ARIPUC 17, 1983, p. 87- 109

AN ORTHOGRAPHY NORMALIZING PROGRAM FOR DANISH*

PETER MOLBÆK HANSEN

The paper is a description of a program which transforms Danish orthographic texts into so-called normalized notation, i.e. a format which can serve as input to a text-to-speech-by-rule algorithm which is under development at the institute. The main features of the program are described, and the peculiarities of Danish orthography which motivate these features are presented. The capabilities and the limitations of the program are illustrated, and the possibilities of future improvements and extensions of the program are outlined.

I. INTRODUCTION

In Molbæk Hansen (1982) I reported on the construction of a grapheme-to-phone algorithm for Danish. That paper reflects a first approximation to a solution of the main problems connected with the design of a computer program capable of converting unrestricted Danish text to a normalized notation which is consistent with the pronunciation of the text and hence acceptable as input to a letter-to-sound-by-rule program.

During the past year the main features of the text normalizing program (henceforth TNP) were developed, and the basic code has been written. The present paper presents the basic design of the TNP and the factors (including some important characteristics of Danish orthography) which have been decisive in selecting that particular design.[1]

In section II some general features of text-to-speech-systems are touched upon, and a rudimentary typology of such systems is suggested. This furnishes the background for a characterization of the text-to-speech algorithm that is being developed for Danish.

---

*) *This work is carried out for the Telecommunications Research Laboratory within the synthesis-by-rule project.*

In section III the features of Danish orthography which motivate the choice of TNP-type are discussed, and the main features of the TNP and of its output, i.e. the normalized notation, are presented.

Section IV contains a description of the organization of the data part and of the processor part of the TNP. Some examples are given to illustrate what the program actually does to unrestricted Danish text.

In section V, finally, the capabilities as well as the limitations of the TNP are commented upon, and some perspectives of overlaying the TNP with facilities for operating with syntactic information are presented.

II. THE CHOICE OF A TEXT-TO-SPEECH SYSTEM

A. TYPES OF SYSTEMS

As is well known, alphabetic writing systems exhibit various degrees of consistency in the sense of Molbæk Hansen (1982, p. 128), ranging from near one-to-one correspondences between letters and phonemes, as in Finnish, over various approximations to more or less complex, yet contextually definable many-to-one correspondences between letters (letter combinations) and phonemes (phoneme combinations) as in French, to the abundance of many-to-one and, more seriously, one-to-many correspondences dominating such orthographies as Danish and English.

In some orthographies there are many irregularities, but such that each of them affects only a few words. In other orthographies there are only one or a few irregularities or other shortcomings, but such that a major part of the lexicon is affected by them. This is by and large true of the Russian and the Italian orthographic systems in which the most serious problem is that distinctive word stress is not indicated, cf. Sherwood (1978) and Lesmo et al. (1978). Another example of how a major part of the lexicon may be crucially dependent - as far as the predictability of the pronunciation is concerned - on an ambiguous spelling is the word final sequence *-ent* in French, which is pronounced [ã] in nouns and adjectives but is mute in 3rd person plurals of verbs.

Thus, although any text-to-speech algorithm has to perform at least some normalizing, it is not surprising that the various existing text-to-speech algorithms differ widely both in general build-up and in complexity. Among the existing text-to-speech systems two main approaches seem to be dominant:

Some systems are exclusively or predominantly rule oriented, i.e., except for a small preprocessor identifying a handful of words with extremely exceptional spellings or with a special phonetic behaviour, they consist of one monolithic rule component which must take care of both letter-to-sound rules

- including "rules" changing exceptional spellings - and phonological and phonetic rules - including rules for manipulating acoustic parameters. In such systems exceptional spellings are changed by "rules" of the same formal status as the rules referring exclusively to context of the expression plane (whether graphemic, phonemic or phonetic). This can be done because, except for cases of homography at word or morpheme level, the spelling of a word or of a morpheme is a reliable cue to its identity. This is the approach of Maggs et Trescases (1980) who take care of the above mentioned problem with French *-ent* by including in their rule system a large number of rules of the type *souv[ent]* → *ã* (in their notational format meaning "*'ent'* becomes [ã] between *'souv'* and space"). The Swedish system developed by Carlson and Granström also belongs here, being based on the explicit desire *"to minimize the lexicon and make the inventory of rules more exhaustive"* (Carlson and Granström (1975, p. 22)).

Other systems consist of several components of comparable complexity, typically including at least (1) a component taking care of deviating spellings by (hopefully) finding them in a more or less comprehensive morph lexicon which contains entries for a large number of morphs, each entry being linked with information on the pronunciation of the morph in question and on its syntactic and other grammatical behaviour, (2) a letter-to-sound rule component, applying to the spellings which are not found in the lexicon and hence considered normal, (3) one or more components taking care of prosodic rules, and (4) a component computing the acoustic parameters. The MITALK text-to-speech system for American English is of this type, cf. e.g. Allen (1976, 1981).

It is probably no coincidence that the lexicon minimizing systems were developed for French and Swedish: the orthographic irregularities of these languages are fewer and simpler than the irregularities of the English orthographic system. However, the choice of system also depends on the applications aimed at, on demands for speech quality, and on demands for real time performance. As a rule, the quality is high in the lexicon type system. The inclusion of a comprehensive lexicon, on the other hand, is to some extent incompatible with demands for (limited memory and) real time performance, cf. Sherwood (1978, p. 670).

B. THE DANISH SYSTEM

The text-to-speech system being developed for Danish is essentially a combination of the two types: we try to combine the advantage of having access to a morph lexicon including grammatical information with the advantage of being able to change, improve, and experiment with the rule system in the easy and flexible way this can be done if the rules are written in a special, linguistically oriented high level programming language based on the ideas of Carlson and Granström (1975). For more general information on the system, in particular the

rule component and the construction of a rule language compiler, see Holtse (1982).

This particular combination also allows us to divide the text-to-speech algorithm into two natural parts: the TNP and the rule component. Obviously, it is conceptually more satisfactory not to have to include "rules" like the French one mentioned above.

Since we aim at high quality output, and since Danish orthography is of the highly irregular type, it became clear at a relatively early stage that the TNP must include a morph lexicon that is basically of the same type as the one described by Allen (1976 and 1981) and developed and improved for English during the last fifteen years. That it actually differs in many details is due primarily to the special features of Danish orthography, as we shall see below.

So far, considerations of memory requirements and program efficiency play only a minor role in our scheme.

III. GENERAL CONDITIONS AND INDICATIONS FOR A TNP

The ideal TNP should, of course, be capable of converting the orthographic representation of any Danish sentence to a normalized notation from which the pronunciation of the sentence can be derived in an unambiguous way.

