FORMAL CONSTRAINTS ON ATN GRAMMARS FOR SPEECH RECOGNITION

Matthew Lennig

Bell-Northern Research

Montreal, Quebec

Paper presented at the Summer Meeting of
the Linguistic Society of America, Albuquerque,
August 1, 1980

Rapid advances in digital technology have led many researchers to become interested in the possibility of natural language communication between humans and machines. Although this might sound like something out of Battlestar Galactica, limited systems for understanding and producing speech are now commercially available. In order to increase the capabilities of these systems, powerful models of linguistic performance are needed both for speech understanding and speech production. Transformational grammar, while useful as a model of production, is of limited value as a model of perception, since it does not provide any procedure for parsing sentences. One of the most promising syntactic frameworks to be proposed for modelling both the encoding and decoding of language is the so-called Augmented Transition Network. The Augmented Transition Network (ATN) was first proposed by W.A. Woods in 1969 as a syntactic representation which could be used for the automatic parsing of text input. In this paper I will explore the consequences of applying Woods' ATN grammar to the problem of parsing speech input.

The speech recognition system I will discuss has been implemented on a small computer. It recognizes sentences from a limited syntax spoken naturally by a single speaker. The system consists of four components: a Preprocessor, which extracts acoustic parameters from the speech signal to be used in recognition, a Syllabifier, which divides the parameterized speech into syllables, a Sentence Recognizer, which directs a breadth-first search of the syntactic space to determine the most likely identity of the unknown sentence, and a Syllable Comparator, capable of calculating a measure of distance (or dissimilarity) between an unknown syllable and a stored template.

Figure 1 shows a sentence-level transition network capable of recognizing French date-and-time expressions. The initial state of this network is the circle labelled S/ in the upper left corner. The final state is S/END at the lower left. Any path through this network from the initial state to the final state represents a grammatical date-and-time expression. For example, the path corresponding to le premier janvier à midi et quart (January first at quarter past noon) begins at state S/ and first accepts the word le by calling a lexical subnetwork called LE/. The function PUSH transfers control to the subnetwork. When the final state of the subnetwork is reached, a POP arc returns control to the higher level calling network, in this case to the state S/LE. To get from state S/LE to state S/DATE, the arc PUSH PREMIER/ is followed. The lexical subnetwork for the word PREMIER/ is shown in Figure 2.

Figure 2 says that the word premier has two acceptable surface realizations: either it can be realized as the syllable prə followed by the syllable mje or it can be realized as a single syllable prəmje. These two alternatives reflect the fact that the Syllabifier sometimes divides the word into two syllables and sometimes leaves it as one syllable, depending on how it is pronounced. The lexical subnetwork compensates for syllabifier variability by providing alternative analysis paths.

When an ACCEPT arc is traversed, the Syllable Comparator is invoked. The function of the Comparator is to determine the acoustic distance (or dissimilarity) between the current input syllable and a stored syllable template corresponding to the phonetic transcription on the ACCEPT arc. The smaller this acoustic distance, the more likely it is that the input syllable is a token of the phonemic class represented by the template. Before the analysis path can traverse another ACCEPT arc, the next input syllable must be read.

After the analysis path has accepted either one or two syllables for the word premier, it executes the POP arc which returns control to the higher level network in state S/DATE. The analysis path then continues to advance through the network, accepting the words janvier, à, midi, et, and quart until state S/END is reached and the analysis is complete. The analysis has associated with it a total acoustic distance which is the sum of all the distances generated by all the ACCEPT arcs traversed. The

smaller this distance, the more likely it is that the analysis path for the sentence is correct. The task of sentence recognition consists of exploring many analysis paths, or hypotheses, in parallel and finding the one whose total distance is the smallest.

The ATN grammar of French date-and-time expressions also includes two constituent-level subnetworks which are referenced at various places in Figure 1, for example on arcs between states S/LE and S/DATE. These constituent-level subnetworks are called TEEN/ and N29/ and are shown in Figures 3 and 4, respectively. TEEN/ accepts a number between dix (10) and dix-neuf (19). N29/ accepts a number between deux (2) and neuf (9).

ATN grammars also allow arcs to execute arbitrary actions and to test arbitrary conditions. Although the French date-and-time syntax uses actions and conditions to handle liaison and other context-sensitive phenomena, I will not discuss this in detail.

