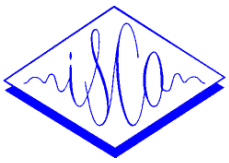


2nd International Conference on
Spoken Language Processing (ICSLP 92)
Banff, Alberta, Canada
October 12-16, 1992

EFFICIENT INTEGRATION OF COARTICULATION AND LEXICAL INFORMATION IN A FINITE STATE GRAMMAR

Antonio Bonafonte, José B. Mariño, Montse Pardàs

Dept of Signal Theory and Communications (UPC)
Apdo. 30002, Barcelona 08080, Spain
email: antonio@tsc.upc.es

ABSTRACT

In this paper, an efficient integration of lexical information in a phonetic finite states automaton is presented. The approach is specially suitable for modelling coarticulation occurring inter and intra words. This approach has been used to extend a speech recognition system in order to give the N-Best sentences of words, not acoustic units, and to build complex lexical trees.

I.- INTRODUCTION

Most of continuous speech recognition systems use syntactic constraints to improve their performance by limiting the search space and by reducing the entropy of the language. The most extended representation consists in finite states automata (FSAs) either for representing regular grammars or for n-grams. This representation has the advantage of its easy integration in one-stage dynamic programming algorithm.

The emissions of the automata can be either lexical or acoustic units. Last approach has the advantage that parts of the words can share states of the automata. In this way the search space is reduced a great deal. However, this representation has the drawback of the loss of lexical information. Such information is necessary in some applications. For instance, it is required when a stochastic grammar is used. It is also necessary to find the N sentences that best match acoustic models.

The paper is structured as follows. In the next section these two approaches are studied and an alternative representation is proposed being a trade off between them: lexical information is retained in one state of each word and a great minimization is accomplished. In section III, we compare the different approaches in terms of the size of the FSAs in a particular application. Also, the criteria followed to design Spanish phonological rules are enumerated. Section IV is devoted to generalize an N-Best algorithm to accounting for lexical information. Some recognition results are presented and discussed in the final section.

This work has been supported by TIC92-1026-C02-02

II.- GRAMMAR REPRESENTATION.

In order to constrain the search space, some recognition systems use a phonetic FSA which is a FSA where each arc represents a phonetic unit. There are some minimization algorithms that reduce the number of branches needed to represent the same language [1,2]. This minimization is specially useful in languages where inflections and derivations are regular, as it is the case of Spanish. This representation has the additional advantage that different pronunciations of the same word and coarticulation between words can be considered in an automatic way. For instance, if units u1-u2-u3 can be uttered in continuous speech as u1-u4, the automaton can be transformed to consider this option without noticing whether the coarticulation occurs inside a word or between words. This is a convenient way to model hard changes in speech, such as deletion and substitution of phones. It is also an easy way to consider context depending units. However, due to the minimization stage, there is no information about the relation between phonetic units and words. In order to obtain the word string, an additional structure, such as a lexical tree, is required. The design of the lexical tree is a tedious work when coarticulation between words is modelled.

Some recognition algorithms require lexical information in order to make some actions. For instance, if a stochastic grammar is used, the cost of a transition between words must be added to the acoustic match cost. In such cases, a common representation is a lexical automaton where each branch is a word. The recognition system substitutes each branch by its equivalent phonetic units. Sometimes, several utterances are possible for some words. This can be considered by defining different word for each transcription and adding branches of new representations in parallel with the original arc. The representation by means of a lexical FSA has two drawbacks. First, as states cannot be shared by different words, the size of the FSAs is much bigger than that of a phonetic FSAs. Second, although it allows to consider different pronunciations inside a word, the application of phonological rules to the boundaries of words is not immediate.

10.21437/ICSLP.1992-14

We propose a methodology to obtain a representation with

* different transcriptions of the words,
* coarticulation between words,
* lexical information in each arc associated with the beginning of a word and
* minimum number of states.

Formalization

A FSA is a 5-tuple $F = (Q, \Sigma, \delta, q_o, q_f)$ where

Q is a finite set of states,
Σ is a finite set of permissible input symbols,
δ is a mapping from $Q \times \Sigma$ to $P \subset Q$,
$q_o \in Q$ is the initial state,
$q_f \in Q$ is the final state.