This ideal goal implies that such important phenomena as sentence stress and intonation patterns should be derivable by rule from the normalized notation. In view of the fact that such phenomena still await thorough research it seems a good policy to start with a TNP which (1) is capable of normalizing single words correctly in the overwhelming majority of cases where this can be done in a principled way, and (2) is designed in such a way that future extensions taking care of sentence phonetic phenomena can be easily and harmonically added to it. This has been the leading principle in the design of the TNP. The more specific requirements were mainly dictated by the specific properties of Danish orthography.

In this section those features of Danish orthography which have been decisive for the design of the TNP will be outlined. In Molbæk Hansen (1982) I mentioned some of these features in a rather unsystematic fashion. At that time I did not fully realize the implications of some of the problems, and I overestimated the importance of others. In the following subsections I shall describe and illustrate those features of Danish orthography which I now consider the most significant ones - both quantitatively and qualitatively - for any attempt to write a TNP for Danish.

A. THE REGULAR SPELLING OF FOREIGN WORDS

For some reason one is apt to believe that irregular spellings occur more often in foreign words than in "genuine" Danish words, i.e. roughly words inherited from Old Danish and medieval loan words from Low German. This is true only if foreign words are defined narrowly as recent loan words, say, words borrowed in the 20th century. However, apart from these there is a large number of older loan words of Latin or Greek origin which are now more or less assimilated to the Danish sound system, and for these the spelling is impressively regular.

One only has to read a few pages in any dictionary (ordinary or retrograde, depending on what one is looking for) to become convinced that if a word is found to contain a morpheme belonging to the group of such classical loans, then, in the overwhelming majority of cases, the pronunciation of the whole word is straightforwardly derivable by rule. This is particularly obvious in the case of words containing obligatorily stressed suffixes like e.g. the verbalizing suffix *-er-*, borrowed (via French *-er* or German *-ier-en*) from the Latin 1st conjugation infinitive ending *-are*, yielding Danish infinitives in *-ere* with stress and stød on the penultimate syllable. Since information on stress and stød is crucial for eventually determining the pronunciation, the identification of such stressed suffixes is, of course, essential, but such an identification would be of less interest if the stems preceding such suffixes were typically characterized by irregular letter-sound correspondences. Fortunately, that is not the case. In Holmboe's dictionary (Holmboe (1978)) there are roughly 1400 verbs of this type, and the overwhelming majority of these exhibit a regular spelling, irrespective of whether or not the *-er-* is considered part of the root. Similar conditions prevail in other suffixes belonging to this category, e.g. *-itet*, *-ens*, *-isme*, *-at* as in *identitet* 'identity', *tendens* 'tendency', *kommunisme* 'communism', and *professorat* 'professorship', respectively.

Foreign words - in particular the numerous ones of Latin origin - which do not contain a stressed suffix also typically take stress according to a general rule, viz. the last vowel followed by a consonant is stressed, cf. *kommando* 'command', *kaskade* 'cascade', *melon* 'melon', *analyse* 'analysis', although this rule is far from being without exceptions, cf. Rischel (1970).

Obviously, these regularities should be exploited somehow in the TNP. We shall see below how this can be done.

B. THE IRREGULAR SPELLING OF GENUINE DANISH WORDS

The major part of the irregular spellings occur in the bulk of genuine Danish words which by and large can be divided into monosyllables and initially stressed disyllables with schwa, orthographically *e*, as the second vowel.

Three main features of the spelling of such words are important for the design of a TNP:

(1) certain vowel letters, notably *i*, *y*, *u*, and *o* are ambiguous, cf. *sidst* /sisd/ 'last' vs. *vidst* /vesd/ 'known'; *tykke* /tygə/ 'thick (plur.)' vs. *stykke* /sdøgə/ 'piece'; *pund* /pun?/ 'pound' vs. *bund* /bon?/ 'bottom'; *ost* /osd/ 'cheese' vs. *post* /pɔsd/ 'mail'; *slog* /slo:?g/ 'hit' vs. *klog* /klɔ:?g/ 'clever';

(2) vowel length is unpredictable from the spelling in many monosyllables, cf. *sal* /sa:?l/ 'hall' vs. *bal* /bal?/ 'ball'; *ben* /be:?n/ 'leg' vs. *pen* /pɛn?/ 'pen';

(3) the occurrence of stød is unpredictable from spelling in many words, cf. *solen* /so:?lən/ 'the sun' vs. *skolen* /sgo:lən/ 'the school'; *skov* /sgɔv?/ 'wood, forest' vs. *tov* /tɔv/ 'rope, wire'.

The following facts should also be taken into consideration:

(4) the bulk of homographs occurs in genuine Danish words, primarily due to one or more of the factors 1-3, cf. *lod* /lo:?d/ 'let (vb. pret.)' vs. *lod* /lɔd/ 'destiny'; *læs!* /lɛ:?s/ 'read!' vs. *læs* /lɛs/ 'load'; *føl* /fø:?l/ 'feel!' vs. *føl* /føl/ 'colt';

(5) the majority of common and frequent compounds are composed of elements belonging to the genuine vocabulary, cf. such common compounds as *ægteskab* 'marriage', *sagfører* 'lawyer', *lommetyv* 'pickpocket', *brombær* 'blackberry' (in which *brom-* is a "quasi-genuine" element which does not occur in isolation, cf. the English word *cranberry*), and/or their initial or final elements belong to this group: *overassistent* 'senior clerk' (in which *over* is the genuine element, and *assistent* is a regularly spelled loan word of the classical type mentioned above), and *regentskifte* 'exchange of sovereign' (in which *skifte* is the genuine element);

(6) double consonant letter before a schwa alternating with word final single consonant letter is typical of such words, viz. in case the vowel is short, cf. *kys* 'kiss' - *kysset* 'the kiss' (vs. *lys* 'light' - *lyset* 'the light' with long vowel), *hjem* 'home' - *hjemmet* 'the home', etc.

C. THE LOCAL INFORMATION IN DANISH WORDS

The rightmost letters of a Danish word contain most of the information needed in order to eventually determine its pronunciation. The leftmost letters contain less information than the rightmost letters, but more information than the medial letters.

The importance of the rightmost letters of a word form is primarily due to two factors: (1) in the majority of cases such important phenomena as word stress, vowel length in stressed syllables and stød in stressed syllables can only be determined from the particular combinations of roots with suffixes and/or endings, (2) a handful of very frequent final letter combinations symbolize either an ending or the final part of a stem, and the interpretation of such final sequences as one or the other is crucial for the pronunciation.