When ATN grammars are used to parse text input, arcs labelled with terminal symbols are traversed only if the input symbol agrees with the symbol on the arc. In analyzing speech, however, the identity of the input syllable is unknown. All that is available are the acoustic distances between the input syllable and a number of stored templates. From these distance we know the relative likelihood that the input is a token of a particular phonemic class but we have no absolute knowledge on the basis of which to traverse or not traverse a particular arc. It may

turn out, for example, that the first syllable of a word is a poor acoustic fit to the stored template but the second and third syllables are excellent fits. Thus we must explore all analysis paths through the syntactic network in parallel and keep track of the cumulative acoustic distance of each hypothesis. Hypotheses are allowed to propagate freely through the network. The only restriction on their free propagation is that after traversing an ACCEPT arc, they become temporarily inactive until the next input syllable is read.

Since all possible paths through the network must be explored in parallel, an enourmous number of hypotheses are generated. Even the small subset of French generated by this date-and-time network contains over half a million sentences. Limits on memory and processing capacity make it necessary to eliminate incorrect hypotheses as soon as possible.

This can be achieved by applying Bellman's (1957) Principle of Optimality for finite state networks, which says the following: the minimum distance path from a given state to the final state is independent of how the given state was reached. Therefore, if two hypotheses arrive at a given state after having accepted the same number of syllables, only the one with the smaller distance need be preserved. The continuation of the path beyond that state will not be affected by how that state was reached.

Because the ATN is not a pure finite state network, Bellman's Principle of Optimality must be adapted to take account of the conditions and actions on arcs discussed above. Thus, before a hypothesis is deleted by application of the Optimality Principle, we must be sure that no condition can be satisfied by the deleted hypothesis that would not be satisfied by the hypothesis which is preserved.

To see how the Optimality Priciple would operate in the French date-and-time grammar, lets suppose one hypothesis has reached state S/MONTH in Figure 1. Suppose it has accepted the string of words: le vingt-et-un mars. Suppose a second hypotheses arrives in state S/MONTH after having accepted the word string le vingt-huit décembre. Both have accounted for the first five syllables of the input sentence in different ways.

Each hypothesis has a cumulative acoustic distance associated with it. By application of the Optimality Principle, we need only preserve one of these two hypotheses: the one with the smaller cumulative distance. The minimum path that either of them would follow from state S/MONTH to state S/END is unaffected by which of the two paths to state S/MONTH was chosen. Therefore, we may as well choose the minimum distance path to S/MONTH and disregard the other one.

It is this Optimality Principle which saves the recognition system from the otherwise inevitable combinatorial explosion of analysis paths. In order to take full advantage of the Optimality Principle to eliminate

suboptimal hypotheses as early as possible, we must be able to detect the crossing or intersection of two analysis paths as soon as it happens. We would like to control the propagation of hypotheses through the network so that a hypothesis in a given state, for example S/MONTH, will not advance beyond that state until all other hypotheses that will cross S/MONTH on the current input syllable have arrived there. As each hypothesis arrives at S/MONTH its distance is compared to the hypothesis already at that state. Only the hypothesis whose distance is smaller is preserved.

If we establish an ordering on the states of the network which is consistent with all possible paths through the network, then all that is required is that we expand hypotheses occupying earlier states first. This guarantees, in our example, that a hypothesis occupying state S/MONTH cannot be advanced until all hypotheses in earlier states have caught up with it.

An elementary result in graph theory (Berge 1962) is that an ordering on the states consistent with all paths is possible if and only if the graph contains no circuits. A circuit is a closed loop consisting of more than one state, as illustrated in Figure 5. Disallowing circuits in ATN grammars enables these grammars to be used for decoding speech input. Since circuits in a syntactic network do not seem to describe any linguistic phenomena which cannot be handled in other ways, the

proscription of circuits from ATN representations appears to be a useful formal constraint which is motivated by performance considertions.

REFERENCES

Bellman, R. 1957. Dynamic Programming. Princeton, NJ: Princeton University Press.

Berge, C. 1962. The Theory of Graphs and its Applications. Translated from French by A. Doig. New York: John Wiley & Sons.

Chomsky, N., 1957. Syntactic Structures. The Hauge: Mouton.

Woods, W.A. 1969 Augmented Transition Networks for Natural Language Analysis. Report No. CS-1, Computation Laboraory, Harvard University, Cambridge, Mass.