In general q_o and q_f can be sets from Q.

In a phonetic FSA the elements of $\Sigma = \Sigma_{ph}$ are elemental units as phones or syllables. In a lexical FSA the elements of Σ are lexical units Σ_{lex}, usually words. The type of automaton would be denoted with the subscript "ph" or "lex" when necessary.

In the following P_{ph} and L_{lex} are assumed to be two optimum automata which represent the same language in different alphabets.

Modelling a language by a FSA can be done with words. However, if the number of words is high, the recognition units are usually smaller. We have called these units phonetic units. In order to compare P_{ph} and L_{lex}, a transformation of the lexical FSA to an equivalent phonetic FSA (L_{ph}) has to be done.

If w_i belongs to Σ_{lex} we design the automaton W^i_{ph} as a succession of states and branches. The only sequence of symbols that is accepted by this automaton is the sequence of acoustic units that defines the word.

The transformation of L_{lex} into L_{ph} consists on changing each branch of L_{lex} with label w_i by the automaton W^i_{ph} and on merging the states q_o and q_f of W^i_{ph} with the origin and end states of the branch.

Both P_{ph} and L_{ph} are representations of the same language. As P_{ph} is a minimum FSA, it could be obtained from L_{ph} by a minimization algorithm. The difference between the number of states and branches of these two representations depends on the application but in the most cases is relevant. For instance, there are a lot of words with the same ending and the same syntactic category. This means that there are parallel branches in L_{lex} which derive in a set of states in L_{ph} which can be minimized.

Alternative transcriptions of words

On real languages, some words can be uttered in different ways. For instance, some sounds can be omitted or changed. There are two ways to consider this phenomenon. The first consists on defining in L_{lex} different words for different transcriptions. This increases very much the size of the automaton L_{ph}. The second consists on modifying slightly the automata W^i_{ph} in order to model the variants of the words. Therefore different transcriptions can be considered without a significant increase of the total number of states.

Lexical information

As it has been mentioned in the introduction, some applications need lexical information associated with states. It can be supplied if a finite transducer (FT) is extended from the FSA L_{lex}. An FT is obtained by taking an FSA and associating an emission symbol on each move, or equivalently on each branch. An FT is a 6-tuple $\Im = (Q, \Sigma, \Delta, \delta, q_o, q_f)$ where Q, Σ, q_o and q_f stand for the same that in the FSA definition, Δ is the output symbols and δ is a mapping from $Q \times \Sigma$ to subsets of $Q \times \Delta$.

In the case we are interested on, Σ is Σ_{ph}, and Δ is $((\Sigma_{lex} + \emptyset) \times \text{Inic} \times \text{End})$. Inic = {true, false} denotes if the input symbol is the begin of a word. End = {true, false} denotes if the input symbol is the end of a word. The label of $(\Sigma_{lex} + \emptyset)$ denotes, in the case it is not $\emptyset$, the word which the input symbol belongs to. The necessity of both sets, Inic and End, will be seen later.

Minimization of the finite transducer

As it has been told, the size of P_{ph} is smaller than that of L_{ph} but it is not possible to build an FT associated with P_{ph} because some branches are shared by different words. We propose a minimization algorithm of the FT associated to L_{ph}, $\Im_{ph}$, that reduces the size of the transducer while retaining enough lexical information for the applications we have considered.

There are several algorithms to minimize the size of an FSA. For instance Hopcroft and Ullman[1] describe a method to build a canonical FSA. We have used the method proposed by Mariñol[2] which achieves a greater reduction of the size paying the prize of not being deterministic. However, if a parallel search is used, the size of the automaton is more important than the fact of being deterministic. The basic idea of the latter algorithm consists in taking all the branches with the same input label and, if possible, to redefine another different set of branches with the same label but smaller than the first set. The new automaton accepts the same language but it is not ensured a relation between the branches of

the original and the new set. We have applied this algorithm as follows: (the sign * means any value)