This may be illustrated by the word final letter combination *en*. In *arsen* /ar'se:ʔn/ 'arsene', *pen* /pɛnʔ/ 'pen', and *igen* /i'gɛn/ 'again' the final sequence *en* belongs to the root. In *arsen* the vowel is long /e:/, and the word has stød according to the general rule that word final stressed syllables have stød if their vowel is long. In *pen* and *igen*,*en* represents /ɛn/, but *pen* has stød, whereas *igen* is stødless (stød is not predictable from the surface segmental structure of a stressed word final syllable with stød basis and short vowel). In *solen* /so:ʔlən/ 'the sun' *en* represents the allomorph /ən/ of the definite singular common gender ending, which has the property of retaining the regular stød in roots like *sol* (with long vowel, cf. *arsen* above) and of adding stød to roots like *søn*, which are stødless in other forms. In *talen* /ta:lən/ 'the speech' *e* represents the final schwa of the regular stødless disyllabic word *tale*, whereas the *n* represents the allomorph /n/ of the same ending as in *solen* and *sønnen*.

Similar one-to-many correspondences between spelling and morphological/phonological structure are characteristic of practically all word final letter combinations which can appear as (parts of) endings and also of a few suffixes consisting of one or more schwa syllables, cf. such frequent final letter sequences as *ens*, *et*, *ets*, *er*, *ers*, *erne*, *ernes*, *ene*, *enes*, *e*, *s*, *t*, *ede*, *edes*, *ende*, *ere*, *eres*, *else*, *elses*. A frequent word final letter combination like *el* behaves exactly like e.g. *en*, although it does not ever represent an ending: it represents a schwa syllable in words like *middel* 'means', *adel* 'nobility' (both with stød, cp. *sønnen* and *solen* above), and *sadel* 'saddle' (stødless, cp. *talen* above); it represents root final /e:ʔl/ in *kamel* 'camel' (cp. *arsen* above); it represents root final /ɛl/ in *farvel* 'good bye', (cp. *igen* above), and it represents root final /ɛlʔ/ in *model* 'model' (cp. *pen* above). Final *e* often represents root final /ə/ of many disyllables belonging to the genuine vocabulary - among Danish phonologists often referred to as β-words - such as *tale* 'speech', cf. above, *kone* 'wife', and many others; in the following sections such letters or letter sequences will be referred to as quasi-

endings when they are root final, and the remaining parts of such roots, e.g. the *kon* of *kone*, will be referred to as quasi-roots.

Other final letter sequences represent structures that obligatorily entail stress on the preceding syllable. This is true e.g. of "suffixes" like *ium* as in *radium* 'radium', etc. Others represent, more or less uniquely, suffixes which obligatorily take stress themselves if not followed by other suffixes, cf. *ik* as in *musik* 'music', etc.

That the leftmost letters of a word form are also important for the phonological behaviour of the whole word is mainly due to the fact that stød on the root is obligatory (if it is segmentally permitted) after certain prefixes, cf. cases like *skue* 'see, view' vs. *beskue* 'take a view of', where the occurrence of stød on the latter form is entirely predictable from the presence of the prefix *be-*.

In this case, as well as in all the other cases involving stress and stød, the heart of the matter is - as is well known among Danish phonologists - that the occurrence and placement of stress and stød in Danish word forms is by and large predictable from the combined information provided by (a) the identity of certain bound morphemes, (b) the segmental structure of root morphemes, (c) the morphological type of the word form, provided that the segmental structure of roots is considered at a rather abstract level, see e.g. Rischel (1970) and Basbøll (1972).

What all this amounts to is that, once certain initial and final conditions are determined, the phonological structure of the whole word form is also basically determined, and that the medial part of a long word form typically exhibits regularity or at most minor segmental irregularities in its orthographic shape.

Apart from cases like *ium* where the spelling uniquely defines the suffix, it is a problem with most of the above mentioned affixes that their spelling will not *per se* identify them. In many cases this is due to the fact that the spelling might as well be part of a larger structure, as was illustrated by the examples with endings. Since the morphological status of such final sequences is so crucial for the determination of stress and stød, it becomes an important task for a Danish TNP to identify these sequences correctly.

D. INDICATIONS OF DANISH ORTHOGRAPHY FOR A TNP

The above mentioned facts of Danish orthography point to a TNP with the following characteristics:

(1) It should have access to a lexicon of morphs and other structures containing crucial information of the kind needed to derive the phonetic shape of a word form by rule.

(2) The morphs to be placed in the lexicon should be (a) all or almost all genuine roots - since these contain the major irregularities; (b) all or almost all affixes - since these contain crucial information and help to delimit the roots. The other entries to be placed in the lexicon should be quasi-morphs of the sort which phonologically act in a morph-like way, cf. e.g. the quasi-ending *-el* which was mentioned above, and a quasi-root like *ad* in *adel* 'nobility'. (This word may thus be said to consist of a quasi-root and a quasi-ending.)

(3) It should scan word forms maximalistically, i.e. in a longest-match-first fashion, from the right end of the word as well as from the left end of the word, to check the input word form against the lexicon. In most cases in running text all morphs should be identified correctly by this approach, the normalized shape of the word forms should be retrievable, and phonologically relevant grammatical information pertaining to the morphs should be accessible. In cases where the word or part of it is not identified - typically when there is an unidentifiable medial residue such as the many foreign roots which are not in the lexicon - that part (or that whole word) should be considered regular.

E. NORMALIZED TEXT

When it is decided that the items to be placed in the lexicon should be all affixes and all roots belonging to the core of genuine items, cf. section II D, much of the design of the normalized notation which is to be the output of the TNP is also determined: If, e.g., a word like *officer* /ɔfi'se:ʔr/ 'officer' is taken to be regular and hence not to be placed in the lexicon, its spelling must be considered normal, and this means that the endings, suffixes, and genuine roots in which final *er* represents unstressed /ər/ must be normalized in such a way that they will not come out ending in *er* in the same way as *officer*.

The normalized notation is comparable to the so-called World English Spelling (WES), see Dewey (1971), although its motivation is of quite another kind. At the moment it is not defined rigidly (partly because its final design may have to depend on technical considerations and possibilities), but an operational version of it which I use in the daily testing work, cf. section IV, has the following main properties:

Word final occurrences of the letters *t* and *s* are separated from roots by a + if they represent endings, cf. e.g. *spis+t* 'eaten' vs. *gnist* 'spark', and *fod+s* 'of a foot' vs. *klods* 'block'.

Word final occurrences of the letter *e* are separated from preceding roots and quasi-roots by a + if they appear as endings or quasi-endings (/ə/) in cases where the preceding syllable has (stress and) stød, cf. e.g. *extern+e* 'external (definite or plural)', *Johann+e* (a personal name), *radikal+e* 'radical (definite or plural)', etc.

