Concept-to-Speech Synthesis by Phonological Structure Matching

By P A Taylor

Centre for Speech Technology Research, University of Edinburgh

This paper presents a new way of generating synthetic speech waveforms from a linguistic description. The algorithm is presented as a proposed solution to the speech generation problem in a concept-to-speech system. Off-line, a database of recorded speech is annotated so as to produce a phonological tree for each sentence in that database. Synthesis is performed by generating a phonological tree called the target tree, and searching the database of trees to find nodes which are the same in both trees. A search strategy using target and concatenation costs is then used to find the optimal sequence of units for the target sentence. This paper explains this algorithm, compares it to existing algorithms and concludes with a discussion of future directions.

Keywords: speech synthesis, phonology, unit selection

1. Introduction

The term *text-to-speech* (TTS) synthesis is used to describe the process of converting given raw text into synthetic speech. *Concept-to-speech* (CTS) is a term often used for speech synthesis where the input is not text, but rather a machine generated message. We can think of a TTS system as comprising two main components: text analysis and speech generation. The text analysis component has to resolve the ambiguities inherent in written text and produce a clean linguistic representation of the sentence to be spoken, e.g. appropriate word stress. In CTS, the situation is very different. There is no prior input text as such, rather a natural language generation (NLG) system generates some text from scratch. In one of the domains used for this work (see section 3), the task is a intelligent museum guide in which descriptions of museum exhibits are generated dynamically according to the interests of the visitor, taking into account the context of what the visitor has already seen. An utterance is generated by the NLG system in response to a query (e.g. "tell me more about object X") by using a database of exhibit information. The output of the NLG system is then fed into the synthesizer which converts this into speech.

Thus in the CTS case, there is no text ambiguity: the generator can annotate the text it it produces with the information needed to guide synthesis. For instance when the word "project" is used, the system knows whether it is a noun or a verb whereas a TTS system has to guess this. Information which is virtually impossible to resolve in TTS can be used quite easily in CTS; as well as word stress, ambiguities arise in many other areas including pronunciation, phrasing and prosody. In general CTS leads to a much richer more reliable linguistic input to the synthesizer. Figure 1 shows a how TTS and CTS systems have different input components, but can use the same speech generation component. NLG systems vary in sophistication, ranging from systems which use simple templates to systems which

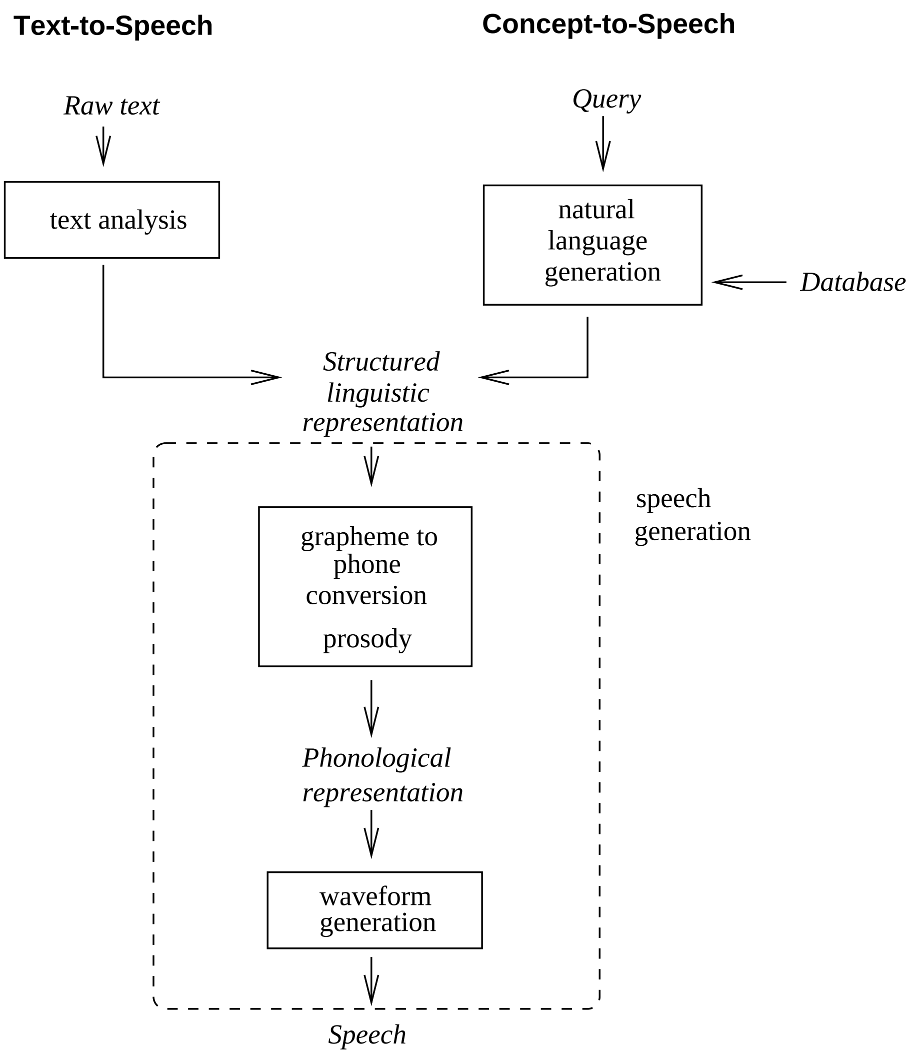
Text-to-Speech
Concept-to-Speech
Raw text
Query
text analysis
natural language generation
Database
Structured linguistic representation
grapheme to phone conversion
prosody
speech generation
Phonological representation
waveform generation
Speech

Figure 1. Text to speech and Concept to speech

generate text using sophisticated linguistic models. Depending on the complexity of the domain, many different approaches are used.

