SPRUCE — High Specification Text-to-Speech Synthesis

Eric Lewis – University of Bristol
Mark Tatham – University of Essex

Reproduced from P. Lee (ed.) *Proceedings of the Second Language Engineering Convention.* London: Department of Trade and Industry, 145-152

Text-to-speech synthesis allows a computer to read text aloud without the direct use of recordings of human speech. Even when there is an indirect use of recordings (as in waveform concatenation), an essential property of the system is that it should be able to speak sentences which have not been recorded. ***SPRUCE*** is a high-level text-to-speech synthesis system which has these properties.

> [*Note:* The presentation in this publication is oriented toward the synthesis properties *per se* of SPRUCE in view of the general theme of the conference. However, the main objective of the overall SPRUCE project was and remains the building of a *computational model of human speech production*. This objective should not be confused with what we perceive to be the 'simpler' task of engineering a device to convert text to speech without due regard to a formal model of human speech production.]

High-level and low-level synthesis

It is important to distinguish between *high-level* and *low-level* synthesis.

- A *low-level synthesiser* is the actual device which is used to create the output sound wave. Examples of low-level synthesisers are the *Holmes* parallel formant device, the *Klatt* hybrid parallel and cascade formant device, the *PSOLA* concatenated waveform system developed by the *Centre National d'Etude des Télécommunications*. These are commercially available low-level synthesisers. But they cannot speak, however, unless they are driven by a matching high-level software synthesiser.
- A *high-level synthesiser* is responsible for generating the input to a low-level synthesiser. If we isolate the output of a high-level synthesiser we find that it contains all the necessary information to make a low-level synthesiser speak correctly. The input to a high-level synthesiser is the text to be spoken and its output is a file or signal for driving a low-level synthesiser. There are several tasks which the high-level synthesiser must perform, including conversion of the orthographic input text to a phonological representation and the assignment of correct intonation patterns.

Low-level synthesisers consistently out-perform their matching high-level systems. That is, the quality of the final speech signal in terms of its naturalness is usually determined by the high-level system. This is easily demonstrated by driving the low-level synthesiser with a carefully hand-crafted input — the result can often be indistinguishable from real speech. As soon as the system is driven by a high-level synthesiser it immediately takes on the characteristics of relatively unnatural synthesised speech. It follows that commercial success in producing good quality synthetic speech rests primarily with the high-level synthesiser.

At present commercially available products are either low-level synthesis systems, or integrated systems where the high-level and low-level subsystems have been 'tuned' to be mutually dependent to produced optimised output quality (for example, *INFOVOX* or *DECTalk*).

SPRUCE is the exception — it is a ***high-level synthesiser*** which is designed to work independently of the low-level device it is driving. That is, ***SPRUCE*** has been designed in principle to drive *any* low-level synthesiser.

SPRUCE

SPRUCE is a high-level text-to-speech synthesis system which is capable of creating files suitable for driving most low-level synthesisers, including both *formant* and *waveform concatenation* systems. For example, a ***SPRUCE*** file can be created to drive the *Holmes*-designed *JSRU* parallel formant synthesiser which requires an input file specifying in 10ms frames the values of eleven parameters, including formant frequencies, formant amplitudes and fundamental frequency. ***SPRUCE*** automatically generates this input file format. Another example of a synthesiser successfully driven by ***SPRUCE*** is the *PSOLA* system from the *Centre National d'Etude des Télécommunications*. This is a diphone-based concatenated waveform synthesiser requiring for its input an allophone string indexed with duration and fundamental frequency for each individual segment. The low-level *PSOLA* system consists of two parts: a set of recorded and marked-up diphones and a set of algorithms for concatenating these and altering their overall durations and pitch. The system is driven by an input file which consists of a string of allophone symbols together with a starting and ending pitch (in *Hz*), and a duration (in *ms*) for each. ***SPRUCE*** automatically generates this input file.

SPRUCE has also successfully driven the *IBM* concatenated waveform synthesiser. This is similar to the *CNET* system, but instead of using diphones it conjoins much longer stretches of stored waveform. The input files for this system also require allophone symbols, pitch and duration markers. ***SPRUCE*** automatically generates this input file format too.

The Overall Architecture of SPRUCE

The ***SPRUCE*** engine has been written in C, and is currently available for SUN workstations (UNIX) and PC compatibles (MS-DOS).

SPRUCE is a knowledge based system, and in its basic form relies on a *word-level dictionary* to gain access to the linguistic knowledge essential for producing a natural sounding output. However, extensions to the basic form of ***SPRUCE*** ensure that the system is not limited to words found in the dictionary, but is in principle capable of speaking any input text.