In word final mono- or disyllabic sequences with *e* (representing schwa) as vowel letter(s), the first or only *e* is considered initial in an ending or quasi-ending, if the word form in question is stødless, otherwise the (first or only) *e* is dropped. In both cases the letter preceding the *e* is taken to be root final, and the root is separated from the ending or quasi-ending by a hyphen, cf. e.g. the following normalized notations: *mand* 'man', *mand-n* (spelled *manden*) 'the man', *mand-ns* (spelled *mandens*) 'the man's', *kon-e* 'wife', *kon-er* 'wives', *sad-el* 'saddle', *sadl-er* 'saddles', *hus* 'house', *hus-t* (spelled *huset*) 'the house', *hus-e* 'houses', *hus-ene* 'the houses', *mål-ne* (spelled *målene*) 'the measures', *rat* 'steering-wheel', *ratt-t* (spelled *rattet*) 'the steering-wheel', etc. In a similar way, *-e* or, in some cases, just - is inserted before certain endings and quasi-endings in order for the word forms in question to conform to the same structural features as the ones just mentioned, cf. *spis-ete* (spelled *spiste*) 'ate' vs. *spis-te* (spelled *spiste*) 'eaten', *kis-ete* (spelled *kiste*) 'coffin', and *pås-eke* (spelled *påske*) 'Easter'.

The vowel letters *i*, *y*, *u*, are replaced by the corresponding upper case letters in cases where they symbolize mid or high-mid as opposed to high vowels, e.g. in *fUnd* 'finding, discovery, find' vs. *pund* 'pound', *lIste* 'list' vs. *pisk* 'whip', *lYgte* 'lantern' vs. *skygge* 'shadow'.

The letter *o* is replaced by (upper case) *O* in cases where it represents /o/ as opposed to the more frequent reflex /ɔ/, and in cases where it represents /ɔ:/ as opposed to the more frequent reflex /o:/, cf. *trOmme* 'drum' vs. *fromm-e* 'pious (pl.)' and *bOg* 'book' vs. *slog* 'hit'.

Boundaries between parts of a compound are marked by =, e.g. *lUft=skib*, 'airship'.

Boundaries between certain prefixes and roots are marked by +, e.g. *be+tragte-lig* 'considerable'.

Final stressed root syllables are changed in many cases where the spelling of such syllables exhibits less than two post-vocalic consonants: if the roots in question have stød-basis and stød in uninflected forms, their final letter is rewritten in geminate form, cf. e.g. *gåå* 'go, walk', *ball* 'ball'. The final consonant letter is also rewritten in geminate form if it represents an obstruent and the vowel is short, cf. *kYss* 'kiss'. If the syllable is closed and the vowel is long, their spelling is considered normal, e.g. *hospital* 'hospital'; otherwise an *h* is added to the final letter, e.g. *metalh* 'metal', *nuh* 'now'.

As can be seen from these examples, the normalized notation is neither phonemic nor morphophonemic in any reasonable sense: many-to-one correspondences between letter and phoneme are allowed; the endings *s* and *t* are not separated from preceding unstressed endings, cf. e.g. *mand-ns* 'the man's'. But, ideally at least, an autonomous phonemic transcription can be derived from it.

IV. THE TNP

The TNP consists of two parts: a data part, i.e. the lexicon, and a program part, i.e. the text processor. In the following the organization of these two parts will be outlined. It must be mentioned that what has actually been implemented is a TNP skeleton consisting of a small lexicon of test items representing the main types of entries and an *ad hoc* version of the processor. There is still much testing and experimenting to be done before a reasonably well defined system which will normalize a large part of the Danish vocabulary correctly, can be implemented. The following description is based on the well established main features of the TNP and on experiences with current *ad hoc* implementations with a limited lexicon.

No attempt will be made to describe concrete, more or less trivial technical details, since such details are dependent on current configurations of hardware and software. Suffice it to mention that the TNP is being developed and implemented under the UNIX operating system running on the PDP-11/60 computer assigned to the text-to-speech project, and that the basic code is written in the C programming language. These conditions guarantee a maximum of portability and flexibility of the basic software, so that it can be moved and adapted to various concrete configurations and applications without much trouble.

A. THE LEXICON

1. THE ENTRIES OF THE LEXICON

The lexicon is to consist of a large number of entries. Each entry is a data structure with two elements:

(1) a character string which has two functions: it permits an identification with a letter combination which actually exists as a Danish morph or is a recurrent part - a quasi-morph - of one or more Danish words; and it contains information on the normalized shape of that morph or quasi-morph. We shall see below how a letter sequence can simultaneously contain two sorts of information.

(2) a bit-pattern containing either/or information on grammatical and other characteristics of the entry. (In the current implementation the size of the bit pattern is 16 bits, which is the size of a PDP-11 machine word, but this is only a practical compromise between the amount of information needed for an entry and the hardware at hand and has nothing to do with the basic software design.)

The following subsections describe the general organization of the lexicon, some important details of the components of the entries, and some of the types of morphs and quasi-morphs the lexicon should contain.

2. THE ORGANIZATION OF THE LEXICON

The entries are arranged in several groups according to the length in letters of the item (morph or quasi-morph) they represent; thus all four letter items are in the same group, etc. Within each group the entries appear in the order that the items they represent would have in a retrograde dictionary, i.e., they are sorted alphabetically from the right end of the letter string, whereby upper case letters are evaluated as equivalent with the corresponding lower case letters.

The grouping according to length is motivated by the search algorithm of the text processor which will search for the longest possible match first.

The dictionary order of the items is motivated by the binary search algorithm that the processor invokes once the group length has been selected.

The retrograde principle is motivated by the fact that both right-to-left and left-to-right scanning technically proceeds letterwise right to left.

In addition to the entries, the lexicon contains some auxiliary information required by the processor in order to determine the beginning and the end of each length group.

3. THE COMPONENTS OF THE ENTRIES

As was mentioned above, each entry consists of a character string and a bit pattern.

The character strings are basically reverted copies of the morphs they represent; for instance, the character string part of the entry corresponding to the morph *lov* 'law' appears as "vol". The character strings in the entries contain both upper case and lower case letters. This is one of the ways in which identificational and normalizational information is condensed; for instance, the character string corresponding to *mund* 'mouth' appears in the lexicon as "dnUm", i.e. with upper case "U", but its place among the four letter words it is grouped with is the one it would have had if it had contained a lower case "u" instead. This structuring of data permits the processor to identify the entry string "dnUm" with the input string *mund* and to replace the latter with the string *mUnd*, which is the correct normalized notation of that morph when it occurs as an isolated word form.

The bit-pattern component contains a number of static one-bit flags which are set or cleared from the outset according to the behaviour of the entry.