In speech generation, on the other hand, the choice is between two quite distinct approaches. In slot and filler synthesis systems, a carrier phrase such as “the train at platform X is now departing for Y” has its slots X and Y replaced by a set of pre-recorded words. While the number of possible messages may be large, it is finite. This sort of system is often contrasted with “genuine” speech synthesis techniques such as diphone synthesis in which arbitrary messages of any sort can be synthesized. The advantages of each system are obvious - because the slot and filler system uses long carrier phrases with appropriate prosody and naturally recorded words it can often sound excellent - however it can only speak the range of messages for which it has recordings. In contrast, while diphone synthesis is capable of generating the speech for any input, its quality is considerably worse. The goal of the research described here has been to bridge the gap between these two different synthesis approaches and build build a synthesis solution that combines the high quality of the slot and filler approach with the flexibility of diphone synthesis.

2. Domain-Specific Synthesis

It is generally accepted in the speech synthesis field that unconstrained text-to-speech synthesis is the only goal of the field. The culture of processing unconstrained input has come about because even early systems were quite capable of producing reasonably intelligible speech from unconstrained input. This is possible in TTS because lexicons and letter to sound rules can convert any letter sequence into a phone sequence. This phone sequence can then be converted into sound using rule or diphone based waveform generation. Of course the quality of the resulting speech is much less natural than human speech, but the fact that even basic systems could handle unconstrained input led researchers to concentrate on improving the quality of speech for this unconstrained input task.

This situation is somewhat unusual in speech and language processing. For example, in speech recognition, there has been a very noticeable development over the last 30 years, from single-speaker, isolated word, low vocabulary tasks to speaker-independent large vocabulary continuous speech tasks. Roughly speaking, the accuracy of speech recognisers over the last 30 years hasn't changed much as error-rates are usually quoted as being less that 10%. However progress has certainly been made because the tasks have been getting steadily harder.

The question is therefore, could more progress be made in synthesis by first making the task easy enough so that more or less perfect synthesis is possible, and then steadily increasing the difficulty of the task until perfect synthesis for unconstrained input is achieved? Yi and Glass (1998) have summed up this situation by use of the graph shown in figure 2. The two previously mentioned speech output methods, unconstrained TTS and slot and filler synthesis are shown on the graph as being examples of the two approaches.

This paper presents *Phonological Structure Matching*, a new algorithm that takes approach B. This algorithm is domain-specific in that it is geared towards producing high performance in a limited domain, just as with a speech recogniser. In considering the problem of domain-specific synthesis, some aims and principles were set out to guide the development of this type of system. It was important that the main limitations of slot-and-filler synthesis were removed. While the vocabulary of domain-specific systems will certainly be specialised, it is simply too prohibitive a restriction to have a fixed vocabulary. Again, with regard to the grammar, more flexibility is needed than with the slot-and-filler approach. It was also considered important to build a system that was automatically trainable/adaptable to new domains. In other words a general technique for domain-specific synthesis was the requirement, rather than a solution for a particular domain.

The proposed solution works for specific domains not by having a pre-determined vocabulary or grammar, but rather by using pre-recorded domain-specific language data to train the system. In other words, the idea is to *bias* the synthesizer's viewpoint of what language is to cover the words and constructions found the in the given domain.

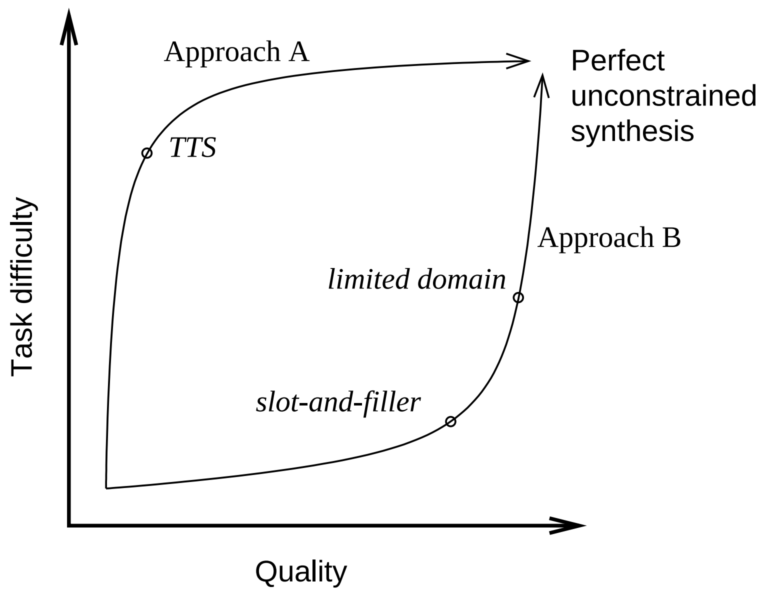

Figure 2. Different approaches to synthesis development, adapted from Yi and Glass (1998). In approach A, unconstrained input is the achieved first and then quality is improved. In approach B, near-perfect quality is achieved first and then flexibility is improved

