# A PREPROCESSOR ALGORITHM FOR THE CSTR TEXT TO SPEECH SYSTEM

M.J. McAllister, J. Laver, J.M. McAllister

Centre for Speech Technology Research
University of Edinburgh

## INTRODUCTION

In any Text-to-Speech (TTS) system there will be textual strings presented as input which fail to conform to the typographical norm on which the design of the system's dictionaries and rule sets has been based. For example, in this article the majority of the space-bounded strings (conventionally termed 'orthographic words') are composed entirely of lower case letters; a non-trivial minority, however, contain other characters.

The signalling of sentence-initial words by the capitalisation of their first letter and the addition of a full stop to sentence-final words are ubiquitous examples of such anomalies. Examples of less commonplace non-standard strings are dates, times, acronyms, abbreviations and mathematical formulae.

The interpretation of these and any other textual anomalies depends on their accurate detection and classification. The remainder of this paper comprises a brief description of an algorithm design to accomplish this identification process and the conceptual framework which underlies it. This algorithm forms part of a text preprocessor module in the CSTR TTS system. The function of this preprocessor is to correct anomalous items to orthographic forms in lower case letters, so that the other modules of the TTS system can convert them to phonemic form in a routine way.

## THE RECOGNITION AND CLASSIFICATION OF ANOMALOUS STRINGS

In the following discussion, a distinction is made between the alphanumeric set (comprising three subsets: the lower case letters, the upper case letters and the digits) and the non-alphanumeric set, which will hereinafter be termed the 'punctuation set'. This comprises any characters available on a QWERTY keyboard other than those listed above, and includes the following:

` ~ ! @ # $ % ^ & * ( ) _ - + = { } [ ] : ; " ' | > < , . ? /

The term 'character' refers to any symbol in the text which is discriminably different from all other characters that the computer can print; the term 'string' refers to any sequence of characters bounded by space, tab, or newline characters.

It is necessary to establish a set of Anomaly Subclasses, each defined by the typographical form of the anomalies they contain. This leads to systematic organisation of anomalies into dictionaries based entirely on typographic form. This lexical organisation allows the preprocessor to search for pre-stored interpretations of forms without recourse to semantic or pragmatic knowledge. This typographic delineation of anomalies also assists the systematic interpretation of new forms, without a prescribed pronunciation.

The set of anomaly types which has been selected characterises all possible combinations of the lower case elements a-z, the upper-case elements A-Z and the digits 0-9 (although other symbols may be included in the string), and is as follows:

| | ANOMALY TYPE | EXAMPLE |
|---|---|---|
| 1. | all lower case | e.g., i.e. |
| 2. | all upper case | NUT, MIT |
| 3. | lower and upper case | PhD, Dr |
| 4. | digits | 1986, 100% |
| 5. | alphanumeric mixtures (1) | 19th, 10p |

## STRING INTERPRETATION

In other modules of the CSTR Text-to-Speech system, the problem of homographs arises: given the two possible pronunciations of the orthographic string 'lead' (/l ii d/ and /l e d/), the system must assign the appropriate pronunciation on the basis of word class information. An analogous problem exists for the Preprocessor module; the string 1066 may represent a either a date, or an integer, appropriately transcribed(2)

/"t e n "s i k s t i "s i k s/

and

/"w uh n "th au z @ n d @ n d "s i k s t i "s i k s/.

In the case just cited, the resolution of the ambiguity would depend on semantic and pragmatic cues as yet not fully researched and certainly not available to any existing Text-to-Speech system; but for a large number of instances, disambiguation can be assisted by the presence of punctuation symbols in the string. The all-digit string '1945', for example, has numerous interpretations, some of which are listed below:

| INTERPRETATION | FORM(S) | POSSIBLE ORTHOGRAPHIC FORM |
|---|---|---|
| year | 1945 | nineteen-forty-five |
| date | 1.9.45 | the first of September nineteen-forty-five |
| time | 19:45 19.45 | nineteen-forty-five |
| integer | 1,945 1945 | one thousand.nine hundred and forty five |
| ratio | 19:45 | nineteen to forty-five |
| decimal number | 1.945 | one point nine four five |
| currency | $19.45 | nineteen dollars and forty-five cents |
| percentage | 19.45% | nineteen point four five percent |

Several issues arising from this data deserve consideration. Firstly, note that in the last three instances the punctuation symbol must be assigned an explicit pronunciation; however, once the appropriate Anomaly Subclass has been identified, the assignment of pronunciation can proceed naturally. For the moment, only the potential for disambiguation of such symbols will be considered. Secondly, alternative full orthographic forms may exist for some of the strings: alternative full forms of the string '$19.45' are 'nineteen dollars and forty-five cents' and 'nineteen forty-five'; though several forms could be listed in the TTS system dictionary, it is not clear what criteria might be invoked to select the appropriate one. Thirdly, some symbols can do no more

than narrow down the range of possible interpretations for a given ambiguous string: the string 19.45 may be a decimal number or a time, but it cannot be interpreted as a date, year, integer, currency or percentage.

