

Natural Language Processing in Text-To-Speech Synthesis

M. Oğuzhan KÜLEKÇİ

May 5, 2004

Abstract

This paper reviews the natural language processing techniques used in text-to-speech synthesis under two primary divisions as *word level operations* and *phrasing for prosody generation*. Word level operations are further divided into three subsections as *preprocessing*, *morphological analysis*, and *text normalization/homograph disambiguation*. The preprocessing subsection describes *tokenization* (word segmentation) for Chinese type languages and *vocalization* for semitic languages. The role of morphological analysis in speech synthesis is argued in the related subsection with its effects on word stress assignment, tagging, and homograph disambiguation. *Non-standard word resolution*, *word sense disambiguation*, and *named entity recognition* in text-to-speech synthesis are discussed separately in the third subsection. Phonological phrasing and intonational accentuation based on syntactic and semantic constituents are explored in the second primary division.

Contents

1	Introduction	3
2	Why use NLP in TTS synthesis?	7
3	Word Level Issues	12
3.1	Preprocessing Tasks	12
3.1.1	Tokenization	12
3.1.2	Vocalization	17
3.2	Morphological Analysis	20
3.3	Text Normalization and Homograph Disambiguation	24
3.3.1	Non-Standard Word Resolution	24
3.3.2	Ordinary Words Requiring Sense Disambiguation	29
3.3.3	Named Entity Recognition	31
4	Phrasing for Prosody Generation	38
5	Conclusion	49

1 Introduction

Speech synthesis is defined as the realization of an input text in a natural language as speech signals. Such a synthesizer is a full-TTS system if able to automatically convert text written in standard orthography of the language concerned into sound [Shih and Sproat, 1996]. This criteria restricts that the input to the system is not in a form of some phonetic transcription but instead a human readable text. An excellent full-TTS synthesizer is expected to read anything that a native speaker of the language can read. In that sense, it is worth noting that the human reader of the text has access to much more information than an automatic speech synthesizer, as he or she certainly *understands* the input. Additionally, the human reader brings experimental and theoretical knowledge at the time of reading and thus has more capabilities to transmit the tone and context of the text to the listeners. Although a writing is composed of finite number of graphemes, the speech realisations of those graphemes are infinite [Shalanova and Tucker, 2003]. That is, a written text has infinitely many speech realizations. The text-to-speech (TTS) synthesis problem may be stated as the task of generating the best among those realizations.

The quality of a TTS system is measured using two metrics: *intelligibility* and *naturalness*. The intelligibility of a speech synthesizer is mainly the ability of the system to generate the correct pronunciation for a given word so that the word is understood well when read. The naturalness on the other hand is somewhat a qualitative measurement of the emotion that the TTS system gives to the listener. Improving the naturalness metric be considered

as making the system as human-like as possible, so that a listener, say, at the other side of a telephone, will be in doubt as to whether the speech is produced by a TTS system or the reader is a human.

Recently, a competition was held in ESCA/COCOSDA¹ 1998, where 17 TTS systems were evaluated using these metrics. The test results indicate that the intelligibility of nearly all the synthesizers were at an acceptable level with small variations but the same was not true for naturalness [Beutnagel *et al.*, 1999]. Most systems did not perform at an acceptable level on the overall voice quality test perhaps because such systems did not pay necessary attention to textual and linguistic analyses. For a good prosody the system has to guess which words are to be emphasized and how much [Shih and Sproat, 1996]. A more natural sounding needs more information to be extracted from what is being read. Thus, the improvements need to be performed on language processing area.

A TTS system has to perform significant amount of work at *phonological, morphological, syntactic, semantic, and pragmatic* levels. Note that those levels are not disjoint and the system has to be seen as a whole. Although most of the recent synthesizers employ NLP on morphological and syntactic levels [Black and Taylor, 1994b, Pfister, 1995, Taylor *et al.*, 1998, Beutnagel *et al.*, 1999, Jilka and Syrdal, 2002, Black and Lenzo, 2003], the same cannot be said for the semantic and pragmatic levels. Problems faced in a synthesizer development very much depend on the language. The language independent part of the work is generally in the area of phonetics/acoustics

¹A workshop of European Speech Communication Association - International Committee for Co-ordination and Standardisation of Speech Databases held in Sydney, Australia. More information about COCOSDA is available at www.cocosda.org

[Shalanova and Tucker, 2003].

The genre of the text that the TTS is deployed on is as important as the language of concern [Lieberman and Church, 1992, Edgington *et al.*, 1996]. The reading of a dialog is different than the reading of a newspaper. The applications on unrestricted text domains may introduce several problems for a language that the standard orthography of it does not possess. As an example, although Turkish does not have a vocalisation problem as in Arabic or Hebrew, a synthesizer reading an e-mail, a chat session, or a SMS message in Turkish would most probably need to resolve `slm nbr` as `selam naber` (hello, how are you?).

Another real life problem that TTS systems are confronted with is mixed-linguality [Pfister and Romsdorfer, 2003]. Most texts in a specific language include foreign words. The inclusion of English words into many world languages or the reading of foreign proper names may cause potential errors in synthesis. Such inclusions are rather frequent and must be handled properly requiring additional resources and processing.

In this paper we review the literature on the use of NLP techniques in TTS systems; an overview of all the issues discussed is given in Section 2 with examples.

Section 3 covers topics that include the word level operations required by a TTS system. First we study problems in some languages caused by the writing systems of those languages: segmentation/tokenization in languages like Chinese, Japanese, Thai, and Korean and vocalization in semitic languages.

Next in this section we discuss the role of morphological analysis on speech

synthesis, arguing that morphological decomposition of words is essential in TTS systems for highly inflective, and agglutinative languages for part of speech tagging purposes, word stress assignments, and various other usages in subsequent text analysis. Most TTS systems perform text normalization prior to morphological decomposition. Sproat [1997] argued against this approach stating that the linguistic analysis phases must be taken into consideration also in text normalization as the normalization processes may benefit from linguistic or contextual information. Note that although morphological analysis has many impacts on syntactic/semantic works in prosody generation, it may also be used in tokenization/segmentation problems. For example, Külekci&Özkan [2001] used morphological analysis in word segmentation of Turkish strings, where those processed strings are without word delimiters as being the outputs of a speech recognition engine.

Text normalization and homograph resolutions are reviewed in the third subsection which includes these topics: text normalization of non-standard words such as dates, digits, abbreviations, e.g. , ordinary words that require word sense disambiguations for correct pronunciation generation, and the role of named entity recognition in TTS systems.

A TTS system needs a syntactic analysis for two reasons: correct pronunciation of homographs, and sentence accentuation with phrase boundary assignment [Pfister and Romsdorfer, 2003]. Section 4 covers phrasal and higher level operations in syntax and semantics. Syntactic parsing, phrase detection and discourse dependent topics such as concept-to-speech phenomena are the basic topics of the section.

It must be noted that within this paper phonetic aspects such as grapheme-

to-phoneme conversions are not investigated simply because they are more language independent [Daelemans and van den Bosch, 1997].

2 Why use NLP in TTS synthesis?

Before deeper examining of natural language processing issues in TTS synthesis, a review of the complete process via some examples may provide a better understanding of the subject.

Although the steps in the synthetic generation of speech are more or less common for any language, some introduce problems that are caused by their orthography and writing systems. For those, a certain preprocessing has to be performed. The *tokenization* (or word segmentation) problem of Chinese like languages (Chinese, Japanese, Thai), in which no white space characters are used to separate the words, and the *vocalization* problem of semitic languages (Arabic and Hebrew), where the texts are written essentially only with consonants, require preprocessing.

The tokenization problem, which is actually the determination of the syntactic words² in a text, is not restricted to Chinese type languages, but many others have to perform this process in some manner. In English, one needs to resolve `We're` as `We are` or `hasn't` as `has not`, and obviously this type of occurrences are very frequent. However, that punctuation issue is a very reduced degree of difficulty for English when compared to Chinese type

²[Sproat *et al.*, 1996] describes the orthographic, syntactic and phonetic word discrimination on the example sentence 'I'm going to show up at the ACL' as: it is composed of eight orthographic words that are separated by seven white spaces, and nine syntactic words with the tokenization of ' I'm ' into 'I am', and eleven phonological words if 'ACL' is to be spelled out while reading.

languages. The Chinese sentence 日文章魚怎麼說 may be tokenized as 日文 (Japanese) 章魚 (octopus) 怎麼 (how) 說 (say) ,or 日 (Japan) 文章 (essay) 魚 (fish) 怎麼 (how) 說 (say) where the first parse (How do you say octopus in Japanese) is correct [Sproat *et al.*, 1996]. Another language written without word delimiters is Thai and as an example the string ไปพามเหสี in Thai can be segmented into two different ways: ไป (go) พาม (carry) เหี (deviate) สี (color) ,or ไป (go) เห็น (see) มเหสี (queen) [Tesprasit *et al.*, 2003]. It is clear that the meaningful solution is the second one in this case.

The following examples of vocalization are taken from [Gal, 2002]: The Arabic word كتاب transcribed in Latin characters as ktb may correspond to kitaab (books), or kuttaab (secretaries). Similarly in Hebrew, saphar (to count) and sepher (book), are both written identically with the consonants spr ספר.

Apart from the language of concern, the type of application is also an important phenomena while designing a TTS engine, and the style of the input texts may introduce problems that are not present in the standard ortography of the language. An example of this situation is an SMS³ reader design in Turkish where most people omit the vowels while writing their messages, e.g. they code bugün ben size gelebilirim (I may come to you today) as bgn bn sz glblrm. Although vocalization is not an issue in Turkish, it must be done for a robust SMS message reader in that language.

Morphological analysis is an important issue in TTS synthesis because of two main factors: tagging and word stress assignment. It is especially

³SMS is the 'Short Messaging Service' used in mobile phones, which enables the user to send short text messages up to 160 characters long to others.

not feasible to perform part-of-speech or syntactic tagging without such a component for agglutinative and inflective languages as each word of those has many different derivations and inflections that can not be compiled into a database. Among the wide range usage of morphological analysis, the following examples are given to express the importance of POS and syntactic taggings: The word `convict` has different pronunciations when used as a verb as in `You convict him` or as a noun in `The convict escaped` [Edgington *et al.*, 1996]. A syntactic analysis (and most probably a context sensitive disambiguation) is to be performed on `They read the book` to resolve whether the verb `read` is in present or past tense.