- Repeat until convergence
  - Repeat $\forall$ i
    - Select all the arcs with input label $f_i \in \Sigma_{ph}$
    - Preserve all the arcs with output label {*, true, *}
    - Form a set with the branches whose output symbol is {*, false, true}. Minimize this set according to the algorithm above mentioned. The new set will have as output label {ø, false, true}. Note that this can reduce the number of nodes. Therefore, branches with other labels f_j, may be minimized[2]
    - Form a set with the branches whose output symbol is {*, false, false}. Minimize this set according to the algorithm above mentioned. The new set will have as output label {ø, false, false}

Lexical information is preserved in branches that correspond with the beginning of a word. Output labels (w_j, true, *) mean that this branch is the beginning of word w_j. Speech recognition algorithm can propagate this information at the same time that the cost of each path.

Minimization of states could have been done by the initial parts of the words but the former criterion has been shown to be more convenient due to the structure of Spanish words.

Coarticulation

In the representation explained up to now, words are modelled as if they were uttered in connected speech. However, coarticulation is one of the aspects that makes more difficult the recognition of continuous speech. Some of the effects are the omission of sounds, substitution of phones, changes on syllabification and creation of diphthongs.

Coarticulation can be considered by defining contextual words in Σ_{lex} and by considering these new words on L_{lex}. This is a tedious work and the size of the automaton increases very much. We have modelled coarticulation by applying phonological rules to $\Im_{ph}$. When these rules, which can be optional or mandatory, are applied, the output labels are adopted from the old ones. To clarify this, suppose the rule

$$f_1 f_2 \rightarrow f_3$$

When the sequence $f_1 f_2$ is found in $\Im_{ph}$, a new branch with input label f_3 is created from the origin of f_1 to the end of f_2. The output labels can be the same that either f_1 or f_2. Besides that, if f_3 is labelled {*, true, *}, predecessors of f_3 must change their labels to {α, β, true} being α and β the old values. The sucessors of f_3 should be changed analogously if output labels of f_3 are {*, *, true}.

After the transformation of $\Im_{ph}$ one may update the automata W^i_{ph}. At the end, W^i_{ph} describes all the considered utterances of word i. A lexical tree of each word can be easily derived.

If the rule is mandatory, path $f_1 f_2$ must be broken. Similar criteria are used with other phonological rules. This modelling can produce initial branches that follow some branches which are not the end of a word. A transition between words happens when a path passes from an ending branch to an initial branch. Therefore both labels, Inic and End, are needed.

After modelling coarticulation the minimization algorithm should be applied.

III.- COMPARISON OF DIFFERENT REPRESENTATIONS

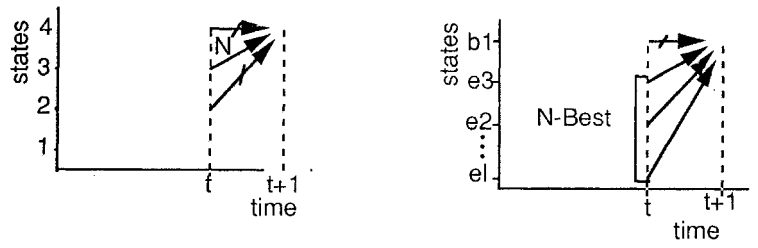
The structure presented was mainly developed to adapt the N-Best algorithm that is described in the following section. The language we want to recognize is the positive closure of L, where L represents any number from zero to one thousand. The sublexical unit we use is the demisyllable. At this moment, the phonological rules have been designed in order to cope with five situations that occur in continuous speech in Spanish: re-syllabification, deletion of phones, change of unvoiced 's' into voiced 's' before voiced consonants, creation of diphtongs and change of occlusive consonants into approximant ones in some contexts. In table I we present the number of branches and the averaged number of predecessors of each branch for P_{ph}, $\Im_{ph}$ and L_{ph}, this last without considering coarticulation. It can be noted that the size of $\Im_{ph}$ is not much greater than that of P_{ph}, although lexical information has been preserved.

