A MULTI-LINEAR REPRESENTATION AND RULE FORMALISM FOR PHONOLOGY AND PHONETICS IN TEXT-TO-SPEECH SYNTHESIS

S. Frenkenberger[1], M. Kommenda[1] and S. Moosmüller[2]

[1] Institut für Nachrichtentechnik und Hochfrequenztechnik, Technische Universität Wien

[2] Institut für Sprachwissenschaft, Universität Wien

ABSTRACT

In general, phonological and phonetic transformation rules not only depend on the phonological or phonetic context, but on all kinds of linguistic information (morphology, syntax, etc.). The proposed multi-linear representation and rule formalism provide an elegant way to bring together information pertaining to different structural levels. The joint processing of these data is of paramount importance in high quality text-to-speech synthesis.

1. INTRODUCTION

There are different approaches to speech synthesis from text; however, every system capable of high quality speech production must include phonological and phonetic components, and these will have to account for processes on both, the segmental and the supra-segmental level. They will be situated near the end of the linguistic processing part of the whole text-to-speech system, right at the interface to the signal generation module.

Nonetheless, phonological and phonetic transformation rules particularly depend on all kinds of linguistic information (morphology, syntax, rhythmic structure,...) which have been derived from the input text in previous analysis steps. Therefore a clearly specified way to store and to process these intermediate results is of vital importance.

For the German text-to-speech system GRAPHON [4] we have evolved an appropriate methodology, which consists in

(1) a multi-linear representation of the information pertaining to different linguistic levels;

(2) a rule formalism operating on this representation.

These two pillars are presented in the following sections, before we will discuss an example to illustrate their application within computational phonology and phonetics for text-to-speech synthesis.

2. THE MULTI-LINEAR REPRESENTATION

2.1 Linguistic Background

The goal of our multi-linear representation is to handle linguistic information concerning different structural levels (graphemic text, morphological and syntactic structure, syllable structure, phoneme sequence, etc.). While segments pertaining to one and the same structural level may be arranged in serial order, such a linear arrangement cannot be found for the multi-level description. Still, this description is not strictly more-dimensional, inasmuch as the different lines of information proceed in parallel. This is why the single lines are called <u>tiers</u> and the whole ensemble a <u>multi-linear</u> representation. Note that the tiers can be synchronized with each other at selected points. This concept reflects ideas developed within the theoretical framework of non-linear phonology [2]. With regard to its practical realisation we have been influenced by the DELTA system [3].

2.2 The Data Structure LIFT

The above concept has been implemented in the form of a data structure called LIFT (LIsts For Transcription, an acronym making allusion to the means of going from one level to another). LIFT may hold all the information ocurring in the course of text-to-speech conversion, starting from the graphemic representation and possibly ending with an allophone sequence or even with speech synthesizer parameters (like pitch or formant frequencies). Between them there are tiers containing the morphological structure, syllable boundaries, prosodic information, and so forth. The elements within a tier are bound up with one another to form a list. Furthermore, whenever two elements belonging to different tiers are to be associated with each other, this connection can be established by setting a link between them.

Figure 1 shows a simplified representation of a section of LIFT generated during the conversion of the german word *Lippen* (*lips*). Each column represents a particular tier. All elements on a row with the symbol '#' on their left side are linked together, i.e. there is a link between these elements as indicated by the dotted lines.

3 THE RULE FORMALISM

3.1 Motivation

The multi-linear representation LIFT provides an elegant way to bring together information concerning different linguistic levels. As for actually exploiting this joint representation in linguistic processing, we need an appropriate rule formalism, which extends the traditional linear description of consecutive segments (e.g. in the well-known SPE-format [1]) to the multi-linear case. In particular, it should be possible to formulate a rule context referring to different structural levels at the same time. On the other hand, the rule-writing linguist should not be supposed to know any implementational details of LIFT nor to have any programming experience.