3. Domain Descriptions

(*a*) *ILEX Museum Guide*

ILEX is a natural language generation system built to serve as a museum guide. It uses a database of museum exhibits which contains a variety of information about each exhibit. Rather than produce a canned description of a given exhibit, ILEX is intelligent in that it delivers unique descriptions of objects depending on a number of contextual factors. ILEX keeps track of which exhibits have already been seen, and hence when viewing a room of Roman swords, the system only gives background information for the first exhibit. As the visitor moves around the exhibits, only the particular details of each exhibit are explained, and these are often contrasted with previous exhibits.

(*b*) *Jupiter Weather Information System*

The Jupiter system is a weather information system developed by the Spoken Language Systems group at MIT. In response to spoken user queries, the system finds web based weather information systems, analyses their content and generates a suitable reply. For this domain, we used 200 typical messages from the system as training data. Within these 200 sentences, there were about 600 unique words. This domain is interesting in that although weather reports are often formulaic, it is still the case that new constructions are occasionally used. The majority of vocabulary items are place names, and while the training data covers the names of the most frequently requested places, new names often occur, and hence the synthesis component must be able to handle this.

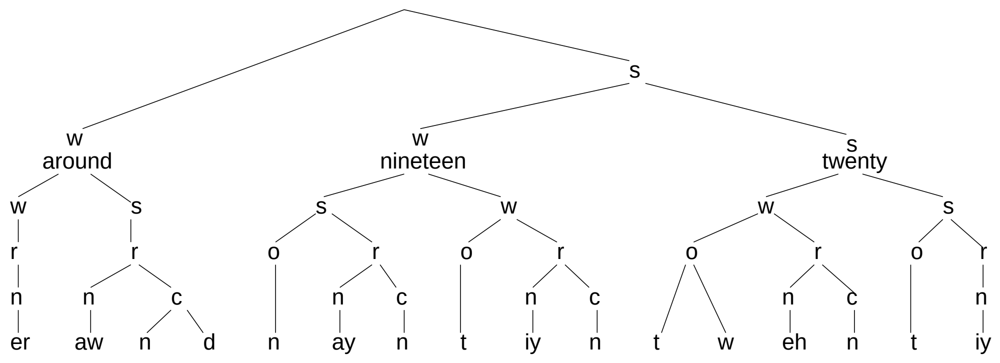

Figure 3. Fragment of a phonological tree

4. Speech Data

The algorithm uses pre-recorded speech from each domain. 62 paragraphs of item descriptions were used for the ILEX domain. The labelled speech contains over 6000 words and 22000 phones. The Jupiter data comprises 200 weather reports, making about 6000 words and 17000 phones. In addition to the domain-specific data, 450 Timit style sentences [**?**] were also used. The utterances were recorded from a single speaker and hand labelled for phone boundaries. The syntactic structure of each sentence was also labelled by hand. A held out set for each domain was used for testing. In the Jupiter domain, 5% of the test set words were not in the training data. In the ILEX domain, this figure was 25%.

5. Basic Phonological Structure Matching

(*a*) *Phonological Trees*

The phonological structure matching (PSM) algorithm is based on concatenating appropriate arbitrary sized units of speech from a database. More specifically, nodes in a phonological target tree generated by the NLG system are matched against nodes in a set of phonological database trees in order to find the biggest units of speech in the database to concatenate.

The database preparation stage is performed off-line and involves building a phonological tree for each utterance. In the current setup, the phonological tree is constructed by combining the metrical tree for the sentence with the sub-syllabic phonological structure. Metrical trees are binary branching trees whose nodes have relative metrical strength relations. If a node is labelled strong, its sister will be weak and vice versa. The above-word part of the metrical tree is formed by first copying the syntax structure generated by the NLG system and then assigning strong and weak nodes according to the nuclear stress rule and other conventions described in Liberman (1975). Below the word, a binary branching tree is formed between the words and syllables, again according to conventions laid down in Liberman (1975). Below the syllable, a traditional onset-rhyme structure is formed which links the syllables to the phones. The result is a single tree which completely describes the phonology of the utterance from phones to the sentence node. Because the timings of the phone boundaries have been marked (by hand) it is an easy matter to determine the start and stop time of any constituent in the tree. An example of part of a tree is shown in figure 3.

At run-time, the first task is to produce a *target tree* representing the phonological structure of the utterance we want to synthesize. This is formed in a similar way to the database trees. The syntactic tree and words come from the output of the NLG, which is then mapped into a metrical tree as above. The sub-word part of the tree is created by using a lexicon and the metrical and sub-syllabic structure rules. This tree is called the target tree as it represents the phonological structure of the utterance we wish to synthesize.

(*b*) *Finding Candidates*

Given the target tree and database trees, the next task is to match nodes in the target tree with those in the database. As each node in the database tree represents a section of recorded speech waveform, the idea is that by finding nodes in the database trees which match the target tree, we are effectively finding suitably stretches of waveform that can be used in actual synthesis.