For instance, in the current implementation each entry has a NOUN flag (set if the entry is a noun or may be part of a noun, cleared otherwise), a COMMON flag (set if the entry is (part of) a noun of the common gender), a VERB flag, and ADJECTIVE flag, a PLUR-R flag (set for a root entry if it takes *-r* in the plural, and set for the entry, whose orthographic form is *er* (since it can appear as a plural ending; the latter entry also has its VERB flag set, since it can appear as the present tense ending in verbs)). There is also a BETA flag (set for those entries which can appear as endings or quasi-endings in stødless word forms, and set for the quasi-roots which are of the regular stødless disyllabic type ending in schwa - among Danish phonologists often referred to as β-words, cf. section III E). Thus a word like *gade* 'street' is split up into a quasi-root *gad* and a quasi-ending *e* (the splitting up of words in quasi-morphs will be further commented on below).

Other important single flags are the ADD-H flag and the DOUBLE-CONSONANT flag, one of which must be set for those morphs and quasi-morphs which end in a stressed short vowel syllable and are spelled with less than two consonants after the stressed vowel: the ADD-H flag must be set in roots not taking stød when not followed by an ending, e.g. *tal* 'number, figure', and *nu* 'now'; the DOUBLE-CONSONANT flag must be set in roots that take stød when not followed by an ending, and in roots without stød-basis, e.g. *lem* 'limb', and *kys* 'kiss', cf. section III E.

The function of these flags is to supply the processor with the information needed to eventually select the correct normalized notation of an input word.

4. THE TYPES OF ENTRIES

In section III D the basic types of entries to be put into the lexicon were mentioned. Three other important features of the contents of the lexicon must be mentioned.

The idea of including in the lexicon the bulk of old mono- and disyllabic roots was motivated by the fact that, in addition to containing crucial information for the determination of stress and stød, this class of words contains the majority of irregular spellings of single segments. However, the decision that practically all such words should be accessible in the lexicon does not necessarily mean that they all have to be entered as such. These words have a property which makes it possible to reduce the size of the lexicon considerably:

A comparison of words like *mod* 'courage', *moder* 'mother', *modet* 'the courage', *moden* 'mature', *moderne* 'modern (unfortunately homographic with *moderne* 'the fashions'; but that is irrelevant to the phenomenon under consideration), *model* 'model', *modellen* 'the model', *models* 'of (a) model', *modellens* 'of the model', *modeller* 'models', *modellers* 'of models', *modellerne* 'the models', *modellernes* 'of the models', *modellere* 'to model', etc. suggests that we need only have entries for the quasi-root

mod- and for the (quasi)-suffixes and/or (quasi)-endings *-en*, *-el(l)*, *-et*, *-er*, *-erne*, *-ernes*, etc. in order to generate these forms. Since the quasi-suffixes (cf. the remarks on foreign words with stressed (Latin) suffixes in section III A) and the inflexional endings are needed anyway, there is no need to list entries such as *model*, *moden*, *moder* separately. The forms may be generated by adding a few extra flags to the bit-pattern of the entries (such as a STRESSED-SUFFIX flag to be set for quasi-roots like *mod* and for the entry *el*, so that the processor can select *modell* as the correct normalization on finding that the input word contains matches for the entries *mod* and *el* in that order, but *mod-er* as the correct normalization on finding that the input word contains matches for the entries *mod* and *er* in that order.

Since potential endings like *er*, *erne*, *ens*, *et*, *ets*, etc. must be identified anyway, there is a problem with roots ending in the corresponding letter sequences: In a word like *lanterne* the *erne* sequence is identical to the final sequence in *planterne* 'the plants' where *erne* represents the definite plural ending of a noun. Since *erne* must appear in the lexicon anyway, there is no need to list *lanterne* in its entirety; indeed it would be erroneous to do so, for thanks to the maximalistic way of searching, cf. section III D and IV B, *lanterne* would be found before *erne*, and this would prohibit *planterne* from being correctly identified, unless, of course, still longer word forms, containing *lant* as a substring were to be listed too, i.e. inflected forms like *planterne*, *planternes*, *slanterne*, *slanternes*, etc., which is the very thing to be avoided, since stuffing the lexicon with numerous entries containing identical information would actually be to abandon the quasi-morph policy. What this means is that lexical entries which cannot appear as endings or other important right-end material are permissible as entries only if they do not contain a rightmost letter sequence which is (orthographically) identical with such an item. Thus a word like *stjerne* 'star' must be split up in the lexicon as a quasi-root *stj* and the quasi-ending *erne* which is needed anyway.

On the other hand, there are cases of homography between certain suffixes and inflected forms of other suffixes or endings which necessitate the inclusion in the lexicon of certain quasi-roots which do not belong to the bulk of old words. Thus, since word final *ens* can be (1) a definite genitive of a common gender noun, as in *gadens* 'of the street', (2) a homophonous quasi-ending as in *Assens* (the name of a town), (3) the stressed quasi-suffix *ens* (with a short vowel) as in *tendens* 'tendency', or (4) the genitive of a root ending in stressed /e:ʔn/ as in *arsens* 'of arsene', the two latter types must have entries for the corresponding quasi-roots (the *tend* of *tendens* and *ars* of *arsen*), since it seems safest to reserve the normalization *-ns* (definite singular genitive of a finally stressed noun) for foreign roots not found in the lexicon, so that e.g. a word like *mikadoens* 'the Mikado's' will be normalized as *mikado-ns*, consistent with its phonemic structure

/mi'ka: do:ʔəns/, and not as *mikadoen+s* (whitch the rule system would transform into /mikado'e:ʔns/), or as *mikadoens* (which the rule system would transform into /mikado'ɛnʔs/).

B. THE PROCESSOR

1. THE INPUT AND SCANNING ROUTINES

The processor operates in the following fashion:

Text is input one sentence at a time, in which process upper case letters are converted to lower case. For each word the length in letters, the number of syllables, and the position in the sentence is memorized.

Each word is subject to one or both of the following routines:

(1) The right-to-left scanning routine: The rightmost letters of the word are compared to lexical entries in the following fashion: if the word is no shorter than the longest group of entries in the lexicon, a match is looked for in that group, otherwise a match is looked for in the group whose length corresponds to the length of the word. If a match is found, the search is stopped, and the address of the matching entry is memorized. If no match is found in that group, a match is looked for in the group with entries shorter by one, and so on, until either a match is found or the whole lexicon has been scanned (thanks to the structuring of the lexicon and the design of the search algorithm, this does not, of course, mean that every entry in the lexicon has to be compared with the input). If no match is found in any of the groups, the right-end scanning is stopped, and program control transferred to the left-to-right scanning routine, cf. below. If a match is found, the whole right-to-left scanning routine is repeated, starting from the position in the word to the right of which the match was found. This routine is repeated until either the left end of the word is reached or no match is found. The left end of the word being reached thus means that the whole word was matched morphwise or quasi-morphwise, and program control is transferred to the interpreter routine, cf. below.