## PUNCTUATION DISAMBIGUATION

Paradoxically, although the punctuation set may serve to disambiguate alphanumeric strings, members of it may themselves be ambiguous with respect, firstly, to their function in a string (as discussed above) and secondly, to their domain of operation. This domain can be either the string in which they appear, or larger linguistic units, such as the whole sentence. For example, the sequence:

'She rang the D.H.S.S. 20 minutes later...'

may be parsed as either a single sentence ('She rang the DHSS twenty minutes later') or one sentence

'She rang the DHSS'

followed by the sentence-initial fragment

'Twenty minutes later...'.

This distinction has important intonational consequences. The source of the ambiguity lies in the dual role of the symbol '.', which may be interpreted as a sentence-terminating character (period) or as an abbreviatory device. In the former instance the domain of influence of the symbol is the whole sentence; in the latter, the string in which it appears. In the following discussion, uses of a character which extend only to the string in which it appears (e.g. the apostrophe and percent sign in: man's, 28%) are termed textual; uses which affect the broader linguistic context (e.g. the round brackets and single quotation marks in: "His name is Spencer - as in 'Marks and Spencer'") are termed metatextual.
Punctuation characters which can occur metatextually are:

? ! . " ' ' ( ) { } [ ] < > / / - , ; : -

Punctuation characters which can occur textually are:

$ # + % ! = . " ' ( ) < > / ^ * - , ; : -

The first step in approaching this problem has been to identify those characters which belong to both classes and to examine the environments in which they occur. The legitimate inclusion of particular textual characters in strings of each of the Anomaly subclasses has been examined in terms of their relative position in the string. The results of this analysis are presented in Table 1. Table 2 classifies the metatextual characters according to string position. In Table 3 the metatextual punctuation characters are shown, classified according to function.

| Table 1: Textual Punctuation Set | | | |
|---|---|---|---|
| Anomaly Subclass | Environment initial | medial | final |
| all lower case | (none) | - / . ' | ' |
| all upper case | (none) | - / . ' | ' |
| lower and upper case | (none) | - / . ' | ' |
| digits | $ # ' . + - | . : , / + - * / = < > ^ | % " ' ! |
| alphanumeric mixtures | ' . + - | . : , / + - * / = < > ^ | (none) |

| Table 2: Environments of the Metatextual Punctuation Set | | | |
|---|---|---|---|
| | Environment string initial | string final | stand alone |
| char: | " ' ( { [ < / | ? . ! " ' ) } ] / , ; : | - & |
| NOTE: no characters occur string-medially | | | |

| Table 3: Functions of the Meta Textual Punctuation Set | |
|---|---|
| Sentence terminators | ? ! . |
| Quote delimiters | " " ' ' , , |
| Text delimiters | ( ) { } [ ] < > / / - - |
| Phrasal structure markers | , : ; - |
| Word surrogates | & |

NOTE:

members of the second and third category must appear in pairs, as specified; the first member of the pair is string initial and the second string final.

Some interesting generalisations arise from these classifications:

(a) no textual punctuation character which occurs string-medially also occurs as a metatextual character.

(b) the quote and text delimiters are the only metatextual characters which occur word-initially, and these always occur in pairs; thus the occurrence of a string-initial metatextual character implies that its string-final equivalent will occur.

(c) the majority of characters which occur both as textual and metatextual elements are found string-finally.

The characters which are ambiguous are therefore:

! . ' "

The environments in which these four elements occur should now be considered. The symbol . occurs metatextually as a sentence terminator, and textually to indicate abbreviation (M.I.T., m.p.h., etc.). The symbol ! occurs metatextually as a sentence terminator, and textually to represent the mathematical concept 'factorial': since we are not attempting to consider such specialised usages here, perhaps this symbol can be dropped from the ambiguous set. The single quote symbol ' delimits quoted material (finally) in its

metatextual function; textually, in a string final position, it is used as a marker of possession (e.g. the bosses' cars) and to indicate the measurement 'foot' or 'feet' (e.g. 12' for 'twelve feet'). The double quote marker " is similarly used metatextually as a quote delimiter and textually as a marker of 'inches': 6" for 'six inches'. String initially, the single quote symbol is used for year abbreviations, (e.g. Expo '86) and as such must always be followed by two digits.

## IMPLEMENTATION

A program has been written in PROLOG to perform the anomaly identification process. The program works in three sections.