First the root node of the target tree is set to be the current node. The database of trees is then searched to see if any node matches the current node. A match is taken to be the minimum requirement for a potential synthesis unit and occurs when the trees beneath the current and database node match with regard to structure, and and have the same terminal nodes (phones). At this stage, other information (such as strong/weak metrical information) is ignored. Each match is added to a list of candidates for the current node. If no matches are found, each daughter of the current node is set to be the current node and the process is repeated. In the worst case, the current node will be a terminal phone node, and there will definitely be matches to that as all phones are present in the database. The result of this is a target tree which has some of its nodes labelled as candidate nodes.

Candidates can be any sort of linguistic unit including phones, syllables, words, phrases and even whole sentences. The top-down search algorithm is designed to pick candidates high up in the database tree, which naturally correspond to longer units. There are several benefits in having longer units. In any type of concatenative synthesis, joins between units can cause distortion, and so reducing the number of joins should help improve the quality of the speech. Associated with this is the basic fact that in concatenative synthesis, “the whole is greater than the sum of the parts” with regard to the nature of phone sequences. For example, while the phonological representation of the word “tests” may be /t e s t s/, the co-articulation of the /s t s/ sequence is extremely complex, and hence it is very hard to decide where the boundaries between the phones occur. Because of this, a single unit which contains this sequence should sound substantially better than a set of units which follow the phonological pattern. As the tree matching algorithm chooses the largest possible units, there is a good chance it would find a match to the word “tests” and if not, at least a consonant cluster matching the /s t s/ sequence (e.g. from the word “rests”).

For higher level units, other factors make longer units more preferable to concatenated sequences of shorter units. Natural rhythm is one of the hardest properties of speech to reproduce syntheticly, but by using units which span several syllables, rhythm effects can be implicitly modelled within the unit.

The tree matching algorithm has significant consequences for domain specific synthesis. While the utterances in the museum domain can’t be generated by a simple slot and filler mechanisms, it is the case that certain stock phrases such as “this is an example”, “in the sixth century” or “collector’s item” occur quite often. If the database has a large amount of domain-specific data, these units will naturally be found frequently, leading to longer overall units on average.

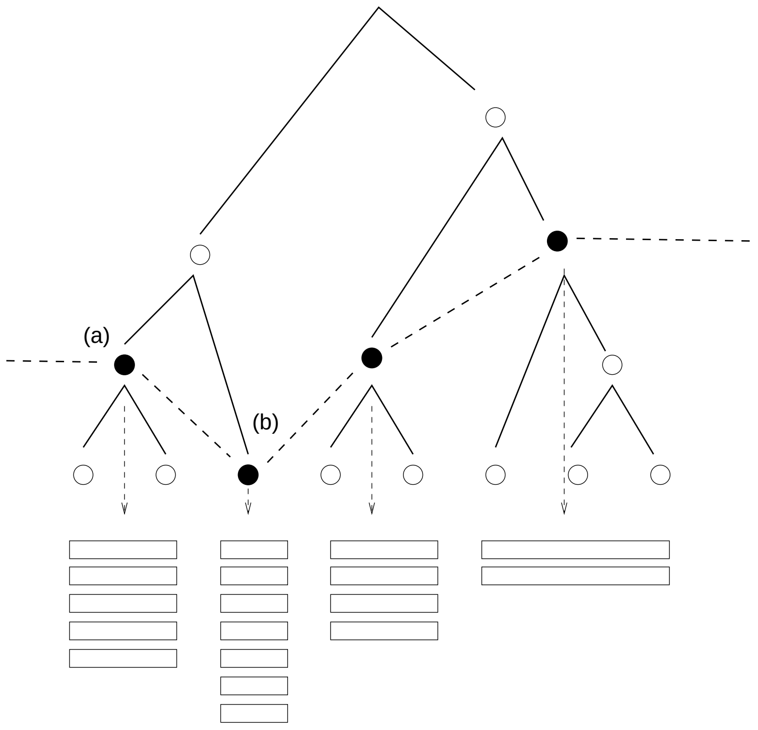

Figure 4. Target tree with candidate nodes at different depths. The candidates at each node form a candidate list, shown at the bottom. Each candidate relates to a different section of speech waveform of the speaker saying the information described by the node.

(*c*) *Selecting Candidates*

The target tree contains multiple candidates and from these the single best set of units needs to be chosen. Although the candidate units are of arbitrary length, the tree structure ensures that there is no overlap between units, which can be a potential problem in non-uniform unit synthesis. Figure 4 shows an example target tree in which the nodes containing candidates are drawn in black. Although node (a) may have units which are higher level and longer than those for node (b), all the candidates for node (a) end at the same point and hence there is no overlap between these and the node (b) candidates. Because of this, a linear list containing the set of units for each candidate node can be created. This list can be thought of as being made up from a number of slots of arbitrary length, where each slot contains multiple candidates. This is also shown in figure 4.

(i) *Target and Concatenation Costs*

The basic tree matching algorithm only performs a rough match between a target node and a node in the database, and hence there are still substantial differences between the various candidate units for a node. A more detailed match is now used to see how well the candidate units match the target node. This match looks at factors such as the strong/weak values in the tree and the position of the node in the tree (phrase-final units sound quite different from phrase initial, etc). According to this measure, each candidate unit is assigned a *target cost* which represents the distance between the target and candidate unit (a cost of zero indicates a perfect match).

