.), 1975.

Cohen, P.S., & R.L. Mercer, "The Phonological component of an automatic speech recognition system", in Speech Recognition: Invited Papers of the IEEE Symp., D.R. Reddy(ed.), 1975.

Erman, L.D. & V.R. Lesser, "A multi-level organization for problem solving using many, diverse cooperating sources of knowledge", in Proc. 4th IJCAI, 1975, Tbilisi, USSR.

Erman, L.D., "The Hearsay-II speech understanding system", ch.16 this volume.

Gill, G.S. et al., "A recursive segmentation procedure for continuous speech", tech. rep., Computer Science Dept., Carnegie-Mellon U., 1978.

Goodman, G., "Analysis of Languages for man-machine voice communication", tech. rep., Computer Science Dept., Carnegie-Mellon U., 1976.

Itakura F., & S. Saito, "Analysis synthesis telephony based on the maximum likelihood method", Proc. 6th Int. Congr. Acoustics, 1968.

Itakura, F., "Minimum prediction residual principle applied to speech recognition", IEEE Trans. ASSP, vol 23, 1975.

Klatt, D.H., "Review of the ARPA Speech Understanding Project", J. Acoust. Soc. Amer. vol 62, no 6, 1977.

Lea, W.A., "Speech Recognition: Past, Present, Future", ch. 4 this volume.

Lowerre, B., "A comparative performance analysis of speech understanding systems", Ph.D dissertation, Computer Science Dept., Carnegie-Mellon U., 1976.

Lowerre, B., "The Harpy speech recognition systems", Tech. rep., Computer Science Dept., Carnegie-Mellon U., 1977.

Markel, J.D., "Digital inverse filtering - A new tool for formant trajectory estimation", IEEE Trans. Audio Electroacoust., vol AU-20, 1972.

Newell, A. et al., Speech Understanding Systems: final Report of a Study Group, 1971. (Reprinted by North-Holland/American Elsevier, Amsterdam, Netherlands, 1973).

Oshika, B.T. et al., "The role of phonological rules in speech understanding research", IEEE Trans. ASSP, vol 23, 1975.

Rabiner, L.R., "Preliminary results in recognition of connected digits", IEEE Trans. Acoust., Speech, Signal Processing, April 1976.

Reddy, D.R., "Computer recognition of connected speech", J. Acoust. Soc. Amer. vol 42, 1967.

Reddy. D.R., et al., "The Hearsay speech understanding system: an example of the recognition process", Proc. IJCAI-73, 1973, Stanford, CA.

Reddy, D.R., "Speech recognition by machine: A review", Proc. of the IEEE (Apr. 1976).

Reddy, D.R., et al., "Speech Understanding Systems: summary of results of the five-year research effort at CMU", Computer Science Dept., Carnegie-Mellon U., 1977.

Tappert, C.C., et al., "The use of dynamic segments in the automatic recognition of continuous speech", tech. rep., RADC-TR-70-22, IBM, Systems Development Div. Research, Triangle Park, N.C., 1970.