Text normalization is a crucial step while building a TTS system. In real life the input text to a speech synthesizer often contains non-standard words, such as abbreviations, acronyms, dates, numbers etc... These non-standard words have to be converted to a sequence of ordinary words for pronunciation. This mapping includes wordifying the numbers, dates, e-mail and web addresses, acronyms, abbreviations and various characters such as percentages and currency symbols. The discussion of resolving the percent sign (%) in Russian given in [Sproat, 1997] is a good example on the subject that the percent sign maps to different surface forms of word `procent`: with numbers ending in 'one', it is used in nominal form `odin procent` (one percent), with numbers 'two', 'three', and 'four', it is rendered in genitive singular form `procenta dva procenta` (two percent), or it maps to `procentnaja` in adjectival form `dvadcati-procentnaja skidka` (twenty-percent discount), and many other forms exist.

Another example is web address reading in Turkish. Those addresses do

not contain the Turkish characters 'ü', 'ö', 'ç', 'ş', 'ı', and 'ğ'. Thus, the word *hürriyet* is written as *hurriyet* in web address *www.hurriyet.com.tr*, but while reading it is pronounced as in its original form.

Acronyms and abbreviations are also problematic in text normalization process. Some abbreviations introduce ambiguity as they may correspond to more than one word. *Dr.* is such an abbreviation in English, which may either denote *drive* or *doctor*. On the pronunciation of acronyms, the letters of the acronym may be spelled out as in *CRC* (cyclic redundancy check), or wordified as in *AIDS*, or partly wordified and spelled as in *OPEC* (whose reading is spelling 'O' and reading the rest 'PEC' as a word), or maybe all the words referred by the letters are pronounced as in *NY* (New York). Determining how to pronounce them is a serious problem.

Some ordinary words need context sensitive disambiguation for correct reading. As an example the word *row* pronounced differently in *The operations performed row by row* and in *When the police arrived, the row ended* where it means a *queue* in the former and *fight* in the later. Both interpretations are nouns, and so cannot be resolved by POS tagging. A word-sense disambiguation is required to generate the correct pronunciation.

Generating the pronunciation of proper names (or named entities in other words) differs from ordinary words in two ways; first, building a pronunciation lexicon containing all the possible named entities is not possible, and second, if letter to sound rules are proposed to be deployed on the process, those rules are not consistent with the ones compiled for standard words. Although in practice most frequently seen named entities are collected in a pronunciation lexicon, a robust system has to be able to produce good

quality speech for the ones that are not present in that lexicon. Because of these problems the detection of proper names in a text and special processing to generate appropriate soundings for them is an important task for TTS systems. The detection process is not a trivial operation even with the information that proper name initials are capitilized in most languages. As an example, the word `apple` refers to the *Apple Computer Inc.* in the sentence `Apple announced a new advance in computer design` versus it is an ordinary noun in `Apple is a nice fruit`. Note also that the type of the named entity may effect the pronunciation in some languages, such as in Turkish, the word `Aydın` has different primary stress assignments in the sentences `Aydın Ege sahillerine yakındır` (Aydın is near to Aegaen costs) and `Aydın zekidir` (Aydın is intelligent), where the word refers to a city in the former and a person's name in the later.

Attention has to be paid for mixed linguality in a TTS system. The inclusion of foreign words and also foreign proper names occurs very often in most languages. `Swiss Diary` and `World Wide Web` are such inclusions in the example German sentences `Der Konkurs von Swiss Diary` and `Er surft im World Wide Web` [Pfister and Romsdorfer, 2003].

Syntax and semantics greatly influence generating the prosody of a sentence. `I saw the boy in the park with a telescope` may explain different events with different intonations. The observer may have seen the boy who is in the park and carrying a telescope or the observer may have seen the boy ,who is in the park, by a telescope or the observer may be sitting in the park and sees the boy with a telescope. A similar example from [Edgington *et al.*, 1996] emphasizes the phrasing with the help of punctua-

tion symbols in the sentence `My husband, who is 27, has left me` versus `My husband who is 27 has left me`. The first includes an explanation regarding the husband, but the second implies a polygamy according to the appropriate phrasing. Speech synthesizers use the syntactic and semantic constituents in recognition of phonological and intonational phrases both for both a more natural sound and a more accurate rendering of the text.

3 Word Level Issues

3.1 Preprocessing Tasks

3.1.1 Tokenization

The first action to be performed by any application that involves natural language processing is *tokenization* [Webster and Kit, 1992]. Tokenization may be defined as segmentation of an input character string into tokens, which are mainly the words. A very well established formal description of tokenization may be found in [Gou, 1997]. Different perspectives on the notions of *word* and *token* from the point of view of lexicography and pragmatic implementations exist. We do not cover those here; see [Webster and Kit, 1992] for discussions. For sake of simplicity we do not distinguish between word/token and assume that they are interchangeable. As words/tokens are the basic building blocks of a language, the further steps of linguistic processing very much depend on this segmentation. The depth and difficulty of a tokenization process depends on two main subjects, the orthography of the language concerned and the application of interest.

Some languages such as Chinese, Japanese and Thai do not have word delimiters. Although there are not so many as such, more than 20 percent of the world population uses those languages. This brings up the importance of tokenization task and numerous studies have been published on the subject. An international segmentation contest (The First International Chinese Segmentation Bakeoff) has been held in a recent workshop on Chinese language processing [Sproat and Emerson, 2003]. Discussions on that contest and citations to further review articles on Chinese word segmentation may be found in [Sproat and Emerson, 2003].

In most languages the words are delimited by white spaces or punctuation marks. For those, the tokenization task is more related with the type of the application, which may be a machine translation, information retrieval or a TTS system, rather than the characteristics of the language's writing system. Different applications may require different standards [Sproat and Emerson, 2003, Sproat *et al.*, 1996]. If one segments the sentence `You've to keep on working` according to the space delimiter, then `You've` is just one token, where in reality it is composed of two as `You` and `have`. This information is crucial for a speech synthesis system. A Turkish noun phrase `Osman'in kalemi` (Osman's pencil) on the other hand must be resolved `Osman'in` as `osmann`, which is a single token from the TTS point of view. From a machine translation perspective, compound words are very important that such a system needs to mark "keep on" as a single entity. Although those are somewhat mixed with morphology and syntax, they are to emphasize all languages somehow need a segmentation process in a varying degree of difficulty according to the writing system and application of concern.

In tokenization research mainly three approaches exist; purely statistical, purely lexical rule-based and hybrids of the two [Sproat *et al.*, 1996]. Purely statistical methods which rely solely on calculation of probabilities for identifying word boundaries did not gain much interest and it has been indicated that the success of such systems is lower than the purely knowledge-based systems [Webster and Kit, 1992]. Recent works on the topic concentrates on combining knowledge and statistics.

Another point to be decided on is whether the segmenter results a single solution to an input string by using all possible knowledge (morphology, syntax ...) without need for further disambiguation, or the segmenter will detect all possible tokenizations and perform a disambiguation process based on a specified evaluation to choose one of the possibilities [Gou, 1997].

Generally speaking, a tokenization process may be thought of a two phase task. The first phase is the look-up operation from the dictionary where words that form the input string when concatenated one after other are selected. There may be, and most probably will be, more than one such group of words. If so, the second phase is selecting the right word set from the others (disambiguation).

The ambiguities observed may be *conjunctive* or *disjunctive* [Webster and Kit, 1992]. Let the input string to be segmented as XYZ where X, Y, and Z are words from the dictionary. If XY is also a word in the dictionary, then the fragment may be segmented as both X/Y/Z and XY/Z⁴ because the compound word XY is composed of X and Y, where each of them is again a word in the dictionary.

⁴Note that slashes ("/") indicate the word boundaries for the examples given in tokenization subsection.

This is named as *conjunctive* ambiguity.

If the input string is as XsY , where X, Y, Xs , and sY are words in the dictionary and s is a string of length bigger than 1, then the second type of ambiguity arises. The fragment may both be tokenized as Xs/Y and X/sY . That problem is due to the overlapping segment s , which may be the prefix of word sY or the suffix of word Xs . This is called *disjunctive* ambiguity.

It is noteworthy here to give the definitions of *critical point* and *critical fragment* [Gou, 1997]. If character c_p in the character array $c_1 \dots c_p \dots c_n$ is always a word boundary in all the possible segmentations of the string, then this point is called a *critical point*. The first and last characters are also critical points by definition. The fragment between two critical points is named as *critical fragment*. Those critical points are the only unambiguous token boundaries in an input string [Gou, 1997], and the disambiguation process is to be performed on critical fragments. An interesting observation on critical fragments is the *one tokenization per source* [Guo, 1998] that has been stated as: "For any critical fragment from a given source, if one of its tokenization is correct in one occurrence, the same tokenization is also correct in all its other occurrences." Informally this observation means that the disambiguations of an ambiguous fragment appearing in different positions in a given context is most probably the same.

The elementary methods in tokenization are modeled by a single framework in [Webster and Kit, 1992]. This structural model, called as *Automatic Segmentation Model-ASM(d,a,m)*, classifies those methods by their three properties as d for direction of search (right-to-left or left-to-right) for string matching operation, a for addition or omission of characters when words

from the dictionary match with some portion of the input string, and m for using principle of minimum or maximum tokenization. To best understand the model, let us examine the *forward* and *backward maximum tokenization* algorithms. The mathematical descriptions of these algorithms and details of the example given below may be found in [Gou, 1997].

Let an input string be ABCD, and the dictionary L be composed of the words $L=\{ A, B, C, D, AB, BC, CD, ABC, BCD \}$. The forward maximum tokenization searches the input string from left-to-right (d parameter of the ASM if left-to-right). The algorithm always tries to match the maximum length dictionary entry always from the left side, which indicates the m parameter of the system is maximum matching⁵. When maximum length word is matched from the left side, the process repeats with the next position until the end is reached. With the forward maximum method, the sample input string ABCD is tokenized as ABC/D.