It is also possible to measure how well two candidate units join together. This is known as the *concatenation cost*. In the PSM algorithm, the concatenation cost is a calculated using phonological and acoustic information. Each candidate unit for a given target node will have exactly the same phone sequence at its terminal nodes. But the phones immediately preceding and following the unit may differ from candidate to candidate. Experience with diphone synthesis has shown that unit context is important in ensuring smoothing joins. For example if we have a phone /X/ in the context of phones /b/ and /c/, /b X c/, this will join smoothly to a phone /c/ in the previous context of /X/, e.g. Xcd. Therefore a smooth join can be expected between a unit /X/ followed by a unit /Y/ if the phone immediately preceding /Y/ in the original speech and phone immediately following /X/ in the original speech are the same. In diphone synthesis, it is also preferable to join phones in the middles rather than at their edges. Given these basic observations, concatenation cost is calculated by a mixture of phonological information (whether the units are in the appropriate context, thus allowing joining in middles) and acoustic information, calculated by measuring the Mahalanobis distance† between the acoustic features of each frame. Mel-scaled cepstra [**?**], F0 and energy are used for the acoustic features.

(ii) *Search and Concatenation*

Selecting the best set of units is a compromise between choosing the units with the lowest target and concatenation cost. As each unit affects the concatenation cost of the next candidate, the selection of candidates can't be done locally but rather has to be done for the whole utterance. This is achieved by using the Viterbi algorithm. The candidate list is turned into a lattice by making a path between each possible pair of nodes at a the boundary between two candidate slots. The Viterbi algorithm moves left to right through this, and in doing so it calculates a partial path cost, which is the sum of the target and concatenation costs of units in a given path. The Viterbi algorithm works by making use of the fact that for a given unit, only the lowest cost path up to that point need be used, any other paths will never be in the lowest overall path for the sentence. So only the single best path for each candidate unit need be kept at each point. Once the search terminates, the units forming the path with the lowest overall cost is selected.

Given a set of candidate units, the final waveform is constructed by extracting the waveforms corresponding to the chosen units and concatenating them.

6. Analysis of the Phonological Structure Matching Algorithm

As diphone synthesis and acoustic/phonetic unit selection are among the most popular algorithms currently being used, it is useful to discuss how they perform against the PSM algorithm.

In both diphone and acoustic/phonetic unit selection synthesis, the waveform generation module shown in figure 1 often comprises of three phonetic components which then feed into the unit synthesis component. This is shown in figure 5. These components take the input phonological representation and use rules or other algorithms to generate a phonetic phone sequence, a duration in seconds for each phone and a F0 contour. The phonetic

† An extension of simple Euclidean in which each component of the vector is normalised with respect to its variance.

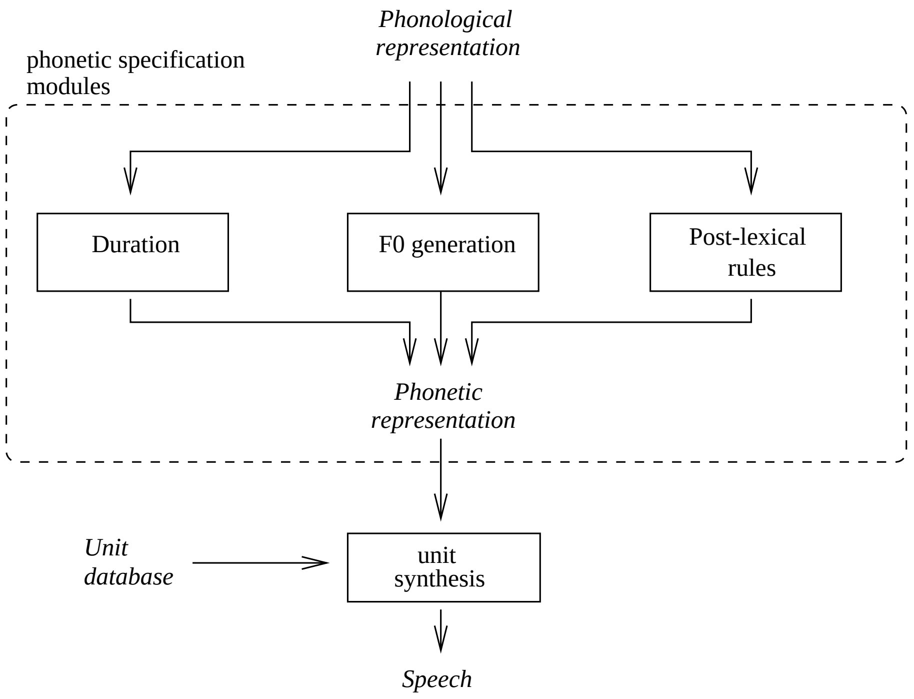

Figure 5. Standard approach to synthesis where a phonological representation is mapped into a phonetic one before selection takes place.

phone sequence is meant to represent the sequence of phones actually observable in the waveform. For example, phone reduction, deletion and assimilation are represented.

