Proceedings of The Institute of Acoustics

A MULTILEVEL COMPUTER SYSTEM FOR AUTOMATIC SPEECH UNDERSTANDING

ROGER K. MOORE

DEPT. PHONETICS AND LINGUISTICS, UNIV. COLLEGE LONDON

Introduction

Recent years have seen the emergence of large-scale speech recognition projects called speech understanding systems (Lesser et al 1975, Walker 1975, Klatt 1977). These systems (SUS's) have been based on the view that independent knowledge sources (KS's), for example: acoustics, syntax and semantics, all cooperate to understand an utterance without necessarily recognising each word correctly (Newell et al 1973). A major consideration in the construction of these machines has been the organisation of the KS's (Reddy and Erman 1975). This paper outlines a technique for describing the structure of any SUS. Also, it presents a computer system, based on this technique, which enables the construction of many different SUS structures.

The Descriptive Technique

The methodology for describing SUS organisation results from an investigation into the key concepts of "level" and "unit" within a complex pattern processor (Moore 1976). This study revealed two fundamental types of process: (a) between level or interlevel processes, and (b) within level or intralevel processes.

An interlevel process relates units at one level to units at another. Usually, a string of units at a lower level is related to a single unit at a higher level. For example, the string of phonemes ((s)(ɪ)(k)(s)) is related to the lexeme (6). This relationship may be written as an equivalence, and a list of such equivalences constitutes a dictionary. Conversions from higher level units to lower level strings, or vice versa, are performed with respect to the information contained within a dictionary. Consequently, an interlevel process is termed the dictionary look-up or LUP process. In pattern recognition terms, the dictionary performs as a training set. Thus all forms of classification, recognition and matching are considered to be LUP type processes.

An interlevel process relates the units at a level to other units at the same level. Specifically, it collects single units to form a string and in so doing applies sequential constraints.

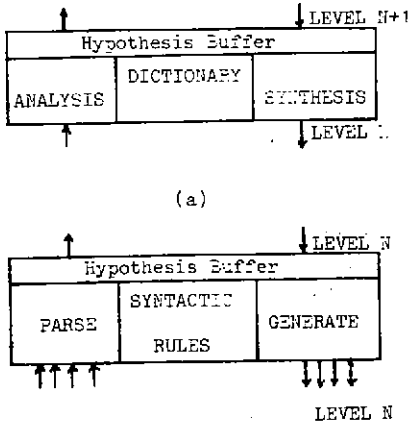

Figure 1. Diagramatic form of (a) an LUP process, and (b) a RLS process.

Conversely, it takes a string and splits it into its constituent units. The sequential constraints may be expressed in a grammar as syntactic rules. As a result, an intralevel process is termed the syntactic-rules or RLS process.

A pattern processor consists of a number of LUP and RLS processes connected together in multiple hierarchies. The nature of the processes allows a mixture of bottom-up and top-down processing. The lowest levels of the hierarchies contain the least abstract units of the system. These are the input-sensory and output-motor channels. Higher levels contain more abstract representations of this basic data.

The descriptive technique may be used to compare any speech recognition structure with that of another. Also, particular features may be discussed in a revealing way. For example, the classical problem of segmentation has several interesting interpretations in terms of the descriptive technique; segmentation by LUP (word spotting), or segmentation by RLS (rates of change of parameters). Above all, the technique provides a vehicle for designing new structures, and it was to this end that a computer implementation was developed.

Computer Implementation

The computer implementation of the descriptive technique is essentially a program which enables the creation of different configurations of LUP and RLS processes. Each process is an independent task under a time-sharing operating system. However, to underline and exploit the fundamental similarity between processes, all LUP processes share a common LUP procedure, and all RLS processes share a common RLS procedure. Communication between processes is performed via a common area of memory accessible by the two procedures.

A controller task organises the process tasks thereby providing various facilities for operating the system:

(1) A system may be configured by: adding a process to the current configuration (ADD), deleting a process from the current configuration (REMOVE), saving the current configuration in a file (SAVE), or retrieving a previously saved configuration (CALL).
(2) A system may be run by: activating all the processes in the current configuration (START), or de-activating all processes (STOP).
(3) A system may be debugged by: looking at the data stored in the communication buffers (DUMP), or sending information to a specified device (FEEDBACK).

At the present time (April 1978) the computer system has an inventory of twenty-seven processes; seventeen LUP and ten RLS. From these building blocks a number of different speech systems have been constructed.

Example Configuration

The system shown diagramatically in figure 2 can perform simple arithmetic operations in response to spoken commands with pauses between the words. The system understands a command such as "What-is three plus five ?" by placing three blocks and then five blocks in a model world. The system then counts the total number of blocks in the world and speaks the result "Eight". The model

world is displayed on a visual display unit showing the addition and removal of individual blocks. To deal with both addition and subtraction the system has both positive and negative blocks. A long period of silence following speech is recognised as "?" and terminates a phrase indicating that a response is required. If the speaker says "two minus ..... " the system will reply "Minus what ?". The command "Clear" erases all blocks from the model and the system responds with "OK".

