MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

A Comparison of Anapron with Seven Other Name-Pronunciation Systems

Andrew R. Golding, Paul S. Rosenbloom

TR93-05a December 1993

Abstract

This paper presents an experiment comparing a new name-pronunciation system, Anapron, with seven existing systems: three state-of-the-art commercial systems (from Bellcore, Bell Labs, and DEC), two variants of a machine-learning system (NETtalk), and two humans. Anapron works by combining rule-based and case-based reasoning. It is based on the idea that it is much easier to improve a rule-based system by adding case-based reasoning to it than by tuning the rules to deal with every exception. In the experiment described here, Anapron used a set of rules adapted from MITalk and elementary foreign-language textbooks, and a case library of 5000 names. With these components – which required relatively little knowledge engineering – Anapron was found to perform almost at the level of the commercial systems, and significantly better than the two versions of NETtalk.

Journal of the American Voice Input/Output Society, 14 (August, 1993), 1-21

1. First printing, TR93-05, May 1993
2. Revised, TR93-05a, July 1993

A comparison of Anapron with seven other name-pronunciation systems[1]

Andrew R. Golding
Mitsubishi Electric Research Labs
201 Broadway, 8th Floor
Cambridge, MA 02139
golding@merl.com

Paul S. Rosenbloom
Information Sciences Institute and
Computer Science Department
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
rosenbloom@isi.edu

Abstract

This paper presents an experiment comparing a new name-pronunciation system, Anapron, with seven existing systems: three state-of-the-art commercial systems (from Bellcore, Bell Labs, and DEC), two variants of a machine-learning system (NETtalk), and two humans. Anapron works by combining rule-based and case-based reasoning. It is based on the idea that it is much easier to improve a rule-based system by adding case-based reasoning to it than by tuning the rules to deal with every exception. In the experiment described here, Anapron used a set of rules adapted from MITalk and elementary foreign-language textbooks, and a case library of 5000 names. With these components — which required relatively little knowledge engineering — Anapron was found to perform almost at the level of the commercial systems, and significantly better than the two versions of NETtalk.

[1]This research was sponsored by NASA under cooperative agreement number NCC 2–538, and by a Bell Laboratories PhD fellowship to the first author. Computer facilities were partially provided by NIH grant LM05208. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NASA, the US Government, Bell Laboratories, or the National Institute of Health.

We gratefully acknowledge the assistance on this experiment of the following people: Mark Liberman, for making the test set of names available; Cecil Coker at Bell Labs, Murray Spiegel at Bellcore, and Tony Vitale at DEC, for supplying data from their systems, and for helpful discussions on the design and analysis of the experiment; Tom Dietterich, for providing the non-copyrighted portion of NETtalk; John Laird, for providing a fast machine for training NETtalk; Connie Burton, for providing access to DECtalk; and David Shapiro, for his excellent guidance on the statistical analysis. We are indebted to Murray Spiegel for arranging for us to borrow a pronouncing dictionary of names from Bellcore. We would also like to thank Bill Freeman and the AVIOS reviewers for comments on this paper.

1 Introduction

This paper presents an experiment comparing a new name-pronunciation system, Anapron, with seven existing systems: three state-of-the-art commercial systems (from Bellcore, Bell Labs, and DEC), two variants of a machine-learning system (NETtalk), and two humans. Anapron is based on a general method for improving rule-based systems through case-based reasoning [Golding and Rosenbloom, 1991]. It applies its rules to generate a first approximation to the pronunciation of a name, and it draws analogies from names in its case library to cover exceptions to the rules. This provides a way of enhancing an imperfect rule set with relatively little effort: obtaining cases — in the form of a pronouncing dictionary of names — is often much easier than the alternative of fine-tuning the rules to anticipate every contingency. For the implementation discussed here, Anapron used a set of rules adapted from MITalk [Hunnicutt, 1976] and elementary grammar texts for French, German, Italian, and Spanish, and it used a case library of 5000 names. With these components — which required relatively little knowledge engineering — Anapron was found to perform almost at the level of the commercial systems in the experiment.