The backward maximum tokenization algorithm follows the same procedure of the forward one with the direction reversed. In the backward tokenization algorithm the matching process is from the right end. The same input is resolved as A/BCD with the backward algorithm.

Another well known tokenization is *shortest tokenization*. In the shortest tokenization the segmentation with the minimum number of words is chosen among the others. For example the input ABCBCD is decomposed as ABC/BCD as this segmentation is made up of just two words where the others are of more than two words.

⁵For Chinese minimum matching does not work as nearly all characters are stand alone words in the dictionary.

The backward and forward tokenizations can be modeled with ASM very well but the shortest tokenization cannot [Gou, 1997]. Thus, although the ASM is a good structural model, some methods in the literature exist that do not fit to ASM.

It can be stated that each tokenization system somehow performs a look-up operation to match some part of the input string with words in a dictionary. Obviously, the quality of that dictionary impacts on the performance of the whole system [Sproat *et al.*, 1996]. Many words may have different inflections or derivations according to the morphology and storing all forms of all words in a database is impractical and inefficient. A better way is to construct the tokenization in such a formalism that the rules of the morphology can be integrated. More over, there is a great probability that the system would face out-of-dictionary words such as proper names and foreign words. These unknown-words must also be handled, which is best done with statistics. For the disambiguation process, the n-gram probabilities of words may be used. The system has to be a mixture of statistical and rule-based approaches to be able to accomplish all these. Sproat *et al.* [1996] propose such a system based on *weighted finite state transducers*. The usage of a finite state methodology is another advantage of the system in that it can be easily integrated to other linguistic parts that are also implemented with finite state techniques, e.g. a finite state morphological analyzer.

3.1.2 Vocalization

In semitic languages, such as Hebrew and Arabic, words are mostly written by only consonants and the vowels are omitted in text. The vowels of a word are

defined by the *'pointings'*⁶ of its characters, where those points are missing in the written text. Arabic contains 6 such vowel diacritics, and Hebrew has 12, although in Hebrew many vowels share the same pronunciation [Gal, 2002]. The famous example of word *ktb* in Arabic may be vocalized (or pointed) in different ways, such as *kitaab* (book), *kutub* (books), or *kataba* (to write), and many more alternatives also with consonant spreadings [Beesley, 1998]. Although the native speakers of those languages do not have serious problems in reading, from the computational linguistics perspective, this ambiguity has to be resolved for any NLP system of those languages [Kamir *et al.*, 2002]. The vowel restoration process is a must in TTS systems of semitic languages [Choueka and Neeman, 1995].

Note that although the semitic languages pose this ambiguity by their nature, there may be similar problems in other languages as well. If we think of an SMS reader in Turkish for example, it is quite highly probable to get an input as *mrhb bgn nslsn?*, which should be converted to *'merhaba bugün nasılsın?'* (hello how are you today?) in standard orthography of the language for the correct speech synthesis. The main approaches of that work in Hebrew and Arabic may be investigated in three groups [Kontorovich and Lee, 2001].

The first and the most basic method is just choosing the most frequently seen *diacritized* (all pointings supplied) form of the input word. It is assumed that the necessary statistics are collected from a fully diacritized

⁶The vowel characters in Arabic and Hebrew are generally formed by supplying points around the consonant characters. For example the word ערב in Hebrew has the following three versions with different pointings: עֶרֶב, עָרַב, עֲרַב [Kontorovich and Lee, 2001]. Thus, some literature on the subject use the word *'pointing'* referring to vocalization.

corpus. When the vowels of an unpointed word are to be restored, the most probable pointed version is selected. Note that this method is context-free as it uses just unigram statistics. The success rate for Hebrew is reported as 77% in [Kontorovich and Lee, 2001] , 68% in [Gal, 2002], and 74% is given for Arabic in [Gal, 2002] by this baseline method.

The second method is by using the *morphological analysis* information [Choueka and Neeman, 1995, Gal, 2002]. After finding all possible analyses of a word with the corresponding vocalizations, a context-sensitive disambiguation (with some statistics and syntactic rules) is performed among those and the best fitting form is selected. The *Nakdan-Text* [Choueka and Neeman, 1995] system reports a 95% success for Hebrew. The part-of-speech obtained from morphological analysis may also be used in vocalization process. With the knowledge of the POS tags of the input, the most probable diacritized form of the word with that tag can be selected [Kontorovich and Lee, 2001]. Certainly, this process also needs a training corpus that is fully vocalized and POS-tagged. [Kontorovich and Lee, 2001] gives the result of such a POS based vocalization as 79% on the Westminster POS-tagged corpus.

The third methodology is by using the *Hidden Markov Models*. As HMMs can include the context sensitive information via the chain of probabilities, it serves a good basis for vocalization. The unpointed word list is taken as *observations* in the HMM and the hidden states are assigned to pointed forms of those. Given an observation sequence of vowelless words w_1 to w_n , the corresponding hidden state sequence d_1 to d_n holds the vowel-annotated forms of the list. Using this idea as a starting point, [Gal, 2002] proposed a *bigram* HMM model. This model assumes that a word pointing is dependent

on the previous word. The author reported the success rates of that bigram HMM, 81% in Hebrew and 86% in Arabic. The system achieved 87% accuracy in *phonetic group classification* in Hebrew, meaning that; although the right pointings are not restored for a given word, the restored vowels have the same pronunciation with the correct ones. It is noteworthy that the phonetic group classification is a sufficient metric from the TTS point of view.

A slightly different model is tried in [Kontorovich and Lee, 2001] where there exist 14 hidden states in each layer. An unpointed observation is thought to be coming from one of those 14 states. The reason for using 14 is that the POS based vocalization study reported also in the same paper used 14 POS tags, and to compare the HMM results with that POS-tag based results, they used that number of states. The parameters of the HMM are trained by Baum-Welch algorithm. The success rate obtained is 81%.

3.2 Morphological Analysis

Morphological analysis is the basis of further syntactic and semantic analyses steps in natural language processing. An efficient analyzer is a must, especially for highly inflective and agglutinative languages, where word formations and representing different forms of a word requires complex structures that cannot be expressed with a small number of simple rules. In text to speech synthesis such an analyzer may be used in building pronunciation lexicons, in primary stress assignments, in tokenization/normalization processes, and in part-of-speech tagging. Note that with its usage in POS tagging, the morphological analysis component serves as a basic building block for phrase

boundary detections, which is essential for generating prosody in a sentence.

Pronunciation lexicons are the lexicons used to convert the graphemic representation of a given word into its pronunciation representation in some form [Oflazer and Inkelas, 2003], so that the input word can be realized as a speech signal. The corpus based word lists for use in speech applications of agglutinative languages, such as Turkish, are inadequate as the rich derivational capability and high number of inflections resulting essentially infinite lexicon [Oflazer and Inkelas, 2003]. It is thus a good idea to encapsulate morphological analysis for pronunciation generation. In [Oflazer and Inkelas, 2003], by using finite state techniques, an input Turkish word is decomposed into all possible morphological analyses and the corresponding pronunciations of each are encoded in SAMPA standard. As an example, the input word "karın" is resolved into following parses, each containing both the tags between '+' signs as a result of the morphological analysis and also the pronunciation representations given between '(' , ')' brackets :

1. (ca:"-r)kar+Noun+A3sg(1n)+P2sg+Nom { *your profit* }
2. (ka:"-r)kar+Noun+A3sg(1n)+P2sg+Nom { *your snow* }
3. (ca:"-r)kar+Noun+A3sg+Pnon(1n)+Gen { *of the profit* }
4. (ka:"-r)kar+Noun+A3sg+Pnon(1n)+Gen { *of the snow* }
5. ("ka-r)kar+Verb+Pos(1n)+Imp+A2sg { *mix it!* }

6. (ka-"r1)karı+Noun+A3sg(n)+P2sg+Nom { *your wife* }

7. (ka-"r1n)karın+Noun+A3sg+P+Nom { *belly* }

A TTS engine in Turkish has to select one of those above to read the word `karın`. This action obviously requires a disambiguation process. As morphological decomposition tags are also included in the results, a morphological disambiguation can be performed to choose the best for a given context. Note that, of the seven different results above, 1–3 (`ca:r"1n`) and 2–4 (`ka:r"1n`) pairs have the same reading although the morphological analyses of each differ. Thus, for a speech synthesizer, contrary to the morphological disambiguation process between 7 items, the selection is among just five as choosing one of 1–3, 2–4, 5, 6, or 7 is enough for correct synthesis. Generally, the disambiguation for speech synthesis is a little bit easier than a full morphological disambiguation.

A complete linguistic analysis for an Italian text-to-speech system [Ferri and nad Donatella Sanzo] is a good example of the morphology usage for word pronunciation generation. The system has a three phase structure: morphologic analysis, phonetic transcription, and morpho-syntactic parsing. The morphological analyzer obtains the morphologic and syntactic features of each word in a given text. Based on this information, the phonetic transcription level marks the stressed syllables and performs grapheme to phoneme conversion. The syntactically related words are grouped so as to generate intonations of prosody at the last morpho-syntactic analysis level. Morphology lies at the heart of the system as both phonetic and syntactic phases depend on the morphological analysis of the first stage.

Despite the somewhat mandatory usage of morphological decomposition in agglutinative languages, other languages, such as English, may also benefit from morphology. An English pronunciation lexicon for speech synthesis is designed in [Fitt, 2001] which includes morphological breakdowns in the lexicon and addresses the advantages of that morphological annotation. As an historical remark, the *MITalk* English synthesizer [Allen *et al.*, 1987] involved a module, *Decomp*, for morphological decomposition of running text. The *Decomp* model parsed an input word into morphemes of three types as prefixes, roots, and suffixes. Each type had also subcategories; the suffixes and prefixes was classified into three levels, and the morphological decomposition by using those levels aimed to assign the correct primary stress for the given word [Church, 1986]. More detailed explanations on that historical approach for stress assignment using morphological analyses can be found in [Church, 1985] and [Church, 1986].