The flow of data within the configuration is as follows: EARS is a RLS process which is the handler for a sixteen channel filter bank. Its output is a string of sixteen units representing spectral energy, once every ten ms. These strings are collected and segmented by STAB (RLS). STAB segments when the Euclidean distance between one spectral slice and the next is zero (i.e. silence). The output of STAB is thus a two-dimensional representation of a word; a spectrogram bounded by silence. SPECT (LUP) matches the spectrogram against its dictionary using a dynamic programming algorithm and the resultant word symbol is passed to SAME (RLS) and MTCH3 (LUP). SAME segments the string of word symbols when there have been a certain number the same in a row (silence is represented by a word symbol). This number is adjusted such that SAME segments after a pause following a phrase but not after a pause between words.

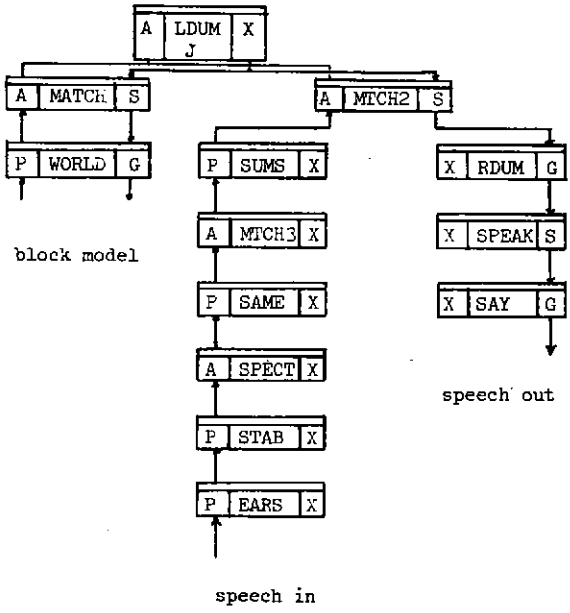

Figure 2. Diagramatic form of a speech understanding system.

An example string which SAME might form would be: ((S)(W)(S)(5)(S)(-)(S)(S)(7)(S)(S)(S)(5)(S)(S)), where (W) is the word symbol for "what-is", and (S) is silence. MTCH3 (LUP) merely echoes the output of SPECT except for (S) which it recognises as 'null' and therefore gives no output. MTCH3 also receives strings from SAME, and on those occasions it outputs (?). This is because, like MATCH and MTCH2, MTCH3 performs an absolute match against its dictionary, and unknown strings (i.e. those not listed in its dictionary) are classified as don't knows. The result of these operations is that the output of MTCH3 might look like:- (W)(5)(-)(7)(?).

SUMS (RLS) collects the output of MTCH3 and applies word syntax, parsing the symbols into strings in which the operators '+' and '-' are associated with the next symbol (usually a digit). MTCH2 converts the parsed symbol strings into a symbol which is used as a representation common to the three hierarchies. This symbol is passed to MATCH which synthesises from it a program of actions for the model world. For example, (+3) is converted to ((+)(+)(+)), where (+) means "add one positive block". WORLD (RLS) performs the actual operations.

When all actions have been performed, as indicated by the arrival of the single element string ((?)), WORLD outputs the contents of the world as a string; ((-)(-)(-)(-)) would mean that the world contained four negative blocks. MATCH turns this string back into the common symbol (-4).which is passed to MTCH2. MTCH2 synthesises ((-)(4)) and RDUM (RLS) simply splits the string into its elements: (-) and (4). SPEAK (LUP) then synthesises a string of control parameters for the speech synthesiser, and the response is spoken by SAY (RLS).

Conclusions

The computer implementation of the descriptive technique is proving to be a real aid for designing speech recognisers. It is true that a particular configuration could be more efficient if it were programmed directly, but it is possible that such a solution might not have been discovered had it not been expressed in terms of the two basic building blocks. The descriptive technique and its computer implementation forms what is possibly a unique conceptual and developmental aid for designing and comparing different speech recognition strategies.

References

(1) D.H. KLATT 1977 JASA 62, 1345-1366. A Review of the ARPA SUS Project.
(2) V.R. LESSER et al 1975 IEEE ASSP 23, 11-23. Organisation of the HEARSAY-II Speech Understanding System.
(3) R.K. MOORE 1976 Ph.D. Thesis. A Descriptive Technique for the Analysis and Design of Speech Understanding Systems.
(4) A. NEWELL et al 1973. Speech Understanding Systems, North-Holland Pubs.
(5) D.R. REDDY and L.D. ERMAN 1975 Speech Understanding, Acedemic Press, 457-479 Tutorial on System Organisation for Speech Understanding.
(6) D.E. WALKER 1975 IEEE ASSP 23, 397-416. The SRI Speech Understanding System

Acknowledgement

This research was supported by the Science Research Council of Great Britain.