Firstly, a sentence is read in as a list of 'orthographic islands': an orthographic island is any sequence of printable characters bounded on each side by a space, tab or newline character. The separation into orthographic islands includes a strategy for dealing with word hyphenation across line breaks. To identify the end of the sentence, a sentence-terminating punctuation character must be found in a meta-textual context. Currently, only local constraints are being used to perform the necessary disambiguation.

Secondly, each orthographic island in the list is tested for the presence of meta-textual punctuation at its boundaries. This section includes an explicit listing of validity constraints, (e.g. hyphen can be textual at the front of a numerical string as a minus sign) which enables a fast and accessible adjustment of such constraints for experimentation. Any meta-textual characters detected are then treated as separate strings.

Finally, all strings (both orthodox and anomalous) other than single-character meta-textual ones are classified by composition. For example, a string containing only lowercase letters is labelled differently from a string containing both upper and lower case letters, or only containing numbers or containing a mixture of alpha/numeric characters. This classification also uses an explicit listing of validity constraints, this time for punctuation characters which may appear textually in a string. This algorithm has been applied to corpora from several sources to test the generality of the framework devised. During this development phase, strings whose composition excludes them from the classes above are being analysed to refine the two sets of constraints used.

The text files on which the preprocessor program has been tested have included some prepared by proprietary text processing packages in which the use of control and escape code sequences is rife. None of these non-standard data have disrupted the algorithm encoded.

PROLOG has been used to develop this algorithm because of its power and flexibility as a rule-based environment; it is, however, an interpreted language and its execution is typically between one and two orders of magnitude slower than compiled 'C' code. The preprocessor algorithm so far developed is currently analysing text data at a rate of about 4,000 characters per minute.

## A TEST CASE

In a test using a technical report text of 80,000 characters, made up of about 12,200 strings, around 2,500 anomalies were detected and dealt with. In addition, about 2,150 punctuation characters were stripped from strings where their position suggested a metatextual role. The anomalies were divided according to composition into alphabetic, numeric, alphanumeric and other strings. In this case all punctuation characters were allowed anywhere in the strings.

Just over 88% of the anomalies were alphabetic strings including fully capitalised words in titles (6%), abbreviations & acronyms (3.6%), compounds joined by hyphens or slashes (8.5%) and initial capitalised words (67.6%). A single anomaly type accounted for 1% of the total number, i.e. the inclusion of an opening or closing bracket string medially. On closer inspection this occurrence produces the first adjustment of our punctuation disambiguation framework. The representation of an optional plural ending on nouns (e.g. 'type(s)') allows the textual use of both left and right round brackets.

Among the numeric strings (total 9.24%), integers other than years accounted for 5.9% of the total number of anomalies, years: 1.8%, number ranges (e.g. '18-30'): 0.68%, signed integers: 0.52%, and each of the other numerical conventions including ordinals, decimals, percentages and text section numbers each accounted for 0.1% or less. The numerical ranges point to another modification of our punctuation disambiguation framework, i.e. the valid inclusion of a hyphen string medially between two numbers.

All of the alphanumeric anomalies highlighted the variety of conventions that will have to be interpreted by a full Text-to-Speech System; these included: number-unit combinations ('300Hz'): 0.72%, ordinals ('27th'), number-word combinations ('3-dimensional'), explicit numberings ('No.10'), number sequences ('1,2...99') and page numberings ('p.88' or 'pp.88-90') each accounting for less than 0.1% of the total number of anomalies.

The strings not fitting any of these categories are by definition composed entirely of punctuation characters, and these accounted for 1.2% of the total number of anomalies. These included 1 typing error, 8 instances of mathematical operators, 20 instances of text-processing software control characters and a new metatextual string '...'. This is also encountered in number sequences as noted above where an appropriate pronunciation might be 'and up to'. In number sequences it performs textually, binding the surrounding numbers into a single phrase, but in running text it performs metatextually, denoting a continuation especially in quotations. In its metatextual role it has no direct transcription but may have a prosodic effect.

## CONCLUSION

The role of a preprocessor in an automatic TTS system is complex. In the long term perspective, it is envisaged that the preprocessor will be capable of handling all classes of anomaly that can be described systematically. As a first step in this process an algorithm which recognises and classifies anomalies encountered in text has been implemented.

## NOTES

(1) On the basis of some preliminary analysis, it was decided that a subdivision of this class into 'upper case and digits' and 'lower case and digits' would not confer any additional benefits.

(2) The phonetic transcription system used throughout is the Edinburgh University Machine Readable Phonetic Alphabet.

## BIBLIOGRAPHY

McALLISTER, J.M., McALLISTER, M.J. & LAVER, J.(1986) "A preprocessor algorithm for the CSTR Text-to-Speech System", Proceedings of the Institute of Acoustics 1986 Autumn Conference: Speech and Hearing. In Press.