Morphological analysis components may also be used on text normalization processes such as homograph resolutions and in POS taggings. The usages will be reviewed in the next sections where related. Note that it may also be used in word segmentation algorithms. A Turkish word segmentation technique using a morphological analyzer is given in [Külekci and Özkan, 2001], which takes a sequence of words concatenated one after other without word delimiters as input, and detects all the possible segmentations by using a morphological analyzer.

3.3 Text Normalization and Homograph Disambiguation

3.3.1 Non-Standard Word Resolution

In most genres of text exist many words that are not ordinary. These *non-standard words* (NSW) can be classified into dates, currencies, numbers, Roman numerals, fractions, abbreviations, e-mail, and web addresses. For text-to-speech systems, NSW introduce some problems in that the generation of their pronunciations differs from standard words greatly [Sproat *et al.*, 2001]. Most of the time a synthesizer needs to map a non-standard word to a sequence of standard words for a correct reading which is called text normalization [Sproat *et al.*, 2001, Olinsky and Black, 2000]. Another problem arises on this conversion process in that the NSW have a higher tendency to be ambiguous than ordinary words [Sproat *et al.*, 2001] which cause various problems. Some of the ambiguities and pronunciation difficulties are as follows:

Date and Times: Dates and times may be expressed in many formats. These formats change from language to language, and most probably each language has more than one style of writing those stamps. 01.10.1999, 1/10/99, 01-10-1999, 1-Nov-1999, 1st November 1999 are some of the valid writings of a date. After normalization they are expected to be read as `first of November nineteen ninety nine`. Note that the format of the date stamp may change due to the language. For example in American English the format is 'Month-Day-Year' where in British English the sequence is 'Day-Month-Year'. The same date may correspond to `tenth of January`

nineteen ninety nine in an American English synthesizer [Edgington *et al.*, 1996]. An ambiguity may occur if the year is omitted. 1/2 may be a date (first of february) or a fraction (one half).

Currencies: A currency token contains a currency symbol at the beginning or at the end, e.g \$10 , 1000TL. First the amount is read, and then the currency is pronounced in general as in 10 dollars. There may be some abbreviations such as \$10K which has to be resolved as ten thousand dollars or some exceptions such as \$10 billion' which must be normalized as not ten dollars billion but ten billion dollars.

Numbers, Roman Numerals, Fractions: The numbers, Roman numerals, and fractions are very frequently seen in texts. Context has an important role in resolution of those. 1986 may be read nineteen eighty six as a year quantifier or one thousand nine hundred eighty six as a cardinal number depending upon the context. Another frequent ambiguity is about the Roman numerals. The I token may have to be mapped to one, to first or to the first singular person preposition I. Fractions may have many different readings. The ratio 1/4 has three pronunciations in Turkish: çeyrek, bir bölü dört, and dörtte bir. Although the meaning does not change much, the correct selection according to the context is an asset for a TTS system in that language.

Abbreviations and Acronyms: One of the most difficult tasks in a TTS system is the resolution of abbreviations and acronyms. Dr. in English may be drive or doctor, St. may be saint or street. An interesting (and maybe exaggerated) example of abbreviations is He tried to walk on the Sun. Howard died. [Edgington *et al.*, 1996]. It is not clear that Howard

died because of an accident happened on the sunday, or Howard died because of he walked on the sun of our solar system. Acronyms on the other hand are spelled, as in IBM, or read as it is an ordinary word, as in NASA in English [Sproat *et al.*, 2001, de Mareuil and Floricic, 2001]. Note that for correct pronunciation of acronyms, the first thing is to detect them in a given text. For the recognition of acronyms in a free text, [Taghva and Gilbert, 1999] introduces an automatic method based on inexact pattern matching algorithm. While reading as a normal word, special care must be taken that the standard pronunciation rules may fail for acronyms. For example, in Turkish AIDS is read as it is in English and does not obey Turkish pronunciation rules, and RTÜK is read as if the first letter is spelled and the rest is pronounced as a word. Boula [2001] argued in his work that the rules of lexical stress assignment may change for acronyms on the pronunciation of Italian and French acronyms.

Among those well known examples of token types in a text that need normalization, are some more miscleanous cases in real life. The absence of Turkish characters in e-mail addresses and URLs causes problems in the readings of those. As an example, the `cozumholding` in the web site name `www.cozumholding.com.tr` is pronounced as if it is `çözüm holding`. Non standard punctuations (`**White** man can't jump`) and some humorous spellings (`goooooooood morning`) for emphasis are the other sources of problems in text normalization process.

[Sproat *et al.*, 2001] is an excellent study on normalization of non-standard words. The proposed methodology involves a step by step procedure. The first task is the tokenization of the input text and detection of NSW's. If a

word could not be retrieved by a simple lexicon look up, then it is marked as a NSW. Note that although this works for English, for agglutinative and highly inflective languages such as Turkish or Finnish, a morphological analysis is required for the decision as those languages' have infinite number of word forms theoretically. During the recognition of non-standard words, one can also benefit from the dictionaries of common abbreviations or similar lists along with some hand-crafted rules.

After the detection of NSWs, some of them may need further splitting down for correct expansion, e.g. `Win2K` has to be split into `Win` and `2K` tokens for further steps. When such a splitting occurs, it must be remembered that these tokens are grouped together [Edgington *et al.*, 1996], as these groupings are important in phrase detection while generating the prosody.

The most important part of the system is the classification of the extracted NSW tokens, by which appropriate tags indicating the type of the NSW are assigned to each. Special care has to be taken while deciding the taxonomy of the NSW's. That classification must be general enough to cover all possible cases. [Sproat *et al.*, 2001] defines three main categories for NSW's as purely of alphabetical characters, containing digits or numerals, and miscellaneous ones. Each category has several subcategories. For example, the first category has an `EXPN` subtype which means that the NSW of that type should be expanded for correct reading, such as `gov't` be mapped to `government` and `N.Y.` to `New York`. The other subcategories of first are `LSEQ`, which means to read the NSW of that type as letter sequence (`IBM`), and `ASWD`, which dictates a reading as an ordinary word (`CAT`). After tokenization and splitting steps, each NSW token is put in one of those

predefined classes. This classification scheme has been achieved via decision trees with the domain dependent and independent features extracted in the study of [Sproat *et al.*, 2001].

The tagged tokens are fed into the tag expansion step where the necessary normalization process is performed on each. This is the module where the mapping of NSW's to standard words (*normalization*) is achieved. As all the words are ordinary after this step, the synthesizer can produce pronunciations.

There is one more problem to be solved: some tokens are ambiguous, because more than one tag may be appropriate or more than one expansion may be possible. Remember the examples that 11/2 may be **e**leven **o**ver **t**wo as a fraction or **e**leventh **F**ebruary as a date. **St.** may be **s**aint or **s**treet. Those ambiguities are resolved by a language model which constitutes the last step of the system. This language model is based on n-grams, which is a way of disambiguation in local context.

The technique proposed by [Sproat *et al.*, 2001] is especially designed on English. The generality of the system is tested by deploying it on both Chinese and Japanese, which show quite different characteristics with the absence of word delimiters and high frequency of homographs observed in most texts [Olinsky and Black, 2000]. [Olinsky and Black, 2000] concluded that the system works also for these languages and so is a good basis of further studies on other languages as well.

3.3.2 Ordinary Words Requiring Sense Disambiguation

Apart from the difficulties faced with the pronunciations of NSW's discussed in the previous section, some ordinary words need word sense disambiguation for correct reading. This type of words, written identically, read differently, may be classified into two cases according to their parts of speech tagging [Yarowsky, 1997].

The first case is of those words having different POS tags, e.g. the word `lives` has different pronunciations in sentences `Three lives were lost` versus `He lives in England`, where the word is a plural noun in the former and a verb in the second.

The second case is when the words have the same part of speech tag and so the correct reading can only be achieved by semantic evidences. Note the word `bass` in the following sentences `John was playing bass guitar` vs. `New places for bass fishing can be found from internet`.

Generally, n-gram taggers, bayesian classifiers, decision trees, and hybrid methods of those have been used in disambiguation [Yarowsky, 1997]. N-gram taggers use the sequence of POS tags around the target word and decides according to the n-gram statistics of POS tags previously collected. That is, to choose the best tag for the word to be disambiguated. The technique is not capable of using long distance word associations, which is beneficial for the semantic disambiguation. As an example, the existence of words like `guitar`, `violin`, `percussion` is a very important information while disambiguating `bass`. Thus, although this method is efficient in the first case situations (different part-of-speech), it does not work for the second

case (same part-of-speech), obviously.

Bayesian classifiers, on the other hand have the ability to capture semantic evidences. Such classifiers begin by collecting a specific number of (e.g. 100) words near the target word. The order of the sequence is not considered. The total word association probabilities of those in the collection determines which pronunciation to be used. This approach does not benefit from local context information and also suffers from data sparseness problem.

Decision trees are complex rule based systems that perform disambiguation according to the features extracted. These features are basically orthographic, syntactic, and lexical properties [Hearst, 1991]. The disadvantages of using decision trees are the large parameter spaces and difficulties in construction of the decision rules. [Yarowsky, 1997] proposes a hybrid method with decision trees. The system depends on measurement of collocations around the target word whose pronunciation is ambiguous. After the recognition of these phrases, the likelihood ratios are computed. That is mainly the probability of one possible pronunciation of the word given the existence of a collocation. The likelihoods are sorted into a decision list from where different models such as neural nets or bayesian classifiers can be constructed for the final decision.

The discourse and collocations near a target word are strong evidences to be used in sense disambiguation methods by the *one-sense per discourse* [Gale *et al.*, 1992] and *one-sense per collocation* [Yarowsky, 1993] properties of natural languages. These properties are summarized in [Yarowsky, 1995] as:

- One-Sense per Collocation: Nearby words provide strong and consistent

clues to the sense of a target word, conditional on relative distance, order and syntactic relationship.