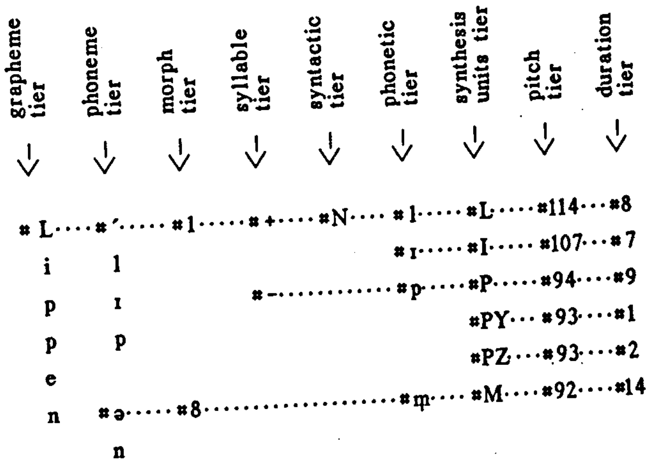

Figure 1: Section of LIFT generated during the conversion of the german word *Lippen* (lips).

3.2 The Rule Syntax

In our approach a context-sensitive rule consists of a condition part and an action part. The first comprises one or more conditions, which may be combined by the logic operators '&' (and) and '|' (or). The latter contains one or more actions. In the first place these can be thought of as transformations; however, in view of the procedure of rule application we need additional functions to control both the scanning of the input and the transformation process.

3.3 The Rule Interpreter GRIP

The described formalism has been put into practice by means of the rule interpreter GRIP (Graphon Rule InterPreter) which translates a source textfile of rules into executable instructions.

3.4 Co-operation with LIFT

GRIP rules operate on a LIFT representation of a given text. They are applied once in a serial order, and every single rule scans an entire text unit (normally a sentence). To be more precise, via a so-called condition pointer a particular tier of LIFT is selected to constitute the input for the rule. (Note that, thanks to the links between elements of different tiers, any structural context associated with a particular element of the selected tier can be tested in the rule's condition part.) A second pointer, the so-called action pointer, directs the actions to a second (possibly the same) tier. During rule application both pointers are moved further along the tiers.

4. APPLICATION IN PHONOLOGY AND PHONETICS

4.1 Example

To illustrate syntax and processing of rules, consider the following phonological transformations encountered in standard Austrian German:

ə-deletion	(1)
progressive nasal assimilation	(2)
syllabification of the nasal	(3)

These can be formalized as follows:

$$\text{ə} \rightarrow 0 \,/\, [+\text{obstr}] \;\_\_\; \text{n} \qquad (4)$$

$$\text{n} \rightarrow [\alpha\ \text{place}] \,/\, \begin{bmatrix} +\text{obstr} \\ \alpha\ \text{place} \end{bmatrix} \_\_ \qquad (5)$$

$$[+\text{nas}] \rightarrow [+\text{syll}] \,/\, [+\text{obstr}] \_\_ \qquad (6)$$

Thus:

Lippen	(lips)	→	['lɪpm̩]
laufen	(to run)	→	['laufɱ̩]
Regen	(rain)	→	['re:gŋ̍]

Using GRIP the rule for e.g. *Lippen* looks like the following:

```
[+E(-1,p) & +E(0,ə) & +E(1,n)]
    del skip(1,r) chg_el(m̩)          (7)
```

The condition part (in square brackets) of (3) consists of three conditions that are concatenated by the operator '&'. Whenever processing of a rule is initiated, the condition pointer addresses the first element of a specific tier, in this case the phonetic tier. Subsequently the condition pointer is moved along this tier. As soon as the element 'ə' is addressed, all three is-element conditions (+E) are met since the preceding element is 'p' and the succeeding element is 'n'. This causes execution of the action part. First, the element addressed by the action pointer ('ə') is deleted (del); next, the action pointer is moved one element further (skip) (note that whenever the element addressed by the action pointer is deleted, the latter addresses the preceding element.); finally the element 'n' is changed into 'm̩' (chg_el).