The experiment involved running Anapron and each of the seven other systems on the same 400-name test set. The resulting pronunciations were piped through a DECtalk™[2] speech synthesizer, in random order. A panel of 14 test subjects judged the acceptability of the pronunciations. Two caveats about the results: first, the scores for the various systems represent text-to-phonetics performance only, not full text-to-speech performance. We essentially factored out the phonetics-to-speech component of each system by using DECtalk, since our goal was to compare systems to Anapron, and Anapron has no phonetics-to-speech component. Second, the way we factored out phonetics-to-speech did not preserve the relative strengths of all systems; in particular, it favored the commercial system from DEC, which was designed to have its pronunciations fed through DECtalk, relative to the systems from Bellcore and Bell Labs. For purposes of evaluating Anapron, this is tolerable, since we are more interested in getting an idea of how Anapron compares to other systems than in getting exact performance figures. But it is important to note that therefore this experiment does *not* support comparisons between one commercial system and another.

The next section gives an overview of the systems involved in the experiment. Sections 3 and 4 present the experimental design and analysis. Section 5 is a conclusion. The pronunciation notation used throughout the paper, from DECtalk, is defined in the appendix.

[2] DECtalk is a trademark of Digital Equipment Corporation.

2 System Overview

Each of the systems in the experiment is described briefly below, followed by a discussion of how Anapron relates to the other systems.

2.1 The Systems

Anapron[3]: Divides pronunciation into five main subtasks: language identification, morphological decomposition, transcription (mapping letters to phonemes), syllabification, and stress assignment. Transcription and stress assignment are each done by a combination of rule-based and case-based reasoning, as follows: the system starts by applying its rules. After each rule application, it uses the rule just applied to index into its dictionary and retrieve names that illustrate exceptions to that rule. If the system finds a compelling analogy between the name it is pronouncing and one of these exceptions, then it modifies its answer to follow the exception rather than the rule. The system decides whether an analogy is compelling based on two factors: the degree of similarity between the two names, as determined by a similarity metric; and the results of an *empirical verification*, which tests out the generalization behind the analogy on other names in the dictionary.

As an example from transcription, suppose the system is pronouncing the name Donahower. One rule that fires for this name says to pronounce the OW as `ow`[4] (as in b<u>oa</u>t). Associated with this rule is a list of dictionary names that illustrate exceptions to it. One such exception is Bower — the rule predicts `ow`, but the dictionary pronunciation gives `aw` (as in b<u>ou</u>t). The system tries drawing an analogy from Bower to Donahower, to see if Donahower is a similar exception. This entails applying a similarity metric to the two names. The metric compares the two names around the OW, matching letters that are identical or in the same abstract class (e.g., "orthographic vowel"). It finds a shared right-hand context of ER# (where # marks a word boundary) and no shared left-hand context. It assigns a degree of similarity commensurate with this amount of shared context. The generalization behind this analogy is that "OW is pronounced `aw` when followed by ER#". The system tests this generalization on other names in its dictionary, and finds that it is correct for all applicable names: Bower, Brower, Flower, Hightower, Hower, and Power. Based on this empirical evidence, together with the score from the similarity metric, the system ends up accepting this particular analogy. Thus it pronounces the OW in Donahower as `aw` by analogy with Bower.

Anapron's rule set includes 619 transcription rules and 29 stress-assignment rules, drawn from MITalk [Hunnicutt, 1976] and introductory textbooks on French, German, Italian, and Spanish. The dictionary contains 5000 surnames, including the 2500 most frequent ones in the US, 1250 sampled randomly from ranks 2500 through 10,000, and 1250 from ranks 10,000 to 60,000. [Golding, 1991]

[3] Anapron stands for <u>Ana</u>logical <u>pron</u>unciation system.