- One-Sense per Discourse: The sense of a target word is highly consistent within any given document.

Although it is not specific for TTS systems, [Yarowsky, 1995] proposed an unsupervised learning algorithm based on these two properties. The algorithm begins with a small number of training seeds given for a polysemous word. For each sense of the target word, some samples are labeled in several texts. These are used as training seeds and by performing a classification scheme based on decision lists, the unlabeled occurrences are added to the set. This way the system learns by itself to differentiate between different senses of the target word. An accuracy rate of 96% with that bootstrapping procedure is reported.

Another recent work that benefits from both discourse and collocations on context sensitive homograph disambiguation for text-to-speech systems is proposed by [Tesprasit *et al.*, 2003] for Thai, which is also a language without word delimiters. The system uses two types of features; presence of *context words* within +/-10 neighborhood of the target and *collocations* that are up to 2 contiguous words around the target word. The technique is based on Winnow, which is a neuron-like network algorithm.

3.3.3 Named Entity Recognition

The words identifying special entities such as persons, organizations, locations are very frequently seen in texts. In a TTS system these named enti-

ties cause problems in: word pronunciation generation and phrase boundary detection for intonation⁷. Generating the correct pronunciations of those proper names is a great challenge [Jannedy and Möbius, 1997] because of the following reasons:

- The number of *named entities* are so large that it is not feasible to create a pronunciation lexicon that includes all. Although storing some of the most frequent ones in a dictionary seems a good idea, a robust system requires both recognition of known names and discovery of new names [Wacholder *et al.*, 1997].
- If we consider also the multilinguality of proper names, the situation is worse. Some of the foreign names has to be read in their original language pronunciation if they have not been linguistically assimilated in the phonological system. For example the reading of McDonald's in a Turkish TTS system has to be identical as in an English synthesizer.
- The grapheme-to-phoneme rules usually do not work for named entities. One may think to use the letter-to-sound rules of the language for named entities also, but most of the time the readings of proper names contradict with common phonological rules and the same grapheme strings may correspond to alternative pronunciations in different names (*idiosyncrasy*).
- The lack of morphological analysis for proper names is also a source of problem as morphology serves a good basis for tagging and primary

⁷It worths to note that the problems (and solutions to problems) of named entities heavily depends on the language of concern, meaning that the importance of the explanations may differ from language to language.

stress assignment processes explained in the previous section. If the morphological analyzer is not capable of decomposing the word, then it may have to undergo syllabification for pronunciation generation, where the grapheme-to-phoneme rules may not be appropriate for the correct reading. Note also that the semantic category of the proper name may be important in correct pronunciation generation. For example the Turkish word 'Aydın' may refer two named entities. First, it may mean the city in Turkey as in the sentence *Aydın güzel bir ilimizdir* (Aydın is a nice city.) and second a person name as in *Aydın dün okula gelmedi* (Aydın did not come to school yesterday). The stress assignments are different in both.

The second problem of named entities in speech synthesis is that the *structural ambiguity* [Wacholder *et al.*, 1997] introduced by the proper names may cause errors in phrase boundary detection, and as a consequence of that, the intonational patterns in a sentence may not be parsed correctly ⁸.

The structural ambiguities of named entities may be classified into three categories with respect to *prepositional phrase attachments*, *conjoined phrases*, and *possessive pronouns* [Wacholder *et al.*, 1997].

- Some of the proper names are formed in a way that two noun phrases are attached via a preposition (e.g. 'at', 'in', 'of'). 'Midwest Center for Computer Research', 'City University of New York', 'The Museum of Modern Art in New York City' may be given as examples of such. Although those named entities are made up of the

⁸Phrase and intonational boundary detections are discussed in section 4.

whole (all noun phrases plus all the prepositions), they may be divided into more phrases erroneously. For example, [The Museum of Modern Art in New York City] is a whole phrase referring a single entity, but it is hard to detect that it is not of the combination of three phrases, as [The Museum] of [Modern Art] in [New York City].

- Conjunctions of proper name phrases is another source of ambiguity. In the phrase Barnes and Noble bookstore, we know that Barnes and Noble is a single named entity and the phrase does not indicate two distinct companies as Barnes and Noble. Contrary to that, in the Xerox and Bell laboratories phrase, it is well known that Xerox and Bell are distinct corporations. A TTS system has to read [Barnes and Noble] as a whole phrase, where [Xerox] and [Bell] should be divided into two.
- The existence of possessive pronoun in a named entity has two meanings. First, it may be a relation between two as in India's Gandhi. Second, it may belong to the single proper name as in Alzheimer's Association. In the first case the items are disjoint ([India's] [Gandhi]), but in the second they form a single component ([Alzheimer's Association]).

Practically it seems best to have a dictionary of named entities along with their pronunciations while designing a full TTS system. If such a named entity dictionary exists, a recognizer of proper names must be able to benefit from that, but a mechanism to discover the new named items, which are missing in the lexicon, must also be implemented for a robust system.

Although the recognition and discovery of proper names in a text depend on the writing system of the language, generally the characteristics used in resolutions of named entities may be investigated in two parts as features coming from the word internal structure and features related with the context [Zhou and Su, 2002, Bikel *et al.*, 1997, Colins and Singer, 1999].

Word internal features are of those especially related with spelling of the word. In most languages the initials of proper names are capitalized. Note that sentence beginnings are also in capitals. Thus an ambiguity arises whether the first word of a sentence is a named entity or not. It is more confusing in some languages (such as English) if the named entity is the second word in the sentence where the first word is something like an adverb, a pronoun, or a preposition [Wacholder *et al.*, 1997]. For example in the sentence `New Zealand is near to Australia` it is hard to decide for a computer program whether the proper name is `New Zealand` or only `Zealand` only by examining only the initial letters of words. The different word internal evidences, other than capital letters inside a named entity (*NameFinder*), may be stated as the inclusion of numbers (`Win2K`), punctuations (`N.Y.`), or special symbols (`AT&T`). Those evidences may be enlarged due to the structure of the language, or miserably may have no effect in named entity identification such as for languages of Chinese type.

Word-external or contextual features are found in the words near the proper names. A noun following a `Mr.`, `St.`, or followed by a `Ltd.` is most likely a proper name. The context sensitivity may be enlarged by looking up more distant words to the left or right around a named entity candidate, which is often a case to test especially for statistical recognition approaches

[Lin and Hung, 2002]. Another word external feature may be questioning whether the candidate word has been previously labeled as a named entity.

With those types of features, the NER problem has been studied mainly in one of two ways: by hand-crafted rule based systems and by statistical methods. The named entity extraction tool, *Nominator*, of IBM is an example for rule based systems [Ravin and Wacholder, 1996, Wacholder *et al.*, 1997]. *Nominator* does not benefit from any statistics or taggings and is based purely on heuristics. Following a proper name candidate search in a given text, it performs a splitting operation if a candidate is a multi-word token for the purpose of determining whether the conjunctions are within the named entity or not as described above. After then it groups the detected names referring to same item (for example JFK and John F. Kennedy are put into same group). Last a very important categorization is performed to classify if a named entity is a person, organization, or location. Note that from TTS point of view this is extremely important for some languages (see earlier examples of Aydın pronunciation in Turkish).

Similar to rule based systems, a finite state approach is given in [Jannedy and Möbius, 1997] for the pronunciation of proper names in German. The system is tested on street names of German where they mostly include **-dach**, **-stein**, **-hecke**, **-allee**, or **-platz**. The idea is to parse the given word using a finite state transducer searching these parts in the word. In case of success, the pronunciation is generated according to the predefined rules. If a failure occurs, the syllabification is deployed and the grapheme-to-phoneme conversion is performed. Note that special treatment has to be done in letter-to-sound conversion of proper names, and it is not the same model used in normal

words of the language [Chung *et al.*, 2003].

The rule-based systems are hard to maintain and scale up. For each language, or maybe for each different domain, the rules have to be compiled differently [Lin and Hung, 2002]. The rules governing texts about drugs may not be the same as the rules compiled for texts about finance. Along the rapid development and robust structure of hand-crafted methods, the statistical methods are preferred more recently. Note that the statistical approaches require the calculation of some parameters for the models. These parameter estimations may be done in various ways including expectation maximization type bootstrappings or boostings [Lin and Hung, 2002, Cucerzan and Yarowsky, 1999, Collins, 2002].

One of the reasons for the wide usage of the statistical methods is that: in some languages the features explained above are absent. In Chinese there is no white space word delimiter and no capitalization or similar evidences occurring in a given text. Thus, the Chinese named entity recognition problem is hard [Ye *et al.*, 2002, Lin and Hung, 2002]. A probabilistic verification method was proposed by [Lin and Hung, 2002] where the hypotheses of whether a candidate is proper or not is tested with a confidence measure. That confidence measure is defined to be a model looking at the right and left contexts of the candidate. If the returned value is above a threshold the candidate is assumed to be a proper name. A similar greedy strategy is also used in [Ye *et al.*, 2002] with a rationality model for named entity recognition in Chinese and both systems report success rates about 90%.

Machine learning based methods are also used in NER. An unsupervised classification scheme was proposed in [Collins, 2002] with an EM-style boost-

ing and decision lists. With limited number of seed rules, the system is reported to be capable of learning the characteristics of proper names. A language independent system was supposed in [Cucerzan and Yarowsky, 1999] which combines morphological and contextual evidences. Again a bootstrapping is performed with an initial training list of known proper names which are no more than a hundred items. The ranking algorithms can also be used for NER [Collins, 2002].

Among the numerous methods used in NER, the most compromising results are reported by the systems based on hidden markov models [Bikel *et al.*, 1997, Zhou and Su, 2002]. The *Nymble* [Bikel *et al.*, 1997] name finder treats each word as an n-tuple of features described above and calculates the required probabilities by just a simple counting on a previously labeled training set. The system puts each given word into one of seven proper name classes or into not-a-name class. 93% success was reported for Nymble in English. A more sophisticated HMM model was proposed in [Zhou and Su, 2002] which uses both deeper internal and external features of words. The success obtained is given as high as 96.6%.