Taking into acount the context 'pən' and 'bən' only, rule (7) is an elegant way to implement the transformations (1,2,3). Since the rules (4,5,6) are valid for any context, (7) has to be extended. By exploiting a feature-based representation of phonemes GRIP allows to combine (4) and (6) in a single rule (8).

```
[+E(0,ə) &
   +F(-1,CONS) & -F(-1,NAS) &
   +F(1,CONS) & +F(1,NAS) &
   -E(-1,r) & -BEG(1,SYLL) ]
del() skip(1,r) chg_F(SYLL)          (8)
```

(8) reads as follows:
Every 'ə', preceded by a non-nasal consonant (+/-F denotes presence/absence of the specified feature), succeeded by a nasal consonant, is deleted (del) and the subsequent nasal becomes syllabic (chg_F(SYLL)).

Note that the sequence 'ər' is treated within a separate rule and thus is excluded in (8). Finally the condition -BEG(1,SYLL) serves to prohibit a syllable boundary between 'ə' and the nasal, e.g. *genommen* (taken): [gə'nɔmən] and not *['gn̩ɔmən].

With regard to the implementation of (5), basically three seperate rules would have to be written, in order to account for each place of articulation (velar, labial, and labiodental). The elegant notation of (5) is due to the notion of "α-place". In (9) we therefore introduce "accept", a GRIP action to copy feature bundles from neighbouring phonemes.

```
[+E(0,n̩) & +F(-1,OBSTR) &
   ( +F(-1,ANT) |
     +F(-1,HIGH & BACK) ) ]
accept(-1, HIGH/BACK/LAB/
           ANT/COR )                 (9)
```

(9) reads as follows:
'n̩' preceded by an obstruent which is either anterior or high and back, accepts the features high, back, labial, anterior and coronal from the obstruent. Since these 5 features serve to describe the place of articulation, the nasal is assimilated, yielding 'm̩', 'ɱ̩', 'n̩', or 'ŋ̍'.

Note that in the condition part the palato-alveolar articulation ('ʃ tʃ...') is excluded. In fact, this should have been done in (5) as well. IPA does not provide for a palato-alveolar nasal, thus (5) takes for granted that in such a case the nearest possible place of articulation will be chosen. With regard to a computer implementation implicit assumptions of this kind have to be analysed very carefully.

4.2 Conclusion

The above rules primarily refer to the phonetic tier. However, other rules, in particular rules concerning supra-segmentals, obviously depend on various kinds of linguistic information (e.g. morphological and syntactic structure).

Within the text-to-speech-synthesis system GRAPHON, phonological and phonetic rules fill up LIFT, exploiting the information previously generated in the morphological and syntactic analysis (cp. fig. 1). To this end neither condition part nor action part of GRIP rules are bound any longer to a single tier as it was mostly the case in the introducing example in 4.1.

The joint processing of context conditions making reference to several structural levels at the same time significantly extends the linear representation of segments in SPE rules. Besides their practical relevance within text-to-speech synthesis, LIFT and GRIP provide the linguist with a powerful tool for rule development and test.

ACKNOWLEDGEMENTS

The research project has been receiving financial support from the "Fond zur Förderung der wissenschaftlichen Forschung (FWF)".

REFERENCES

[1] CHOMSKY, N. & HALLE, M. (1968), *"The Sound Pattern of English"*, New York: Harper & Row.

[2] CLEMENTS, G. (1985), *"The geometry of phonological features"*, Phonology Yearbook 2. 225-253.

[3] HERTZ, S. R. (1988), *"The Delta programming language: an integrated approach to non-linear phonology, phonetics, and speech synthesis"*, Working papers of the Cornell Phonetics Laboratory 2, 69-122.

[4] KOMMENDA M. & POUNDER A. (1986), *"Morphological Analysis for a German text-to-speech System."* Proc. COLING'86, Bonn. p. 263-268.