[4] Pronunciations are given in DECtalk notation, which is defined in Appendix A.

The Orator®[5] System (Bellcore): First looks up the name in a small exception dictionary (about 2500 entries). If the name is not found, the system determines what language it is from. It then breaks the name into morphemes. Each morpheme is pronounced by dictionary lookup if possible, else by rules. The rules are sensitive to orthographic context, morpheme boundaries, and language. The rules were specially developed for names, with the philosophy of mimicking the anglicizations that are commonly heard in the US, rather than adhering strictly to the native pronunciations. A rule compiler converts the rules into a finite-state machine for run-time efficiency. [Spiegel and Macchi, 1990]

TTS (Bell Labs): Applies dictionary-based methods, the simplest of which is direct lookup. The lookup is done in a dictionary of the 50,000 most frequent surnames in the US. If direct lookup fails, the system tries progressively riskier methods to derive the name from dictionary entries. The methods include, among others: appending a stress-neutral ending to a dictionary name to get the target name (e.g., Abelson = Abel + son); finding a dictionary entry with a different suffix, and performing suffix exchange (e.g., Agnano = Agnelli − elli + ano); and drawing a rhyming analogy from a dictionary entry (e.g., Alifano from Califano). If all of the dictionary-based methods fail — which rarely happens — the name is passed to a rule-based system, Namsa. [Coker *et al.*, 1990]

DECvoice II (DEC): Uses an early version of DEC's name-pronunciation software; replaced by a later version in DECtalk PC. First performs dictionary lookup. For names not found, the system identifies the language of the name, and applies rules for that language. Language identification is done in two steps. First, a set of filter rules is applied. The filter rules determine what language the name is from, or at least what languages it is *not* from, based on characteristic letter patterns in the name. If multiple candidate languages remain, the system chooses among them via trigram analysis. [Vitale, 1991]

BP-legal[6] (NETtalk): A connectionist network that learns to read aloud by being trained on a dictionary. The network pronounces one letter at a time. Its output is a set of activation levels, which are mapped to the nearest bit string that represents a legal phoneme/stress pair. [Sejnowski and Rosenberg, 1987]

BP-block (NETtalk with block decoding): Like BP-legal, but with a "block decoding" postprocessor added: after the system has found the nearest legal phoneme/stress pair for each letter of a word, it looks in the word for letter sequences of length 1–5 that also appeared in the dictionary. For these sequences, it copies the dictionary pronunciation that is closest to the pronunciation it already had. [Dietterich *et al.*, 1990]

MJW (a human): A 27-year-old male Computer Science Phd student at Stanford University. He grew up near Austin, Texas, but has close to "newscaster" pronunciation, with just a slight Southern twang. He had some German in high school and some Spanish in elementary school.

[5] Orator is a registered trademark of Bellcore.

[6] BP stands for "backpropagation", the procedure used for training the network.

TJG (a human): A 32-year-old male Psychology postdoc at Stanford University. He grew up in New Jersey, and has a mild accent of that region. He studied French and some German and Hebrew, and has travelled abroad extensively.

2.2 Relation to Anapron

The method each system uses to incorporate rules and cases serves as a useful basis for relating Anapron to the other computer systems. Anapron starts with rule-based reasoning, and draws analogies from cases in its dictionary to cover rule exceptions. Two of the commercial systems — the Orator system and DECvoice — do rule-based reasoning plus dictionary lookup of names or name morphemes. The difference between these systems and Anapron is in how they use cases: these systems map a case to other occurrences of the same case, whereas Anapron maps a case more generally to any new name that is deemed compellingly similar. The cost of Anapron's increased generality, however, is that it needs a similarity metric to help it judge the similarity between cases.

The two NETtalk-based systems, BP-legal and BP-block, work purely from cases. The difference between these systems and Anapron is that these systems do not use rules at all. This puts these systems at something of a disadvantage when being compared to Anapron, in that Anapron works from a superset of their knowledge. However, this disadvantage is due to the systems' own limitation in accepting only one form of knowledge.