4 Phrasing for Prosody Generation

For some years, the TTS systems are now able to generate highly intelligible synthetic speech from unedited text input [Nooteboom, 1997], but they have some deficiencies in naturalness [Beutnagel *et al.*, 1999]. As the researchers aim to build synthesizers that produce speech close to humans' speaking as much as possible, more attention has to be paid for *prosody* generation. The

prosody in a humans' reading depends mainly on two characteristics. The first one is the emotional situation of the reader while reading (for example if he is sad, angry, bored etc.) and the feelings that the read item gives to the reader (e.g. the subject may make the reader angry, sad, happy, etc.). Although it is possible to tune the parameters to make the machine read a text reflecting a given feeling, extracting those feelings or determining how the reader feels about what is being read from raw text is hard. Thus, this type of the prosody generation is more related with the behavioural investigation of humans which is not discussed in this study.

The second type of prosody generation is dependent on syntax, semantics, and maybe pragmatics of the text. Many reviews state that earlier studies generally believed that syntax information is sufficient for prosody ,but later understood that the syntactic structure only provides a good basis for prosodic structure, and the effect of the semantic and discourse also have great impact [Bachenko and Fitzpatrick, 1990, Wang and Hirschberg, 1991, Lindström *et al.*, 1996]. That is, syntax is necessary but not sufficient for prosody generation. Note that the prosody generation for more natural sounding needs higher levels of linguistic information.

In theory, the prosodic structure of a given utterance is a hierarchy of levels, from low to high : *syllable*, *foot*, *prosodic word*, *clitic group*, *phonological phrase*, *intonational phrase*, and *phonological utterance* [Nooteboom, 1997]. A foot is a lexical word which is made up of several syllables. If some of the consecutive foots in a sentence are pronounced together, they form up a single prosodic word. A prosodic word may also be equal to just one foot if the pronunciation of that foot is not related with its neighbors. For example,

in the sentence I am going to the school the foots I and am build up a prosodic (or phonetic) word and going is also a stand alone prosodic word and foot. A clitic group is a prosodic word plus the clitics to its right or left. In the previous example sentence the (clitic) plus school (prosodic word) is a clitic group. The phonological phrases are the concatenation of prosodic words (or clitic groups, if there exists any) and similarly intonational phrases are the union of those phonological phrases. The highest level is the phonological utterance which is the sentence that is the combination of intonational phrases.

In practice, the prosody generation process of a sentence begins with the words in it. Each word in the utterance has primary and maybe a secondary lexical stress in its syllables⁹. When a word is intonationally prominent than others, as is the case in natural speech, the primary stress assigned syllable is accented and these words are said to bear *pitch accent* [Hirschberg, 1993]. Although each word has a stress assigned syllable with the morphological analysis, the word is not accented all the time. When *deaccented*, no significant difference is observed between the syllables of the word. For example the word *object* has stress on its first syllable when used as a verb and in I *object* your plan it is accented, but in They *object* it too it is not [Hirschberg, 1993]. Additionally, a deaccented word may or maynot be *cliticized*. When cliticized, the word is pronounced with its adjacent word as a one phonological word. As an example *to* is in cliticized form in *wanna* and not cliticized in *want to*.

⁹The lexical stress assignment process is a word level issue related with the morphological analysis and investigated in section 3 within this paper.

The question of what makes a word to be accented or deaccented is not totally answered yet, but as mentioned before, syntactic, semantic and pragmatic properties are believed to effect accentation. [Hirschberg, 1993] argued that surface order or some type of information status may be used in prediction of pitch accents. As an example for surface order, in the Turkish sentence *Bugün ben Ankara'ya gidiyorum* (I am going to Ankara today), *Ankara'ya* is accented, but if it is said in a different word order as *Ben Ankara'ya bugün gidiyorum*, *bugün* is accented. Many different information status phenomena may be defined, which are effecting accentation, such as focus, contrastiveness, given/new distinction etc. For example *given/new* distinction states if a content word in the sentence is introduced for the first time in the discourse (meaning 'new') or has been observed previously (meaning 'given'). If new, the word is accented, if not, it is deaccented, e.g in sentence *There are lawyers and there are good lawyers* the first lawyer is accented, but the second is not.

Most text-to-speech systems perform accentation based on content word/function word distinction. This approach seems easy and adequate for some right headed languages such as English but is not suitable for some languages such as Turkish. [Lieberman and Church, 1992] names this approach as *chinks 'n chunks* algorithm by calling function words as *chinks* and content words as *chunks*. The algorithm assumes that any phrase in a given sentence is of the form (chink*)(chunk). A phrase consists of all function words to the left of a content word. The parse of an example sentence with this algorithm is as [There are several important changes] [in the way] [the quantifier rules] [will work] [for the remainder] [of the course].

Note that a POS tagger is needed for the algorithm, which at least labels the words as chunks or chunks. On the other hand [Hirschberg, 1993] states that especially in synthesis of longer texts, this approach is problematic also for English. She argues that POS based parsing is not effective in complex nominals such that `city hall` has the stress on the second word, `tax office` has on the first, and `city hall tax office` has on first and third, where all these phrases are noun-noun phrases. Those type of word groups are frequent in English, and special processing for each type is costly. If someone wants to use chunks 'n chunks type algorithm, some rules to handle exceptions (or modifications) are necessary.

Detection of phonological (or prosodic in other word) phrases in a sentence is very crucial in prosody generation of a sentence. The syntactic parse of the utterance is assumed as a required information generally [Black and Taylor, 1994a, Bachenko and Fitzpatrick, 1990]. In their excellent study, Bachenko&Fitzpatrick [1990] state that syntax (syntactic categories of words and syntactic gaps between phrases) effects *phonetic quality*, *word prosody*, *phrasal stress*, and *segment quality*. The first two of those are investigated previously in this paper. For the remaining two, segment quality and phrasal stress, the following examples are given: For phrasal stress effect, `power units` has stress on `units` in the sentence `If house current fails, power units from battery`, where it has stress on `power` in `The power units failed`. The former is a verb-noun sequence and the later is a noun-noun sequence. For the segment quality, the word `to` is pronounced weak in `We spoke to John`, but in `Who did you speak to?`, it is read strongly because of the gap associated with question word `who`. Bachenko&Fitzpatrick [1990] begin with

finding the phonological words at the first step for the detection of phonological phrases. At the second stage the formation of phonological phrases is performed via some rules that take syntactic constituents into account. Basically, a syntactic head (verb, noun, adjective, adverb) and the material that intervenes between it and a preceding head is assumed to be a phonological phrase. After this finding of phonological phrase boundaries, some salience rules are applied to get which of them are prominent. This maybe viewed as the detection of intonational phrases in the theoretical structure discussed in the beginning of the section.

As *head* is central in the formation of phonological phrases in the study of [Bachenko and Fitzpatrick, 1990], Lindström *et al.* [1996] proposed to use dependency graphs which are of the form head and modifiers and so seem appropriate more than the syntax trees. The idea is deployed on a Swedish text-to-speech system, where the output of a morphosyntactic component is used to build dependency graph of utterances. The feasibility of the system is demonstrated by comparing the results with the human read sentences and it seems appropriate to use also dependency graphs in prosody generation.

There is not a direct mapping from syntactic parse of a sentence to its prosodic phrasing. A famous example on this is This is [the cat that caught] [the rat that stole] [the cheese]. The square brackets mark off the noun phrase constituents detected by syntactic analysis. This sentence is better synthesized as [This is the cat] [that caught the rat] [that stole the cheese]. Thus, although syntax is a good starting point, it is not always the true solution. In the same study, Bachenko&Fitzpatrick [1990] argued that the syntax/prosody misalignments maybe a consequence

of semantic considerations and readjustment rules, a point which take those considerations into account, are necessary for correct phonological parses.

Black&Taylor [1994a] propose a four step algorithm for the generation of prosody for an input sentence where each word in the given utterance has a POS tag, the syntactic constituent structure is given and also an *act* (yes/no question, statement, greeting etc.) is defined for the sentence. At the first step, the prosodic phrase boundaries are detected with a similar work of [Bachenko and Fitzpatrick, 1990], and pitch accent informations (accented, deaccented, etc.) for each word are assigned according to the work of [Hirschberg, 1993]. After converting the informations of those two steps into a prosody tree in third step, the system uses the sentence act for the assignment of intonational boundaries. The rules related with the act of the sentence are deployed, and the resulting scheme defines the intonational phrases. By the use of that act information, different readings of the same sentence are possible, which is the case in real life. It is noteworthy here to state that practical examples of prosody generation for Italian are given in [Ferri and nad Donatella Sanzone, 1997] and a detailed work on generation of intonation in Swedish is defined in [Horne and Filipson, 1997].

Although the phonological phrases can be extracted from syntax somehow, the intonational phrases are more difficult to detect [Atterer and Klein, 2002]. [Atterer and Klein, 2002] uses some form of chunk 'n chunk algorithm for the detection of prosodic phrases in a German TTS system where a restructuring process of the chunks are performed via some rules. After the assignment of the prosodic phrases, intonational phrases are formed by first inserting breaks according to punctuation symbols. With the idea that the utterances

are generally divided into equal length intonational phrases, further breaks are obtained by length and balancing constraints.

Different from the heuristics to specify intonations, Prevost&Steedman [1994] proposed an *information structure* which aims to synthesize speech from semantics and discourse of the text. Later, he had also used this approach for spoken language generation [Prevost, 1996]. These type of synthesizers are named as concept-to-speech systems (CTT) to distinguish them from TTS systems which rely mostly on syntactic and ortographic information. It must be noted here that CTS systems also benefit from ortography and syntax, but build the main pronunciation scheme based on semantics. The proposed information structure is of two level. In the first part the utterances are divided into semantic propositions rather than syntactic constituents. Those propositions are either *themes* or *rhemes*. Basicaly, theme is the *topic* and rheme is the *comment*. In a wider sense, theme is the link by which the utterance is connected to previous sentences, and rheme is the core contribution of the utterance to the context [Prevost, 1996]. In the second part after the identification of themes and rhemes (which are actually the intonational phrases), the words that are bearing pitch accents in those segments are to be detected. This detection process is the searching of emphasis among the words in every phrase. The given/new distinction and contrastiveness are used to recognize the focus words.