The output of these components forms a phonetic representation which is then fed to the unit synthesis module. If this is a standard diphone synthesizer, the diphones for the phone sequence are found in the unit database and concatenated. This concatenated waveform will have the F0 and duration of the diphones as they were recorded, and this is unlikely to match the phonetic specification of pitch and duration produced by the F0 and duration generation module. Signal processing techniques such as PSOLA (Moulines & Charpentier, 1990) and residual excited LPC (Hunt et. al, 1989) are therefore used to change the F0 and duration of the concatenated waveform to match the phonetic specification.

Diphone synthesis has two main problems. Firstly the signal processing introduces distortion. Secondly, pitch and duration aren't the only factors which make two tokens of the same phone sound different. For example, phones in stressed syllables are different from ones in unstressed syllables, and phones in phrase initial position sound different from ones in unstressed position. A proposed solution to these problems has been termed *unit-selection* and a number of systems can be said to belong to this framework (Sagisaka et al, 1992), (Hunt & Black, 1996), (Campbell & Black, 1996), (Donovan & Woodland, 1995), (Breen & Jackson, 1998), (Black & Taylor, 1997), (Conkie, 1999), (Cronk & Macon, 1998).

While the details differ, the basic principle behind unit-selection is to have multiple examples of each type of unit (e.g. diphones) which have different inherent pitch and duration. Instead of selecting just on phone identity as in diphone synthesis, pitch and duration are also taken into account. The result is a waveform whose pitch and duration match the phonetic specification more closely than with diphone synthesis. From here, systems differ as to whether or not they use signal processing. Those that do usually only have to make small adjustments, and so the distortion that is introduced is less than with diphone synthesis. Other systems abandon the use of signal processing altogether, in the hope that the prosody of the concatenated waveform is close enough to the specified prosody.

The PSM algorithm can be viewed as a type of unit selection algorithm in that it selects from multiple units of the same basic type. Indeed the use of the Viterbi algorithm to find the path which best optimises target and concatenation cost is taken directly from the Hunt and Black unit selection algorithm (Hunt & Black, 1996). The major difference between the PSM algorithm and the others is that PSM selects on *phonological* criteria while the others select on *phonetic and acoustic* criteria. With regard to figure 5 the PSM algorithm can be viewed as simply moving the selection criteria up a level in the linguistic hierarchy: the modules in the dotted box are deleted and selection is performed on the phonological representations directly.

The phonetic specification modules can be seen as a process which transforms the phonological specification into a phonetic one. For example, phonological information such as phrase finality is manifested in the phonetics by the relevant phones having a longer duration. There are three main reasons why we have to chosen to use phonological information. Firstly, a phonological representation is more compact than a phonetic one, and this reduces the size of the feature space used in selection. As the phonetic and phonological representations contain the same information (one is just the transform of the other), no loss of power or information is involved in making this decision. Secondly, the phonetic specification modules often make errors. If their output is used for selection, matches will be made to inappropriate targets. In speech synthesis systems, errors generally multiply throughout processing, so while the phonological representation may also contain errors, it will generally have the same or fewer errors than the phonetic representation. Finally, these modules take a somewhat crude view of the relationship between phonology and phonetics. For example, all assimilation is performed at a symbolic phone level. In the previously mentioned examples of the word “tests”, the post lexical rules have a choice between producing /t e s t s/, /t e s/ or /t e s s/ etc. In fact, none of these phonetic sequences really describes the complexities of articulation involved in pronunciation of this word. But by simply saying select a unit whose *underlying* phonology is /t e s t s/, the synthesizer doesn’t have to worry about the exact nature of the surface phone sequence.

7. Back-off and Signal Processing

As previously mentioned, unit selection systems fall into two groups, those which use signal processing and those which don’t. While it has been interesting to see how far one can get without signal processing, there is little chance this can really succeed due to the combinatorics of units. For illustration purposes consider the variation required of a single unit. In normal male speech, F0 can vary from about 80Hz to 270Hz, and taking a 10Hz quantisation as being adequate, that means we need at least 20 units of the same type to represent pitch variation. But the pitch is not constant throughout a unit, and so we need units for rising and falling F0 contours. Assuming this can be modelled by begining, middle and end values, the number of possible units to model all pitch variation is 20x20x20 = 8000. Duration variation is similar and we might need 15 units to cover this variation (from say 50ms to 200ms in 10ms steps). Given that the smallest practical units are diphones (about 2000 exist in English), we would therefore need 8000x15x2000 = 240,000,000 units, cor-