Dictionary knowledge provides information to two principal modules in the system: a module of phonological tasks and a module of prosodic tasks. These modules are processes, relying on knowledge based algorithms to make contextually determined modifications to dictionary entries which could not be predicted prior to a scan of any particular input text.

1. *Phonological tasks* are those which alter the basic pronunciation of an individual word according to its context in the input text. A modern non-linear model has been adopted to underlie ***SPRUCE*** phonological tasks. This leads to superior control of some of the finer nuances of pronunciation. A non-linear hierarchical approach uses the syllable as the basis of the algorithms applied to the segmental aspects of phonological processing — the concept of individual segments within the syllable has been abandoned in favour of a purely symbolic representation of segments at this stage of the synthesis.
2. *Prosodic tasks* are those which alter fundamental frequency and durations throughout a sentence according to the way in which the syntax (and to a certain extent the semantics) of a sentence unfolds. For this reason the input text has to be parsed for contextually determined syntactic information which could not be known to the system in advance. The prosodic contours of duration and intonation are therefore assigned in this module.

Finally it is necessary to assemble an output file which contains all the information necessary to drive a low-level synthesiser. The detail of information required in the file depends on which synthesiser is to be driven.

The Dictionary

Let us consider the basic ***SPRUCE*** system. This makes full use of a dictionary. There are three possibilities

- the dictionary can be very large and contain most of the words of English (say, 100,000 words);
- the dictionary can contain a smaller number of 'most frequent' words (say, the 10,000 most frequent words in general usage);
- the dictionary can be of the 'limited domain' type and contain potentially a complete vocabulary associated with a particular subject area. An example of this would be a weather forecasting application.

The most satisfactory of these three is the third, limited domain type. Here all words associated with a particular domain are known to the system — together with as much linguistic knowledge as is necessary for a natural sounding output.

When ***SPRUCE*** receives a text input it scans each incoming sentence, looks up each word in the word dictionary and retrieves important semantic, syntactic and phonological information needed for synthesising the word from both segmental and prosodic viewpoints.

The Phonological Tasks

The non-linear model of phonology adopted by ***SPRUCE*** is based on a modified version of *Articulatory Phonology*. This particular theory of phonology has certain shortcomings in respect of its handling of some relationships between phonological and phonetic representations — these have been corrected in ***SPRUCE***. The model has been chosen because in enables us to establish a good transition between highly abstract representations (as in the orthography of the input text or in the high-level phonological representations of words in the dictionary) and the low-level phonetic representations needed in ***SPRUCE*** output files. Linear models of phonology (as found in early generative grammar, for example) are not suitable because they do not handle well the transition from phonology to phonetics. Earlier pre-generative models are not suitable because they fail to adopt a hierarchical approach in their descriptions (placing phonology and phonetics on the same level, for example).

The phonological tasks in ***SPRUCE*** are essential for the smooth and natural transition from word to word as sentences unfold. The tasks are essentially concerned with contextual transformation which occur in the pronunciation of words and syllables. ***SPRUCE*** develops its phonological tasks with operations performed basically at the level of the syllable, making, for example, subtle changes to the way the beginning and ending of a syllable will sound when pronounced — even if it appears to begin and end with the 'same' sound.

Phonological tasks can be related to prosodic tasks since some of them are time dependent, and part of prosody is concerned with durational aspects of utterances.

The Prosodic Tasks

SPRUCE prosody — the assignment of intonation and timing patterns for utterances — is based on a parse performed on each input sentence. The parsing algorithm is very fast and is the result of experiments designed to determine the optimum balance between speed and error, there being an inverse correlation between the two. The ***SPRUCE*** parsing algorithm knows that it must produce a result and knows also what the probability of error is for any one operation performed. Together with the subsequent algorithm for assigning intonation the prosodic system attempts to produce an intonation contour which minimises error. The errors which do occur (and some are inevitable for any system) are 'chosen' for minimal perceptual impact. This is an important aspect of ***SPRUCE*** prosody and contributes considerably to its success. Sensitivity to its own errors is a major design philosophy in ***SPRUCE***.

The parsing is primarily syntactic, though some rudimentary semantic parsing is used if the system detects that syntactic parsing will not provide optimal results (in a given time). The results of the parse are used directly in generating the appropriate prosodic contours - intonation and timing — for the utterance. The algorithms used are complex and hierarchically organised in the general spirit of the phonological model chosen as the basis for ***SPRUCE***. They have been, however, carefully optimised and run very fast; on a 486 66M*Hz* MS-DOS machine intonation assignment occurs in real time in the vast majority of cases.