A non-match being recognized before the left end of the word is reached means that only the rightmost portion of the word was matched, viz. the portion from the position reached after the last match found to the right end of the word. In this case program control is transferred to the

(2) left-to-right scanning routine, which is (conceptually, at least) a mirror image of the right-to-left scanning routine, i.e. it scans the word from left to right up to, but not including the position reached by the right-to-left scanning routine, and memorizes the addresses of any matching entries.

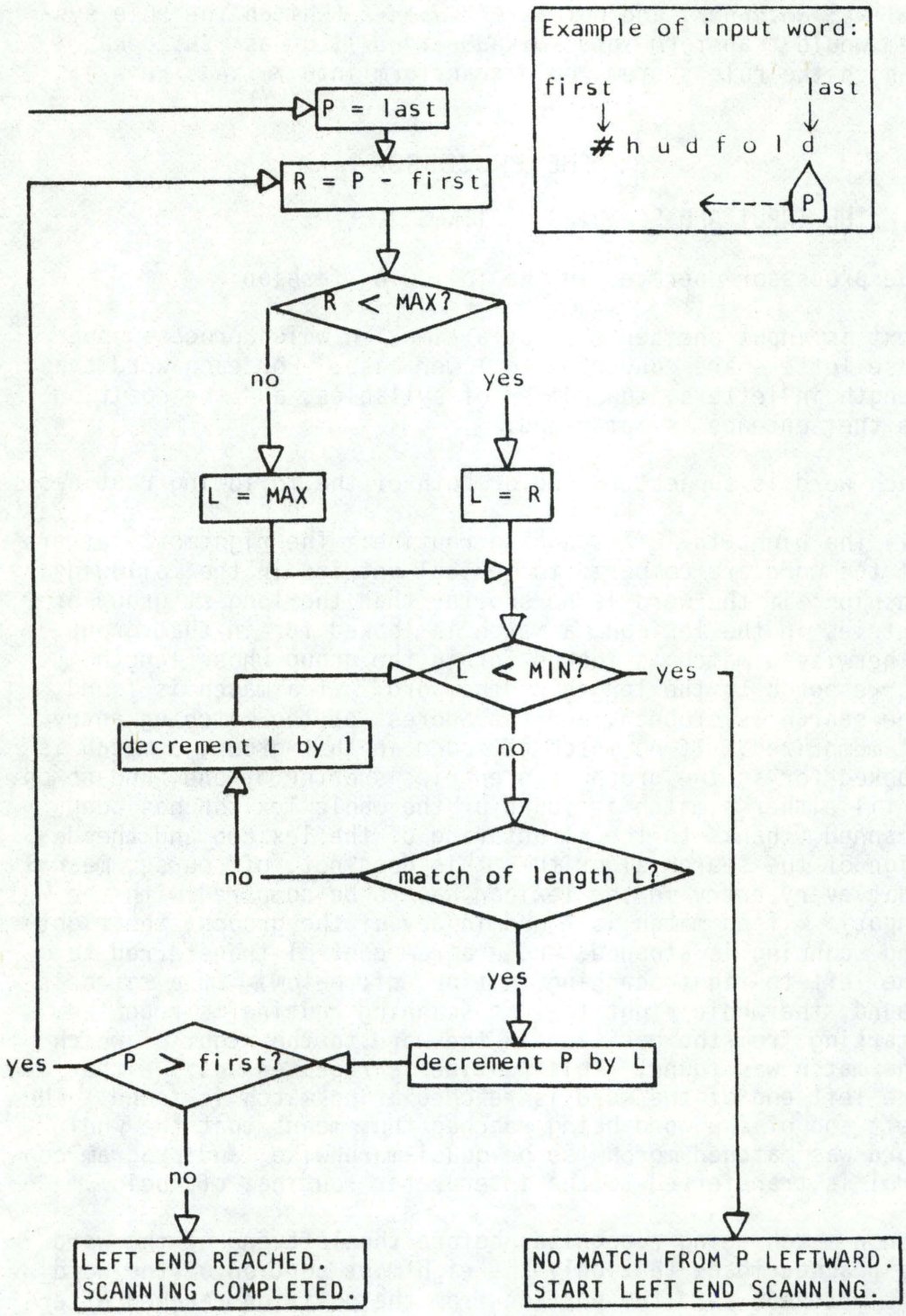

Figure 1

Flow chart of the right-to-left scanning routine. P *= current position in the word;* last *= rightmost position in the word;* first *= leftmost position in the word - 1;* R *= length of the current left remainder of the word;* L *= current length of match to be looked for;* MAX *= length of longest entries in the lexicon;* MIN *= length of shortest entries in the lexicon.*

What all this means is that the processor carves its way, as it were, from both ends of the word until either the word is matched in its entirety, or a left-end, right-end, or - more frequently - medial residue is left unidentified.

Figure 1 is a flow chart of the right-to-left scanning routine.

2. THE INTERPRETER ROUTINE

This routine is the intelligent part of the processor. It does the important job of combining and interpreting the information handed over from the input and scanning routines, and selects a normalization procedure according to its interpretation. The information at its disposal is (1) its knowledge of found matches and access to their entries (which, as we have seen, consist of two components, each with its own type of information), (2) its knowledge of unidentified residues, and (3) its knowledge of the position in the word of morphs, quasi-morphs, and the residue (if any).

The following example will give the reader an idea of the basic philosophy underlying the interpreter routine:

In a word like *afreagerer* 'abreacts, works off' the scanning routines will first identify the two rightmost letters as the lexical entry *er*. Then the next two letters are recognized as the same entry *er*. Then the next four letters are recognized as the entry *reag* (although this entry is neither a genuine root (or quasi-root) nor an affix, it should appear in the lexicon because of its combinability with the stressed suffix *ens*, cf. *reagens* 'reagent'; see further section IV A 4). Finally, the remaining two letters are recognized as the entry *aF* (see further below). In other words, the interpreter routine is presented with one of the easier cases, viz. that of the whole word being matched. It now begins to weigh the evidence, as it were, and its way of doing this can be described, somewhat metaphorically, as follows:

The rightmost item, *er*, considered in isolation, reduces the possible analyses of the word to be one of the following:

(a) a word in which *er* belongs to the root and is phonologically /ɛr/ (occurs probably only in two monosyllables, viz. *er* 'is', and the pronoun *jer* 'you (plur.)';

(b) one of a few monosyllables ending in /e:ʔr/, cf. *sner* 'snows' (where *e* belongs to the root and *r* is the present ending), and *fjer* 'feather' (where *er* belongs to the root);

(c) a polysyllable where *er* is a stem final syllable, in which case *er* is phonologically /e:ʔr/, cf. *klaver* 'piano';

(d) a polysyllable where *er* is unstressed and phonologically is /ər/, and where the preceding syllable takes stress and stød, cf. *mager* 'lean' (where *er* belongs to the root), *bøger*

'books' (where *er* is a plural ending), *kommer* 'comes' (where *er* is the present ending of a verb as in the actual example);

(e) the same as d), except that the preceding syllable does not take stød, cf. *bager(1)* 'baker' (where *er* is a nominalizing suffix) and *bager(2)* 'bakes' (where *er* is the present tense ending);

(f) the same as e), except that the preceding syllable is unstressed, cf. *magiker* 'magician' (where *er* is a nominalizing suffix), *hoveder* 'heads (noun plur.)' (where *er* is a plural ending).