The remaining computer system, TTS, is the most similar to Anapron, in that it does both rule-based reasoning and a non-degenerate form of case-based reasoning. There are three principal differences between the two systems, however. First, TTS applies cases before rules, the opposite of Anapron. This reflects an underlying dichotomy of approaches: TTS is based on having a large dictionary that will allow it to look up or derive most names it will encounter. Anapron, on the other hand, is based on having a decent set of rules that will cover broad, regular aspects of pronunciation, leaving a relatively small set of idiosyncratic behaviors to be handled by analogy. The second difference between TTS and Anapron is that in TTS, the case-based and rule-based components are essentially independent; they are just called sequentially. In Anapron, they are more tightly coupled: the system retrieves cases specifically to contradict whichever rule was applied. The third difference between TTS and Anapron is in how each system does case-based reasoning. TTS runs through a fixed sequence of methods (suffix exhange, etc.) to derive whole names from large parts of other names, while Anapron transfers one letter cluster or aspect of stress at a time via a general analogical mechanism. The trade-off between the two approaches is basically one of generality versus efficiency: Anapron can find a wider class of analogies, but TTS can be optimized to find the types it knows about very quickly. Anapron requires correspondingly general knowledge (a similarity metric), compared to TTS's more specialized knowledge (a sequence of analogy types to try).

3 Design

Two main issues shaped the design of the experiment. The first was selecting a performance task. We chose the general-purpose task of simply reading a list of names, since the goal of the experiment was similarly general-purpose — to get an idea of how Anapron compares with other systems. For a more specific goal, such as evaluating systems for telephone-based reverse directory assistance, a correspondingly specific performance task is more appropriate (see Basson *et al.* [1991]).

The second main issue was choosing the form of output of the systems. We took the phonetic transcriptions produced by each system, and piped these through DECtalk. This essentially factored out the phonetics-to-speech component of each system, making the experiment a comparison of text-to-phonetics methods. This is suitable for comparison with Anapron, since Anapron just does text-to-phonetics. Piping the pronunciations through DECtalk had a side-benefit, in that it hid the identities of the systems from the judges. Judges might otherwise have developed a bias against a particular system, e.g., because it mispronounced a name they knew, or simply because it had a mechanical voice (as opposed to the human systems in our experiment). Using DECtalk also had a drawback, though, in that it did not represent the full text-to-speech capability of each system.

Given the preceding design decisions, we carried out the experiment in four steps: (1) compile a list of names to test the systems on; (2) run the names through each system; (3) make a cassette tape of the systems' pronunciations; and (4) have a group of test subjects judge the pronunciations on the tape. Each step is described below.

3.1 Test set

The test set was drawn from the Donnelley corpus, a database of over 1.5 million distinct surnames covering 72 million households in the US. Some names are more common than others — we will refer to the number of households that have a particular name as the *frequency* (of occurrence) of that name. To a rough approximation, the distribution of names in Donnelley follows Zipf's Law [Spiegel, 1985]. Zipf's Law states that if items are rank-ordered by frequency of occurrence, the frequencies are inversely proportional to the ranks. Thus the top-ranking names occur in huge numbers of households (e.g., Smith, ranked #1, occurs in over 670,000 households), but frequency drops off rapidly, ending with a long flat tail of names that occur in just one household (Chavriacouty and about 650,000 others).