Pragmatics

SPRUCE is under continuous development. It is being released at this moment in time because the basic high-level synthesis engine is complete and ready for commercial exploitation. However, the next major extension of the engine has already begun.

As yet no commercial synthesis system is able to respond to the speech output demands of the varying pragmatics of sentences. Pragmatic effects in speech are nuances of pronunciation which convey to a listener aspects of the meaning of a sentence which are not encoded in the words themselves (semantics) or in their ordering (syntax). Whenever a person speaks he or she necessarily encodes something of their feelings and attitude by means of subtle usage of 'tone of voice'. Linguists have made some observations about this phenomenon. Phoneticians and phonologists are beginning to establish the acoustic characterisation of some pragmatic effects. Indeed in the Essex **Speech Research Group** there is a renowned nucleus of ongoing research in *Pragmatic Phonetics.*

It is clear that the next generation of speech synthesiser, particularly when the application involves a dialogue situation, must include a systematic rendering of pragmatic effects. Thus for example effects such as the use of insistent or warning tones of voice become necessary in situations involving danger. A pragmatics module is under development for ***SPRUCE***, and is already beginning to show some promise. The demonstration tape accompanying this description has an example of subtle variation of the acoustic signal for the phrase 'five-thirty' for conveying progressively increasing emphasis and insistence.

For the successful inclusion of a pragmatics module in high-level synthesis it is necessary to have not just an understanding of how the acoustic signal must be made to vary, but also a formal, theoretically based means of triggering the varying acoustic signal at the right moment. The ***SPRUCE*** pragmatics algorithm is beginning to show the necessary robustness for some of the commoner pragmatic effects.

Output Files

As a high-level synthesis system ***SPRUCE*** is designed to produce output files for driving the major types of commercially available low-level synthesisers. These vary considerably in their requirements. Let us consider two extremes

a. the parallel formant synthesiser designed by John *Holmes* which has been the basis of most work in speech synthesis in the UK for thirty years, and
b. the *PSOLA* concatenated diphone waveform system from *CNET* — a modern system capable of very high quality speech.

a. The *Holmes* synthesiser (which has been used as the basis for several ***SPRUCE*** demonstrations) is driven by an input file arranged as a matrix of rows and columns. The columns are the various parameters of the system, and there are eleven of these, including the frequencies and amplitudes of three formants, the amplitude of a fourth fixed formant, a continuously variable switch for selecting between periodic or aperiodic excitation, fundamental frequency, etc. The rows are 10ms time frames. The system requires, therefore, a file specifying numerical values for each of 11 parameters updated every 10ms.

The high-level synthesiser in this case must supply continuously changing formant values ('transitions'), as well as timing information (expressed via the row format) and fundamental frequency information. ***SPRUCE*** is designed primarily to provide timing and fundamental frequency information — for driving the *Holmes* synthesiser the additional formant information has to be computed. This is done in the general spirit of the phonological model incorporated: an inventory of words and/or syllables is assembled based on normalised parametrically analysed real human speech. On demand, words or syllables are pulled from the inventory and adjusted or denormalised for inclusion in the particular utterance being synthesised. The 'neutral' files thus assembled are now subjected to normal adjustment of ***SPRUCE*** generated intonation and timing to produce a final file in the required format.

Speech resulting from this technique has excellent prosodics, but its quality is very dependent on the accuracy of the parametric analysis of the original human speech. Nevertheless the output is very much better than traditional systems which rely on single sets of stored values for individual phonemes which are then strung together according to linear phonological models. This class of system includes the *Holmes JSRU* text-to-speech synthesis system as well as *DECTalk.* ***SPRUCE*** successfully drives these systems.

b. *CNET*'s *PSOLA* low-level synthesis system gives superior results to those produced by formant systems. The system consists of a set of diphones together with algorithms for conjoining them and adjusting timing and fundamental frequency. Input files for driving the system are much simpler than those required for formant synthesisers — it is necessary to supply only information concerning the duration and fundamental frequency of each allophone in an utterance. The *PSOLA* algorithms and the recorded diphone waveforms take care of all other information required.

   In driving such a system ***SPRUCE*** is working almost optimally since no time or computational power is required for computing parametric representations of utterances. ***SPRUCE*** quickly produces the appropriate files for driving a *PSOLA* low-level synthesiser on a sentence-by-sentence or phrase-by-phrase basis. All ***SPRUCE*** effort is devoted to computing correct timing and intonation patterns to match the input sentence. The only departure from a fully optimal working is the requirement to base the ***SPRUCE*** output file on an allophonic representation — which is contrary to the non-linear non-segmental philosophy. However the conversion is done rapidly in the final stages — all of ***SPRUCE***'s important computations are done earlier within the non-linear paradigm.