	# of branches	# pred. (average)
P_{ph}	135	4.3
L_{ph}	239	8.3
$\Im_{ph}$	161	5.0

Table I. Number of branches and average of the number of predecessors of each branch as a function of the lexical information retained

IV.- THE N-BEST ALGORITHM

The grammatical representation described above can be used by most of the N-Best algorithms currently used, if the syntax is represented by a FSA[3,4]. We have used it to generalize the algorithm presented by Mariño and Monte[5]. In this algorithm, the N-Best hypotheses are propagated through the states of a HMM network. Two different cases are



Inside the model: transition driven by the best candidate.

Between models: the candidates must be sorted.

Figure I.

considered: transition inside models and transition between models (fig. I). Inside the models, transitions are based on the first hypothesis. The decision whether a transition between models has to occur or not is also based on the best candidate but, if a transition occurs, then the N-Best predecessors are chosen. As a consequence of driving the paths inside a model by the best hypothesis, two different paths converging into a model must be combined in the first state of the model. Therefore, the boundaries between these models must be the same for every one of these paths. In the case that a model represents a word, this combination might be an important drawback. However, in our system the problem is not so considerable because we represent different entrances in a word in order to model the coarticulation between words. Moreover, the selection of candidates takes place at each unit of the word and not only at the beginning of them.

The algorithm just reviewed gives the N-Best strings of units but some strings can be lexically equal if several phonetic transcriptions of words are allowed.

To solve this problem, when sorting the N-Best paths in a transition between models, paths should represent different sentences. This can be easily done if one uses a representation of the grammar with lexical information as the proposed above. The recognition algorithm identifies the word when each path passes through an initial unit and it propagates this information. Each path has a lexical string associated which can be compared when sorting the different candidates. The path is updated when it passes through an arc with output label {*, true, *} and it comes from a branch with label {*, *, true}.

We have also tried a simplification consisting in looking only at the last word of the lexical string. In this case, when a transition between models happens, two paths can be propagated together if

- the previous word is different or
- they arrive together (which implies that they do not represent the same sentence by an induction principle).

For instance, if all the hypotheses come from the same word, as in the case shown in figure II, only those which arrive together can be propagated. The decision is based on the best candidates, for example, if Cost(H1) < Cost(H1') then H1 and H2 would be propagated.

It can be shown that this simplification is reasonable if words are not too short. In fact, although some words we have used have only one phone, the results obtained with this simplification have been identical to those obtained with the first version of the algorithm we have presented.

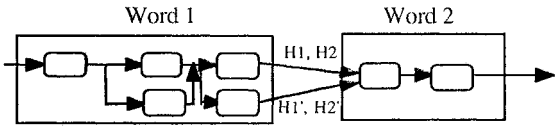


Figure II. Both paths arrive from the same word so the criterion is "to propagate the complete path whose first candidate is the best".

We have tested the algorithm recognizing telephone numbers. The training database consisted on 40 strings of numbers from 0 to one million. The results obtained recognizing without lexical information present hypotheses which are different phonetic transcriptions of a few different semantic sentences. With lexical information only the best transcription of each meaning is calculated.

REFERENCES

[1] Hopcroft J.E., and Ullman J.D.(1971) "An n log n Algorithm for Minimizing States in a Finite Automaton", CS71-90, Computer Science Department, Stanford Univ., Stanford, Calif

[2] Mariño J., Nadeu C. and Lleida E.(1988) "Finite State Grammar Inference for Connected Word Recognition", *Proc. of the European Signal Processing Conference, Grenoble, Sept. 1988, Vol 2, pp. 1059-1062*

[3] Schwartz, R. and Y.L. Chow "The N-Best Algorithm: An Efficient and Exact Procedure for Finding the N Most Likely Sentence Hypotheses", *Proc. of the ICASSP-90, Albuquerque, April 1990, Vol 1, pp. 81-84*

[4] Soong, F and Huang, E. "A Tree-Trellis Based Fast Search for Finding the N Best Sentence Hypotheses in Continuous Speech Recognition", *Proc. of the ICASSP-91, Toronto, May 1990, Vol 1, pp. 705-708*

[5] Mariño J. and Monte E. (1989) "Generation of Multiple Hypotheses in Connected Phonetic-Unit Recognition by a Modified One-Stage Dynamic Programming Algorithm", *Proc. of the European Conf. on Speech Communication and Technology, Paris, Sept. 1989, Vol 2, pp. 408-411*