For a better understanding of the concept, the following examples are taken from [Prevost, 1996]: In the sentence, [The BRITISH amplifier produces] [CLEAN treble.], the first square brackets enclose the theme, and the second marks the rheme of the sentence. The words written with upper case letters are the

focus words to be accented. This sentence likely to be the answer of the question 'What does the British amplifier produce?'. The same sentence may be parsed as [The BRITISH amplifier] [produces CLEAN treble.] ,where the first part is now the rheme, and the second part is the theme, if it is the answer of the question What produces a clean treble?. As can be seen from this example, a single utterance may be segmented into different themes and rhemes depending on the discourse. Also note that complex sentences may have more than one theme/rheme segmentations.

Rule based heuristics are used for the identification of themes and rhemes which is actually a difficult problem [Hiyakumoto *et al.*, 1997]. Hiyakumoto *et al.* [1997] propose a two level operation for this purpose. The algorithm begins by segmenting an input utterance into propositional constituents that are centered around verbs, adverbs, and prepositions (by the assumption that a POS tagging is initially provided). Each proposition should have only one verb complex. The punctuation symbols are also considered in this step. After the construction of those propositions, each proposition is subdivided into theme and rheme with some predetermined rules that are related with surface order or semantic interpretation of the verb constituent.

For the inspection of focus in themes/rhemes, the givenness and contrastiveness metrics are used in [Hiyakumoto *et al.*, 1997]. Hiyakumoto *et al.* [1997] benefits from a lexical database WordNet, where semantic relationships are encoded within. In that database each word is assigned to one of the four category; noun, verb, adverb, and adjective. Each category has some number of synonym sets that are organized around semantic relations. As an example the nouns are connected with 'is-a', 'part-of', 'member-of',

and 'made-of' relations.¹⁰ The authors give well defined distinct algorithms for givenness identification and also for contrastive stress assignment. With these procedures the focus items are detected in each theme/rheme segment. Besides the fact that the system is just deployed on a small test set, the results are given to be encouraging for more natural sounding.

Intonational phrases not only effect naturalnes in speech, but also has a great role in the meaning of a sentence sometimes. Wang&Hirschberg [1991] discuss on the sample sentence **Bill doesn't drink because he's unhappy** to emphasize this. If uttered as a single intonational phrase, the sentence means Bill does indeed drink, but the cause of his drinking is not his unhappiness. If it is read as two phrases; the interpretation changes that Bill does not drink and the reason for this is his unhappiness. In the same study, [Wang and Hirschberg, 1991] proposes a classification and regression tree (CART) analysis for the prediction of intonational phrasing from text. Based on some features, which include syntactic constituents and some other variables extracted from speech database of the corpus worked on, the system is trained, and 90% success is reported on the experimental test set. One of the main goals of the study in using CART analysis was to discover which features are relevant in automatic intonational phrase boundary detection.

Similarly, Sonntag&Portele [1997] have investigated the linguisting concepts effecting the prosody of spoken utterances. It is argued that the prosodic function of a sentence has to be seperated into two as purely prosodic and segmental influences. The segmental influences are mainly the pauses occuring in speech which generally mark the intonational phrase boundaries.

¹⁰For detailed information on WordNet, please refer to www.globalwordnet.org.

The authors aimed to investigate the factors that are independent of these pauses, and for this purpose the segmental information is removed from the test signals in the experiments of that study. In this way, although the intelligibility of the speech signal is reduced, all the information that is carried by prosody is made observable. Emotions, syntactic structure, dialogue acts (whether the sentence is an affirmation, negation, suggestion, or request) and given/new distinction are discussed as the main variables of prosody. Perception tests are performed to measure whether these variables can be detected from the signal. As an example, the following test is performed to investigate the effect of syntactic structure on pronunciation of a sentence: The subjects of the test hear of a speech signal which is the stimuli of a sentence whose segmental information is destroyed. The listeners are then asked to assign one of the given syntactic structures for the sentence heard. Approximately 70% times the listeners select the correct structure. Another test is made to measure the effect of dialogue act in prosody, where the listeners are asked of the act of a heard speech. For the details of the tests please refer to the original paper.

Another influence of accent in a sentence is the *anaphora* resolution. An anaphora is a reference to an item whose ambiguity is hard to resolve without discourse. Piwek [1997] discusses this subject an following example is taken from his work : `John fed the animals. The cats were hungary.` If an accent is made on `the cats` , then it is understood that John fed many animals and among all the cats were hungary. Otherwise, it is the case that John just fed the animals and those animals are actually cats. A context sensitive disambiguation has to be performed to select between the first interpretation,

which means `cats` are a subset of `animals`), and the second interpretation where the `animal` set is equal to the `cats`. Numerous discussions and examples are found in [Piwek, 1997], and it is to note that referent identification has to be done for correct prosody generation.

5 Conclusion

Linguistic analysis issues for text-to-speech synthesis are attempted to be reviewed from a natural language processing perspective within this study. At the end, a very general road map of language engineering towards building a TTS system may be summarized with the following stages:

1. Requested by the language's orthography or the application area of interest, a preprocessing including tokenization or vocalization is performed.
2. The non-standard words in the input text are resolved and converted into sequences of ordinary words.
3. After the syntactic words remain through by the help of normalization process, morphological analysis is deployed on those words for POS tagging (and maybe a deeper syntactic tagging). In this stage, the stressed syllables of every word are also determined.
4. Named entity recognitions and word sense disambiguations are made wherever appropriate.
5. Syntactic and/or semantic parses of the input are done by which phonological phrases and intonational boundaries are detected. Along with

information structure and the knowledge of stressed syllables coming from morphological analysis, the prosody of the utterance is generated.

References

- [Allen *et al.*, 1987] Jonathan Allen, M. Sharon Hunnicutt, and Dennis Clatt. *From Text to Speech: The MITalk System*. Cambridge University Press, Cambridge, 1987.
- [Atterer and Klein, 2002] Michaela Atterer and Ewan Klein. Integrating linguistic and performance-based constraints for assigning phrase breaks. In *Proceedings of COLING-2002*, Taipei, Taiwan, August 2002.
- [Bachenko and Fitzpatrick, 1990] J. Bachenko and E. Fitzpatrick. A computational grammar of discourse-neutral prosodic phrasing in English. *Computational Linguistics*, 16(3):155–170, September 1990.
- [Beesley, 1998] Kenneth. R. Beesley. Consonant spreading in Arabic stems. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the 36th annual meeting of ACL and 17th International Conference on Computational Linguistics*, pages 117–123, San Francisco, California, 1998. Morgan Kaufmann.
- [Beutnagel *et al.*, 1999] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. Syrdal. The AT&T Next-Gen TTS system. Joint Meeting of ASA, EAA and DAGA, Berlin, Germany, available at <http://citeseer.nj.nec.com/beutnagel199nextgen.html>, March 1999.

- [Bikel *et al.*, 1997] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: A high performance learning name-finder. In *Proceedings of Fifth Conference on Applied Natural Language Processing*, pages 194–201, Washington, D.C., 31 March–3 April 1997.
- [Black and Lenzo, 2003] Alan W. Black and Kevin A. Lenzo. *Building Synthetic Voices*. Language Technologies Institute, Carnegie Mellon University, <http://festvox.org/bsv/>, 2003.
- [Black and Taylor, 1994a] Alan W. Black and Paul Taylor. Assigning intonation elements and prosodic phrasing for English speech synthesis from high level linguistic input. In *Proceedings of ICSLP'94*, volume 3, Yokohama, Japan, 1994.
- [Black and Taylor, 1994b] Alan W. Black and Paul A. Taylor. CHATR: A generic speech synthesis system. In *Proceedings of the International Conference on Computational Linguistics COLING-94*, Kyoto, Japan, 1994.
- [Choueka and Neeman, 1995] Y. Choueka and Y. Neeman. Nakdan-text, an in-context text-vocalizer for modern Hebrew. BISFAI-95, 1995.
- [Chung *et al.*, 2003] Grace Chung, Stephanie Seneff, and Chao Wang. Automatic acquisition of names using speak and spell model in spoken dialogue systems. In *Proceedings of HLT-NAACL'2003*, pages 32–39, 2003.
- [Church, 1985] Kenneth Church. Stres assignment in letter to sound rules for speech synthesis. In *Proceedings of ACL'85*, pages 246–253, 1985.

- [Church, 1986] Kenneth Church. Morphological decomposition and stress assignment for speech synthesis. In *Proceedings of ACL '86*, pages 156–164, 1986.
- [Colins and Singer, 1999] Michael Colins and Yoram Singer. Unsupervised methods for named entity classification. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing*, pages 100–110, 1999.
- [Collins, 2002] Michael Collins. Ranking algorithms for named-entity extraction: Boosting and voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496, Philadelphia, July 2002.
- [Cucerzan and Yarowsky, 1999] Silvio Cucerzan and David Yarowsky. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing*, pages 90–99, 1999.
- [Daelemans and van den Bosch, 1997] Walter M.P. Daelemans and Antal P.J. van den Bosch. Language-independent data-oriented grapheme-to-phoneme conversion. In Jan P.H. van Santen, Joseph P. Olive, Richard Sproat, and Julia Hirschberg, editors, *Progress in Speech Synthesis*, chapter 2, pages 77–89. Springer-Verlag, 1997.
- [de Mareuil and Floricic, 2001] Philippe Boula de Mareuil and Franck Floricic. On the pronunciation of acronyms in French and in Italian. In *Proceedings of Eurospeech 2001*, Aalborg, Denmark, September 3-7 2001.