   The limiting factor in the *PSOLA* low-level synthesis system is the diphone set. The present set of English diphones at *CNET* is not entirely satisfactory, compared with, for example, the sets available for synthesising French or German. The ***SPRUCE*** team has experienece with diphone synthesis and can develop a special diphone set optimised for the system.

   The concatenated waveform class of system includes the *CNET PSOLA* design for French, German and Italian, and one or two other systems such as the recently described experimental system being developed by IBM (UK). SPRUCE successfully drives the CNET system with English diphones and the *IBM* non-diphone system.

SPRUCE Applications

The high quality output associated with ***SPRUCE*** is due in part to the accuracy of its prosodic algorithms, but is also due to the fact that it is dictionary-based. So long as the text input to ***SPRUCE*** contains only vocabulary which is already in the dictionary ***SPRUCE*** continues to provide a high quality output. When the input text contains words which are not in the dictionary the output quality may deteriorate on those words. It may also deteriorate for the whole sentence if ***SPRUCE*** fails to assign the correct syntactic markers to such words, and

therefore parses the sentence incorrectly. It follows that successful applications using ***SPRUCE*** currently are those whose vocabulary can be largely determined in advance.

In the past, the goal of many of the designs for speech synthesisers was to be able to achieve a spoken output from any input text. But if we step back and redefine the goal then success is much more likely. This is the accepted situation for speech recognition — all commercially available systems are vocabulary based; if an application extends beyond the 'known' vocabulary the system will fail.

SPRUCE works best where restricted vocabulary constraints are accepted. This applies to all situations where the synthesiser is to work together with a recogniser — that is, in a dialogue application. Here the limiting factor is the system which attracts the most constraints — the recogniser — not the syntheiser. ***SPRUCE*** will have a vocabulary at least as great at the recogniser, and then additional words associated perhaps with instructing the user on how to get the most out of the dialogue system.

Frequently Asked Questions

1. Is there a limit to SPRUCE vocabulary?

The answer to this question is in principle: no. There is no real reason why a particular ***SPRUCE*** application should not have a vocabulary of, say, 250,000 words (far greater than any recogniser which might also exist in the application). But there are two reasons for not having more words than necessary

- it takes time to set up such a vocabulary,
- dictionary storage requirements are increased.

2. What is available now from SPRUCE?

The ***SPRUCE*** engine is available immediately. This engine incorporates the vocabulary search module, the phonology and prosod **Centre National d'Etude des Télécommunications** ics modules and the modules which correctly derive the necessary output files. The engine does not contain any specific vocabulary nor an optimised method of speaking words or abbreviations not found in the vocabulary. In addition the basic engine has only a command line interface.

3. Why is the *SPRUCE* engine so basic?

The ***SPRUCE*** engine has been designed to maximise its ability to be adopted for the widest possible range of applications — this means that specific requirements are not included, like, for example, a particular windowed interface. The design incorporates carefully placed hooks, however, which enable quick tailoring of interface requirements to encapsulate the basic engine.

4. How do I know if *SPRUCE* is good for my application?

After you have heard some of the demonstration versions of ***SPRUCE*** we invite you to provide a small, minimal specification of your application. We will arrange a demonstration of ***SPRUCE*** working on this subset of your application. We will explain how the system can be enlarged to cover the complete application you have in mind. The final application-oriented version of ***SPRUCE*** will therefore have been developed optimally for your application — the only disadvantage will be a delivery delay of a few weeks, depending on the scope of the application.

Some Technical Details.

The ***SPRUCE*** system for producing files to drive a concatenated waveform system is currently composed of six C source modules totalling 140K bytes. The executable file is 180K and includes, as well as the compiled source code, some indexes for assisting in the

access of the binary dictionary and the inventory of phonetic/prosodic information stored on disk. These occupy about 55K of memory.

For a dictionary of about 100 words the binary lexicon occupies 3K of disk space and the inventory of phonetic/prosodic information occupies 1.5K. The application vocabulary can be as large as required, of course, but the consequences of increasing the vocabulary are a corresponding increase in the amount of memory and disk space needed. The system has been designed such that the disk access time is independent of the size of the dictionary.

For a full ***SPRUCE*** system comprising 100,000 words and 10,000 syllables it is estimated that the indexes would occupy 500K of memory and the lexicon/inventory would occupy about 5Mbyte of disk space.

SPRUCE has been developed on the PC platform and runs in almost real time on a 486 processor. The time taken to convert a sentence of about 10 words, once stored in the computer, to a file on disk suitable for input to a concatenated waveform system takes about 5/100ths of a second.

The code is ANSI-C and moves across onto UNIX systems with few changes required.