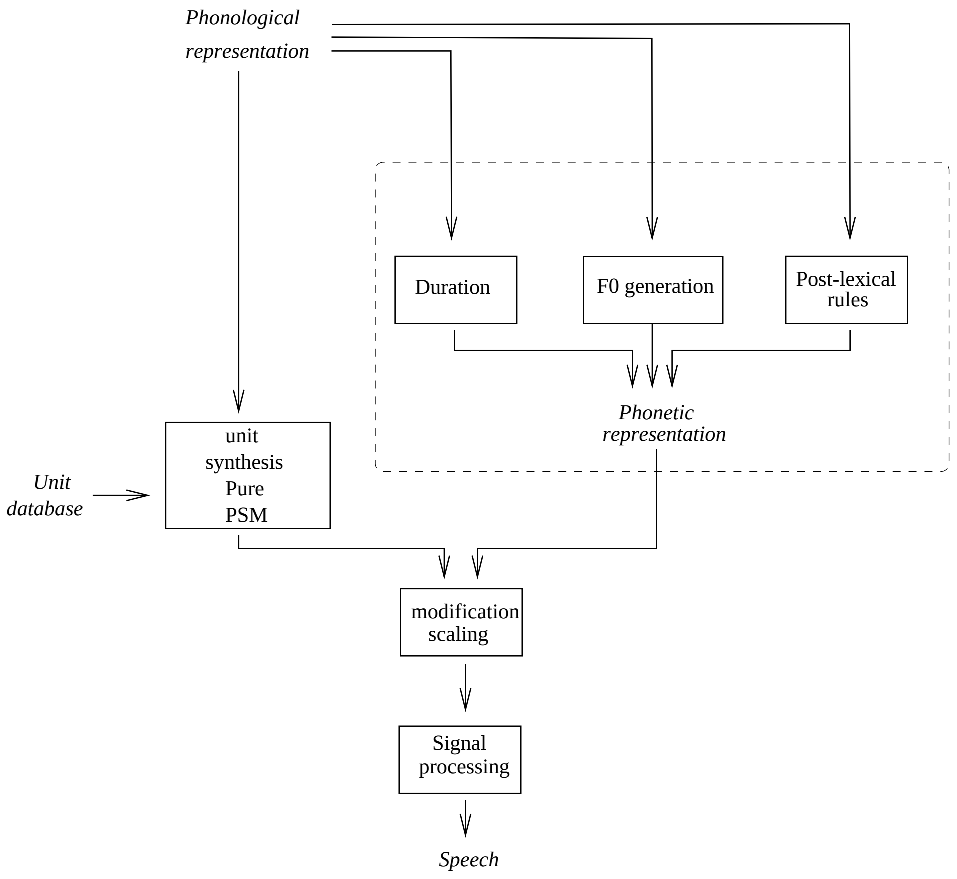

Figure 6. PSM combined with signal processing

responding to hundreds of years of recorded speech. That said, it is still sometimes the case that the distortion caused by signal processing is deemed to be worse than the speech having the wrong prosody. However, as signal processing algorithms improve through further research, the balance will swing back decidedly in their favour.

In acoustic/phonetic unit selection, the application of signal processing is straightforward - the signal processing is used to make the concatenated waveform's prosody the same as that of the output of the phonetic specification modules. With the PSM algorithm the situation is slightly more difficult as there is no phonetic specification. As it is unreasonable to require a signal processing module that performs phonological modification directly, we use a back-off scheme that employs a phonetic specification procedure similar to that of an acoustic/phonetic unit selection module, shown in figure 6. The PSM algorithm works as before, but a record is kept of how well the best unit for a given slot matches the target phonological representation. In parallel, a phonetic specification is produced as for phonetic/acoustic selection. A further module is now use to decide what the final prosodic modification should be. If the phonological match is good, the original unit prosody will be kept. If the phonological match is bad, the prosody from the phonetic specification is used and signal processing performs the necessary modifications. Usually, the situation is somewhere between the two and so a mixture of the unit and specified prosody is used, weighted to take into account the closeness of phonological match, the known accuracy of the phonetic specification and the amount of distortion that the signal processing will produce. This ensures that a suitable compromise between the naturalness of unmodified speech and the desirability of having suitable prosody is found.

8. Performance

No formal evaluations have been carried out yet, but it is worth giving some informal impressions regarding system quality. The best examples of the PSM are a vast improvement on diphone synthesis with regard to naturalness. Many of these good examples are bordering on being indistinguishable from natural speech. The reasons behind this improvement are those laid out above: using longer units with natural prosody and minimal signal processing. At present the PSM algorithm also makes some bad mistakes, and hence the occasional example is worse than diphone synthesis. We feel that these bad examples are more to do with teething troubles regarding a very new system rather than anything systematically wrong with the algorithm itself.

The voice quality of the PSM algorithm is fairly similar to that of acoustic/phonetic unit selection. Where the PSM really wins is in areas such as rhythm, timing, phrase-final lengthening and intonation. This is again for the above mentioned reason that the PSM algorithm models these implicitly rather than by using explicit modules which make errors.

9. Future Work

(*a*) *Phonological Structure*

Our choice of phonological structure is by no means optimal. While the use of metrical trees has proved successful, there are many other types of phonological structure described in the literature and many of these may prove more suitable. The best representation for this algorithm is one that describes the phonology as accurately and compactly as possible: an accurate representation will cover all the required affects adequately and a compact representation will help in producing tractable cost functions.

(*b*) *Training*

There are a number or parameters and weights in the system which perform functions such as measuring the relative important of stress vs phrase-finality, whether F0 continuity is more important than spectral continuity, and how much signal processing modification to use. Currently these are set by hand using informal listening experiments. This is obviously not ideal, but it is difficult to see an easy alternative. The fundamental problem is that there is no straightforward relationship between acoustic and perceptual measures of speech. While spectral discontinuity can be measured taking the mahalanobis distance of two spectra, this is only a vague indicator of how humans perceive spectral discontinuity at unit joins. Some recent work (Chappell & Hanson, 1998), (Plumpe & Meredith, 1998), (Wouters & Macon, 1998) has started to focus on the design of perceptually weighted acoustic measures, and these might be used in the future in training the system parameters.