Consider now how the identification of the next item from the right will reduce the possibilities: At this stage, thanks to the combined information of the flags of the two rightmost items - which happen to be identical - and of the positions of these items, possibilities a, b, and e can be immediately discarded, i.e. the remaining phonological representations of the last two syllables could be (i) /e're:ʔr/ as in *gerer dig!* 'behave yourself!', (ii) /ərɛ:ʔr/ as in *moderer!* 'moderate! (imp.)', (iii) /ərər/ (which does not occur in Danish words), or (iv) /e:ʔrər/ as in the actual example.

The information obtainable from the next item from the right, *reag*, which is known (from the bit-pattern component of its entry) to be a verbal root of the type which is right-compatible with stressed Latin suffixes like *ens* (forming deverbative nouns), and *er* (forming verbal stems with such roots), will - when combined with the information accumulated so far - discard possibilities i, ii, and iii, and the processor can now be certain that it is dealing with a simple word or a compound the last part of which is the present tense of the verb *reagere*. The remaining, leftmost item *aF* is known to be a word which can appear as the first part of a compound verb like *afreagere*. As can be seen, the leftmost item appears in the lexicon in the form *aF*, where the *F* is a normalized segment equivalent with *v* when not word final. The procedure selected by the interpreter routine will now produce the correct normalized notation *aF=reager-r*. Incidentally, thanks to the maximalistic search algorithm, if the word had been *afficerer*, the third item from the right would have been recognized as *affic*, this item being listed in the lexicon because it belongs to a handful of quasi-morphs in which initial orthographic *af* is not the (genuine) word *af*, cf. Molbæk Hansen (1982, p. 132).

It is an important property of the interpreter routine that it is, at least in principle, designed to act like an average Danish reader. In the case of *afreagerer* it happened to recognize *reag*, just as a Danish reader would - *ceteris paribus* - recognize that part of the word as something which exists and which has, among other properties, the ability to fit in between the prefix /av/ and the /e:ʔrər/ part which would indicate that he was dealing with a finite verbal form. But the interpreter routine should - if properly designed - produce the same normalized notation of the word even if it had not found

reag in the lexicon: suppose we did not know the *-reag-* (or, for the benefit of linguists and Latinists, *-re-ag-*) part, what would our guess have been? It would definitely have been that we were dealing with some foreign verb which we happen not to be familiar with, but which is of the *-ere* type, in this case prefixed with /av/; this is what any Dane would do if he were presented with a nonsense form like *afpumonerer*. (There would have been the technical difference that the right-to-left scanning routine, not finding a medial letter sequence (like *pumon*) in the lexicon, would have left it to the left-to-right scanning routine to identify *af*, but that is of minor importance here.)

Another illustration of the work of the interpreter routine is a comparison between its treatment of (1) certain forms of *damp* 'steam', (2) *drabant* 'halberdier, satellite', and (3) *urbant* 'urbane (adj. neut. or adverb)':

damp would be recognized immediately and, appearing without an ending, it would be normalized as *damp*. If it had been the definite form *dampen* 'the steam', the information that it is a finally stressed, common gender noun (its BETA flag is cleared, and its COMMON flag is set) combined with the entry *en*, which can appear as the definite ending for such nouns, would have yielded the normalized notation *damp-n*. If it had been *dampet* 'steamed', the information (thanks to another flag) that it is compatible with the entry *et*, which can appear as the past participle suffix, would have yielded *damp-et*; (NB: not *damp-t*, since the entry *damp* has its COMMON flag set so as to prohibit it from combining with the (normalized) ending *-t*, which belongs with finally stressed neuter nouns).

In *drabant* the *ant* part would first be recognized as either representing the adjectivizing or nominalizing suffix *ant* or the stressed, stem final syllable of a regular adjective in *an* with the neuter (or adverbalizing) ending *+t*. Next *drab* would be recognized as being compatible with certain suffixes like *ant*, and that would settle the matter, yielding the normalized notation *drabant*. Incidentally, this case illustrates the entry-saving policy mentioned in section IV A 4: the entry *drab* is, of course, needed as the neuter noun *drab* 'homicide'; thus, in addition to having its STRESSED-SUFFIX flag set - so as to make it compatible with the lexicon-suffix *ant* as in the case under consideration - this entry should have its NOUN flag set, and its COMMON flag cleared. Notice that, if a nonsense form like **drabens* were to appear as input, it would be normalized as *drabens* (which the rule system would transform into /dra'bɛn?s/, cf. section IV A 4) if the following conditions were met: (a) *drab* was found in the lexicon, (b) *drab* had its STRESSED-SUFFIX flag set, and (c) the entry *ens* had its STRESSED-SUFFIX flag set like the entry *ant*. The entry saving is possible in such cases because, typically, only one of several possible combinations of morphs or quasi-morphs exists. Thus a word like *fodens* 'of the foot' would be normalized correctly as *fod-ns*, because it would have its COMMON flag set (like *ens* would), and this would be decisive even if it

had its STRESSED-SUFFIX flag set too; but the latter flag would be responsible for the normalization *fodant* - quite similar to *drabant* - if the non-existent form **fodant* were to appear as input, cf. also below.

In *urbant*, finally, *ant* would be recognized as in *drabant*; then, supposing that *urb* is not found in the lexicon, the next item from the right would be recognized as the entry *b*, which has to be in the lexicon because it is the initial consonant in a word like *ben* 'leg' (where *en* represents a potential ending, cf. IV A 4). Finally, the remaining part of the word would be recognized as the entry *ur*. In this case the interpreter routine - if properly designed - ought to discard the possibilities *ur=b=ant* (since single consonants cannot be parts of a compound), and *ur=bant* (since (a) *bant* was not found in the lexicon, and (b) *ant* is probably either the monomorphemic stressed suffix *ant* or the bimorphemic structure *an+t*, cf. above). Of the remaining possibilities *urbant* (i.e. an unidentified root *urb* suffixed with *ant*) should be discarded since that suffix is supposed to have its (quasi)-roots - like *drab* and *fod*, cf. above - in the lexicon, and the default case *urban+t* should be chosen.