- [Edgington *et al.*, 1996] M. Edgington, A. Lowry, P. Jackson, A. P. Breen, and S. Minnis. Overview of current text-to-speech techniques: Part I - Text and linguistic analysis. *BT Technical Journal*, 14:68–83, 1996.
- [Ferri and nad Donatella Sanzone, 1997] Giuliano Ferri and Piero Pierucci nad Donatella Sanzone. A complete linguistic analysis for an Italian Text-to-Speech system. In Jan P.H. van Santen, Joseph P. Olive, Richard Sproat, and Julia Hirschberg, editors, *Progress in Speech Synthesis*, chapter 2, pages 123–137. Springer-Verlag, 1997.
- [Fitt, 2001] Susan Fitt. Morphological approaches for an English pronunciation lexicon. In *Proceedings of Eurospeech 2001*, Aalborg, Denmark, September 3-7 2001.
- [Gal, 2002] Ya'akov Gal. An HMM approach to vowel restoration in Arabic and Hebrew. In *Proceedings of ACL'02 Workshop on Computational Approaches to Semitic Languages*, University of Pennsylvania, Philadelphia, July 2002.
- [Gale *et al.*, 1992] William A. Gale, Kenneth W. Church, and David Yarowsky. One sense per discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, pages 233–237, 1992.
- [Gou, 1997] Jin Gou. Critical tokenization and its properties. *Computational Linguistics*, 23(4):569–596, 1997.
- [Guo, 1998] Jin Guo. One tokenization per source. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the 36th annual meeting of ACL*

and 17th International Conference on Computational Linguistics, pages 457–463, San Francisco, California, 1998. Morgan Kaufmann.

[Hearst, 1991] Marti Hearst. Noun homograph disambiguation using local context in large text corpora. In *Proceedings of the 7th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, Oxford, October 1991.

[Hirschberg, 1993] Julia Hirschberg. Pitch accent in context: Predicting intonational prominence from text. *Artificial Intelligence*, 63(1–2):305–340, 1993.

[Hiyakumoto *et al.*, 1997] Laurie Hiyakumoto, Scott Prevost, and Justine Cassell. Semantic and discourse information for text-to-speech intonation. In *Proceedings of ACL'97, Concept to speech generation systems workshop*, pages 47–56, Madrid, Spain, July 1997.

[Horne and Filipson, 1997] Merle A. Horne and Marcus K. D. Filipson. Computational extraction of lexico-grammatical information for generation of Swedish intonation. In Jan P.H. van Santen, Joseph P. Olive, Richard Sproat, and Julia Hirschberg, editors, *Progress in Speech Synthesis*, chapter 6, pages 443–457. Springer-Verlag, 1997.

[Jannedy and Möbius, 1997] Stefanie Jannedy and Bernd Möbius. Name pronunciation in German text-to-speech synthesis. In *Proceedings of Fifth Conference on Applied Natural Language Processing*, pages 49–56, Washington, D.C., 31 March–3 April 1997.

- [Jilka and Syrdal, 2002] Matthias Jilka and Ann K. Syrdal. The AT&T German text-to-speech system: Realistic linguistic description. In *Proceedings of ICSLP'2002*, September 2002.
- [Kamir *et al.*, 2002] Dror Kamir, Naama Soreq, and Yoni Neeman. A comprehensive NLP system for modern Arabic and modern Hebrew. In *Proceedings of ACL'02 Workshop on Computational Approaches to Semitic Languages*, July 2002.
- [Kontorovich and Lee, 2001] Leonid Kontorovich and Daniel D. Lee. Learning semitic vocalizations with hidden markov models. available at www.cs.huji.ac.il/~oslkonto/hmmvocs.ps, 2001.
- [Külepci and Özkan, 2001] M. Oğuzhan Külepci and Mehmed Özkan. Turkish word segmentation by using morphological analyzer. In *Proceedings of Eurospeech 2001*, Aalborg, Denmark, September 3-7 2001.
- [Lieberman and Church, 1992] Mark Y. Liberman and Kenneth W. Church. Text analysis and word pronunciation in text-to-speech synthesis. In Sadaoki Furui and M. Mohan Sondhi, editors, *Advances in Speech Signal Processing*, pages 791–831. Dekker, 1992.
- [Lin and Hung, 2002] Yi-Chung Lin and Peng-Hsiang Hung. Probabilistic named entity verification. In *Proceedings of COLING-2002*, Taipei, Taiwan, August 2002.
- [Lindström *et al.*, 1996] Anders Lindström, Ivan Bretan, and Mats Ljungqvist. Prosody generation in text-to-speech conversion using depen-

- dency graphs. In *Proceedings of ICSLP'96*, volume 3, pages 1341–1345, Philadelphia, PA, USA, October 3–6 1996.
- [Nootboom, 1997] Sieb G. Nootboom. Text and prosody. In Jan P.H. van Santen, Joseph P. Olive, Richard Sproat, and Julia Hirschberg, editors, *Progress in Speech Synthesis*, chapter 6, pages 431–434. Springer-Verlag, 1997.
- [Oflazer and Inkelas, 2003] Kemal Oflazer and Sharon Inkelas. A finite state pronunciation lexicon for Turkish. In *Proceedings of EACL workshop on finite state methods in NLP*, Budapest, Hungary, April 13-14 2003.
- [Olinsky and Black, 2000] Craig Olinsky and Alan W. Black. Non-standard word and homograph resolution for Asian language text analysis. In *Proceedings of ICSLP'2000*, Beijing, China, 2000.
- [Pfister and Romsdorfer, 2003] B. Pfister and H. Romsdorfer. Mixed-lingual text analysis for Polyglot TTS synthesis. In *Proceedings of the Eurospeech 2003*, 2003.
- [Pfister, 1995] Beat Pfister. The SVOX text-to-speech system. Computer Engineering and Networks Laboratory, Speech Processing Group, Swiss Federal Institute of Technology Zurich, available at www.tik.ee.ethz.ch/~spr/publications/pfister:95d.ps, September 1995.
- [Piwek, 1997] Paul Piwek. Accent interpretation, anaphora resolution and implicature derivation. In P. Dekker, M. Stokhof, and Y. Venema, editors, *Proceedings of 11th Amsterdam Colloquium*, pages 55–60, University of Amsterdam, ILLC/Department of Philosophy, 1997.

- [Prevost and Steedman, 1994] Scott Prevost and Mark Steedman. Specifying intonation from context for speech synthesis. *Speech Communication*, 15(1-2):139–153, 1994.
- [Prevost, 1996] Scott Prevost. An information structural approach to spoken language generation. In *Proceedings of ACL'96*, pages 294–301, University of California, Santa Cruz, USA, June 23–28 1996.
- [Ravin and Wacholder, 1996] Yael Ravin and Nina Wacholder. IBM research report : Extracting names from natural-language text. Technical Report RC 20338, IBM T.J. Watson Research Center, 1996.
- [Shalanova and Tucker, 2003] Ksenia Shalanova and Roger Tucker. South Asian languages in multilingual TTS-related database. Technical Report HPL-2003-9, HP Mobile and Media Systems Laboratory, Bristol, January 2003.
- [Shih and Sproat, 1996] Chilin Shih and Richard Sproat. Issues in text-to-speech conversion for Mandarin. *Computational Linguistics and Chinese Language Processing*, 1(1):37–86, August 1996.
- [Sonntag and Portele, 1997] Gerit P. Sonntag and Thomas Portele. Looking for the presence of linguistic concepts in the prosody of spoken utterances. In *Proceedings of ACL'97, Concept to speech generation systems workshop*, pages 57–63, Madrid, Spain, July 1997.
- [Sproat and Emerson, 2003] Richard Sproat and Thomas Emerson. The first international Chinese word segmentation bakeoff. In *Proceedings of the*

Second SIGHAN Workshop on Chinese Language Processing, pages 133–143, July 2003.

[Sproat *et al.*, 1996] Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22(3):377–404, 1996.

[Sproat *et al.*, 2001] Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. Normalization of non-standard words. *Computer Speech and Language*, 15(3):287–333, July 2001.

[Sproat, 1997] Richard Sproat. Multilingual text analysis for text-to-speech synthesis. *Natural Language Engineering*, 2(4):369–380, 1997.

[Taghva and Gilbert, 1999] Kazem Taghva and Jeff Gilbert. Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*, 1:191–198, 1999.

[Taylor *et al.*, 1998] Paul Taylor, Alan W. Black, and Richard Caley. The architecture of the Festival speech synthesis system. In *Proceedings of Third ESCA/COCOSDA International Workshop on Speech Synthesis*, Australia, November 1998.

[Tesprasit *et al.*, 2003] Virongrong Tesprasit, Paisarn Charoenpornasawat, and Virach Sornlertlamvanich. A context sensitive homograph disambiguation in Thai text-to-speech system. In *Proceedings of HLT-NAACL'2003*, pages 103–105, Canada, May 2003.

- [Wacholder *et al.*, 1997] Nina Wacholder, Yael Ravin, and Misook Choi. Disambiguation of proper names in text. In *Proceedings of Fifth Conference on Applied Natural Language Processing*, Washington, D.C., 31 March–3 April 1997.
- [Wang and Hirschberg, 1991] Michelle Q. Wang and Julia Hirschberg. Predicting intonational phrasing from text. In *Proceedings of ACL'91*, pages 285–292, University of California, Berkeley, California, June 17–21 1991.
- [Webster and Kit, 1992] Jonathan J. Webster and Chunyu Kit. Tokenization as the initial phase in NLP. In *Proceedings of COLING-92*, volume 4, pages 1106–1110, 1992.
- [Yarowsky, 1993] David Yarowsky. One sense per collocation. In *Proceedings of the ARPA Human Language Technology Workshop*, 1993.
- [Yarowsky, 1995] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.
- [Yarowsky, 1997] David Yarowsky. Homograph disambiguation in text-to-speech synthesis. In Jan P.H. van Santen, Joseph P. Olive, Richard Sproat, and Julia Hirschberg, editors, *Progress in Speech Synthesis*, chapter 2, pages 157–172. Springer-Verlag, 1997.
- [Ye *et al.*, 2002] Shiren Ye, Tat-Seng Chua, and Liu Jimin. An agent-based approach to Chinese named entity recognition. In *Proceedings of COLING-2002*, Taipei, Taiwan, August 2002.

[Zhou and Su, 2002] GuoDong Zhou and Jian Su. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, Philadelphia, July 2002.