(*c*) *Candidate Selection*

At present, the PSM algorithm attempts to find the biggest possible matches in the database to a given node. If a match is found, the remainder of the database is searched for other similar matches. If no match is found, the units matching the node's daughters and searched for. Problems can occur if only a small number of matches are found for a node and none of those matches are particularly good. In these cases, it may have been better to use smaller units which match the target better. Current work is looking at ways to extend the search past a node with candidates to find candidates for its daughters. During selection, the relative merits of long units with high target costs are then matched against shorter units with lower target cost but an extra concatenation costs between the units.

(*d*) *Measurements of Task Difficulty*

It would be useful to have a measure of how difficult a particular domain is, as this could give an indication of expected quality from a system used in this domain. We have not as yet come up with any firm decisions regarding this, but it seems possible that the measures of vocabulary size and perplexity used in speech recognition could be adopted. Perplexity is a measure of word entropy and measures the amount of regularity in a corpus. This is a useful indicator in synthesis because of joins between units. If the perplexity of a domain is low, the number of possible word sequences will be lower and hence the chance of units being found with the appropriate phonological context is higher.

(*e*) *Text-to-Speech*

The reason the PSM algorithm has been put forward as a useful solution to concept-to-speech is that it is easily adaptable to a given domain. However, the algorithm also works in standard text-to-speech synthesis. The TTS problem can be seen in PSM terms as simply having a much larger domain, and hence the training data should cover a wide variety of text styles rather than being domain specific. The output speech quality is obviously worse for TTS than for CTS, firstly because the domain is bigger and secondly because of errors in text analysis. However, the basic strength of the PSM algorithm (that it uses phonological rather than phonetic/acoustic selection criteria) will also be true in a TTS task.

Acknowledgements

A substantial part of this work was carried out while the author was a visiting researcher at the Department of Speech, Music and Hearing at KTH Stockholm. The author wishes to express his thanks to Björn Granström for making that visit possible and also to everyone at KTH for making the experience there so rewarding.

The author would also like to express his thanks to Alan Black and Richard Caley for many useful discussions regarding this work. Joe Polifroni and Janet Hitzeman deserve thanks for generating the Jupiter and ILEX sentence training sets respectively.

This work was partly funded by the UK Engineering and Physical Science Research Council (grants GR/L53250 and GR/K54229).

References

Black, A. and Taylor, P. (1997). Automatically clustering similar units for unit selection in speech synthesis. In *Eurospeech97*, volume 2, pages 601–604, Rhodes, Greece.

Breen, A. P. and Jackson, P. (1998). A phonologically motivated method of selecting non-uniform units. In *International Conference on Speech and Language Processing 98*.

Campbell, N. and Black, A. (1996). Prosody and the selection of source units for concatenative synthesis. In van Santen, J., Sproat, R., Olive, J., and Hirschberg, J., editors, *Progress in speech synthesis*, pages 279–282. Springer Verlag.

Chappell, D. T. and Hansen, J. H. L. (1998). Spectral smoothing for concatenative speech synthesis. In *International Conference on Speech and Language Processing 98 98*.

Conkie, A. (1999). A robust unit selection system for speech synthesis. In *137th meeting of the Acoustical Society of America*.

Cronk, A. and Macon, M. (1998). Optimized stopping criteria for tree-based unit selection in concatenative synthesis. In *International Conference on Speech and Language Processing 98*.

Donovan, R. and Woodland, P. (1995). Improvements in an HMM-based speech synthesiser. In *Eurospeech95*, volume 1, pages 573–576, Madrid, Spain.

Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *International Conference on Acoustics Speech and Signal Processing, volume 1, pages 373-376,*. IEEE.

Hunt, M., Zwierynski, D., and Carr, R. (1989). Issues in high quality lpc analysis and synthesis. In *Eurospeech 89*, pages 348–351.

Liberman, M. (1975). *The Intonational System of English*. PhD thesis, MIT. Published by Indiana University Linguistics Club.

Moulines, E. and Charpentier, N. (1990). Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9(5-6):453–467.

Plumpe, M. and Meredith, S. (1998). Which is more important in a concatenative speech synthesis system - pitch duration of spectral discontinuity. In *Third ESCA/IEEE Workshop on Speech Synthesis, Jenolan Caves, pages 231-326*.

Rabiner, L. R. and Juang, B-H. (1994) *Fundamentals of Speech Recognition*. Prentice Hall.

Sagisaka, Y., Kaiki, N., Iwahashi, N., and Mimura, K. (1992). ATR – ν-TALK speech synthesis system. In *Proceedings of International Conference on Speech and Language Processing 92*, volume 1, pages 483–486.

TIMIT: The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (1990), Collected by the Massachusetts Institute of Technology (MIT), Stanford Research Institute (SRI), and Texas Instruments (TI). The printed documentation which is also available from NTIS (NTIS PB91-100354),

Wouters, J. and Macon, M. W. (1998). A perceptual evaluation of distance measures for concatenative speech. In *International Conference on Speech and Language Processing 98*.

Yi, J. R. W. and Glass, J. R. (1998). Natural sounding speech synthesis using variable length units. In *International Conference on Speech and Language Processing 98*.