These examples ought to illustrate the most important properties of the interpreter routine and, in particular, of the combined function of that routine and the special structuring of lexical entries outlined in section IV A.

V. CAPABILITIES, DEFICIENCIES, AND PERSPECTIVES OF THE TNP

An implementation of the TNP with a comprehensive lexicon and with an optimal choice and distribution of bit-patterns ought to successfully normalize the following types of isolated words, provided that they have no homographs:

(1) Most simple words belonging to the genuine vocabulary, and all inflected forms of such words, cf. *mand* 'man', *lyve* 'lie', etc.

(2) The majority of those high frequency compounds which consist of material belonging to the genuine vocabulary, and all inflected forms of such words, cf. *søpindsvin* 'sea urchin', *springkniv* 'flick knife', etc.

(3) Most simplex foreign words - possibly with affixes - and inflected forms of such words, cf. *kalamitet* 'calamity', *vegetar* 'vegetarian', etc.

(4) Many compounds the initial and/or final parts of which belong to the genuine vocabulary, and inflected forms of such words, cf. *kvindeemancipation* 'emancipation of women', *pensionsalder* 'pensionable age', *efterrationalisere* 'rationalize (afterwards)', etc.

The main shortcomings of the TNP are due either to the maximalistic way of scanning the lexicon, or to the fact that the processor loses track of the morph boundaries in certain cases, notably in long compounds consisting exclusively of foreign material.

As Allen (1976) has rightly pointed out, the maximalistic principle is insufficient in cases where the morph composition is ambiguous as e.g. in *solur*; this form can be interpreted as either *so=lur*, literally: 'sow-lure', 'nap taken by a sow' (!) or *sol=ur* 'sundial', literally: 'sun-watch'. Of these interpretations the latter is, of course, superior, mainly for pragmatic reasons; but there is no morphological, syntactic, or phonotactic reason for not preferring the former interpretation. Because of the maximalistic, right-end-first policy of the processor, *so=lur* will be the interpretation chosen; but it is easy to adduce cases where the actual choice will, by chance, be pragmatically superior to the alternative, cf. e.g. *sømand* which could be either *sø=mand* 'sailor', literally: 'sea-man', or *søm=and*, literally: 'nail-duck', of which the former is a more or less lexicalized, common compound, whereas the latter is pragmatically dubious, to say the least, but nevertheless entirely acceptable as a Danish compound (note that the pronuncations would be different in the two cases).

As Allen has pointed out, an algorithm finding both (all) the combinations could in some cases evaluate them and "pick a winner". For an English example - *scarcity* - see Allen (1976 p. 436). It is unclear to me whether the extra software overhead needed to take care of such situations would be worth while for Danish. It is clear, however, that this difficulty can be easily overcome in applicational versions of a text-to-speech system by including in the lexicon the relatively few problematic cases any selected corpus of words (e.g. a frequency dictionary) would contain.

The other main deficiency is the fact that the processor will lose track of the morph composition of long foreign compounds like *basilarmembran* 'basilar membrane' (note that in this case, as in many others of the type in question, the word is matched by a noun phrase in English). This is, of course, serious in texts with a high frequency of such words. But such compounds are typically highly specialized words occurring in special (technical) texts for which special extensions of the core lexicon could be made.

So far, there remains two major categories of problems, viz. those of homography and of sentence phonological phenomena closely connected with syntactic structure. Note, however, that the policy of supplying each lexical entry with a bit pattern containing information on such lexical properties as word class etc. is compatible, almost *par excellence*, with overlaying the TNP with routines for exploiting such information. This perspective has been one of the main motivations for designing a TNP with properties as outlined in this paper.

The current work is concentrated on preparing facilities for testing the output of the TNP against well defined lexical material such as, e.g., the corpus of a frequency dictionary (Maegaard and Ruus (1979)). Although such tests will probably disclose numerous minor flaws and redundancies in the current implementation - in particular in the choice of grammatical and other flags in the bit-pattern component of lexical entries - the general strategy is believed to be sound.

VI. NOTE

1. In Molbæk Hansen (1982) the abbreviation XCO (exception removing component) designates what roughly corresponds to the TNP.

ACKNOWLEDGEMENTS

I am indebted to Hans Basbøll, Jørgen Rischel, and Nina Thorsen for valuable suggestions.

REFERENCES

Allen, J. 1976: "Synthesis of speech from unrestricted text", *Proceedings of the IEEE, vol. 64, no. 4*, p. 433-442

Allen, J. 1981: "Linguistic-based algorithms offer practical text-to-speech systems", *Speech Technology 1,1*, p. 12-16

Basbøll, H. 1972: "Some remarks concerning the stød in a generative grammar of Danish", *Derivational Processes (Ferenc Kiefer, ed.) (Stockholm 1972)*, p. 5-30

Carlson, R. and Granström, B. 1975: "A text-to-speech system based on a phonetically oriented programming language", *Speech Transm. Lab., Quart. Prog. and Status Rep. 1/1975*, p. 17-26

Dewey, G. 1971: *English Spelling: Roadblock to Reading*. Columbia University, New York: Teachers College Press

Holmboe, H. 1978: *Dansk Retrogradordbog* (Copenhagen, Akademisk forlag)

Holtse, P. 1982: "Speech synthesis at the Institute of Phonetics", *Ann. Rep. Inst. Phon. Univ. Cph. 16*, p. 117-126

Lesmo, L., Mezzalama, M. and Torasso, P. 1978: "A text-to-speech translation system for Italian", *Int. J. Man-Machine Studies 10*, p. 569-591

Maegaard, B. and Ruus, H. 1979: *HYPPIGHEDSLISTEN DANwORD L1A*, Inst. f. anv. og mat. lingv. og Inst. f. nord. fil. University of Copenhagen

Maggs, P. et Trescases, P. 1980: "De l'écrit à l'oral. Un programme sur micro-ordinateur pour machine à parler à l'usage des aveugles francophones", *Le français moderne, revue de linguistique française no. 3*, p. 225-245

Molbæk Hansen, P. 1982: "The construction of a grapheme-to-phone algorithm for Danish", *Ann. Rep. Inst. Phon. Univ. Cph. 16*, p. 127-136

Rischel, J. 1970: "Morpheme stress in Danish", *Ann. Rep. Inst. Phon. Univ. Cph. 4*, p. 111-144

Sherwood, B. A. 1978: "Fast text-to-speech algorithms for Esperanto, Spanish, Italian, Russian, and English", *Int. J. Man-Machine Studies 10*, p. 669-692.