It is difficult to construct a test set that is representative of all of Donnelley and still of tractable size. One strategy would be to select names at random according to the naturally-occurring distribution. But the quality of the resulting measurements would depend on the distribution — we would get poor representation of the rare names. Instead, we constructed a test set consisting of equal numbers of names from various points along the frequency

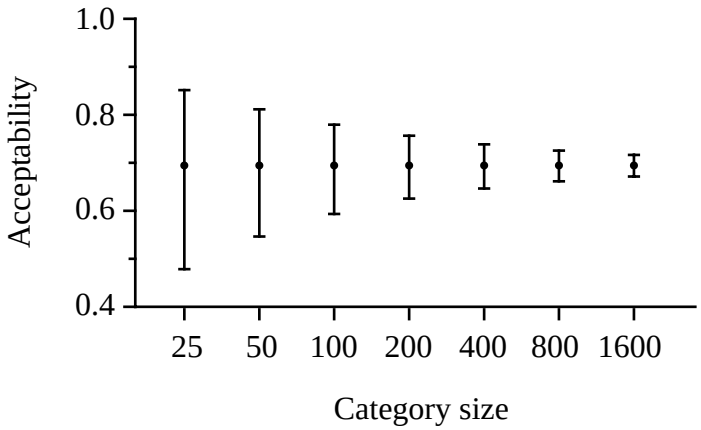

Figure 1: The 95% confidence interval for measuring an acceptability rating of 0.70, shown for various choices of category size. The category-size axis is scaled logarithmically.

spectrum. This allowed us to get reliable readings on how system performance varied as a function of name frequency. The question then arose of which frequencies to sample. Since our goal was to evaluate Anapron's performance, we picked points that were most informative for that purpose. This meant that at places in the frequency spectrum where Anapron's performance function changed quickly, we took more closely-spaced measurements; at places where the function was more constant, we sampled less often. It turns out that Anapron's performance function drops about linearly as frequency is decreased exponentially — this was determined in a pilot study. Thus we sampled frequencies that were distributed roughly exponentially: frequency 1 (ultralow), 32 (low), 256 (mid), and 2048 (high). This left out names above frequency 2048, but there are fewer than 4500 such names — thus they could all be covered with a moderate-sized dictionary; they are not the ones that Anapron is targetted for. As for choosing the names at each of the four frequencies, we chose the names for a frequency F randomly from the names of frequency F in Donnelley. If Donnelley had fewer names at frequency F than we wanted to sample, we chose names from the narrowest symmetric frequency band around F that would suffice.

Finally, there was the question of how many names to pick in each frequency category. We again based our answer on the pilot study of Anapron's performance. This showed that the acceptability rate was in the ballpark of 0.70. We then asked what size confidence interval in this measurement was satisfactory, and chose the size of the categories accordingly. Figure 1 shows the confidence intervals for several choices of category size, calculated using the standard error of a proportion [Fleiss, 1981, p.14]. It shows that it takes a fair number of names to get a reliable reading for a frequency category. We chose a category size of 100. This gives a somewhat broad confidence interval, but was the largest size that was considered practical. It resulted in a test set of 400 names, which took 1 1/2 hours to read (given that each name had up to 8 different pronunciations, one per system).

3.2 Data collection

Pronunciations for the names in the test set were gathered as follows: for the computer systems, the system was run on the test set, and its phonetic transcriptions were collected. For the humans, we asked them to read the list of names aloud, as if they were a teacher taking roll call, and we tape-recorded their pronunciations.

The NETtalk system also needed to be trained. We trained it on the same 5000-name dictionary used by Anapron. This involved first converting the dictionary into NETtalk notation, in which each spelling is aligned with its transcription and stress pattern. We configured NETtalk as Sejnowski and Rosenberg did: with 120 hidden units, learning rate 0.25, momentum coefficient 0.9, and random starting weights in the range $[-0.3, 0.3]$. Training proceeds in a series of epochs, where each epoch consists of running the full dictionary through the network, and adjusting the weights via a backpropagation procedure so as to reduce error. The code, written in C, for the backpropagation procedure was taken from *Explorations in PDP* [McClelland and Rumelhart, 1988]. Training is complete when the total error of the network drops to a specified target level. We set the target level to 2450, obtained by scaling Dietterich's value of 445 [Dietterich *et al.*, 1990] for the size of our dictionary — his dictionary had 5807 total letters, ours had 31975. For Dietterich, training took 30 epochs. For us, after 30 epochs the error was 7471, more than triple the target level. We continued training for a total of 150 epochs. It took about 2 1/3 hours of CPU time per epoch on an IBM RS/6000, for a total of over 2 CPU weeks. At this point we contented ourselves with the resulting error level of 4102 (still 67% larger than desired). How long would it have taken to reach the target level? The total sum-of-squares error, TSS, in the network decreases as a power law of the number of epochs, N. Using regression analysis, we obtained a close fit (R^2=99.1%) of a power law to the data for the 150 epochs:

$$\mathrm{TSS} = 27320\, N^{-0.385}$$

From this formula, we project that it would have taken 525 epochs (almost 2 CPU months) to reach the target error level.

3.3 The cassette tape

Once pronunciations were obtained for all systems, we made a cassette tape of DECtalk (version 2.0) reading the pronunciations aloud. For the computer systems, this required first converting the phonetic transcriptions from their original notation into DECtalk notation. For the humans, it required transcribing the pronunciations directly into DECtalk notation. At this point, we had 8 transcriptions — one per system — in DECtalk notation for each name in the test set. For each name, we deleted duplicate transcriptions, and permuted the rest randomly. We randomized the order of names as well. We then fed the transcriptions through DECtalk, to obtain a tape with between 1 and 8 pronunciations of each of 400 names. The tape was 1 1/2 hours long, with a total of 1650 pronunciations.

The main shortcoming of this procedure is that it involved running pronunciations through a different synthesizer than the one they were designed for. To adapt the pronunciations to the new synthesizer, we had to: (1) undo any optimizations for the original synthesizer — most systems have been tailored to produce whatever output sounds best on the synthesizer they are using; (2) switch to the new synthesizer's phonetic notation; and (3) reoptimize the pronunciations for the new synthesizer. This conversion affected different systems different amounts. In particular, it favored the commercial system from DEC, which was intended to be used with a DECtalk synthesizer, relative to the systems from Bellcore and Bell Labs. This is tolerable for purposes of getting an idea of how Anapron compares with the other systems. However, to reduce the impact of the degradation, we only counted each system's *egregious* errors when analyzing the results, where an egregious error is a pronunciation that, according to the judges, "no one would say" (see Section 4.1). This measure should be largely unaffected by the above conversion difficulties. Other factors — such as DECtalk's intelligibility for words spoken in isolation — affect all systems equally.

Following is a description of the non-trivial parts of the conversion into DECtalk notation. For the DEC system and the humans, only item (1) was applied.

(1) Normalize choice of phonemes

In some cases, DECtalk notation provides multiple ways of representing highly similar or identical sounds. For instance, "bore" can be transcribed as `b´aor`, `b´owr`, or `b´or`. In our test set, this can lead to names with several pronunciations that sound alike. To avoid torturing the judges with such redundant pronunciations, we did two things: (i) we expressed r-colored vowels using DECtalk's `ar`, `er`, `ir`, `or`, `ur`, and `rr`; and (ii) we collapsed both types of schwa, `ax` and `ix`, to `ax`.

(2) Incorporate stress level into choice of vowel phoneme

DECtalk has no stress marker for 0-stress; so instead we indicated 0-stress through the choice of vowel. Specifically, we used a schwa or syllabic consonant if and only if the syllable was 0-stress. To enforce this, we reduced all short vowels in 0-stress syllables to schwa. (We left long vowels as is, since we never regard them as 0-stress.) Conversely, we promoted all schwas in non-0-stress syllables to `ah`. Examples:

Sherrod	`shehraad` ⟶ `sh´ehraxd`	;	Reduce (short) `aa` to `ax`
Turney	`trrniy` ⟶ `t´rrniy`	;	Leave (long) `iy` as is
Chun	`chaxn` ⟶ `ch´ahn`	;	Promote `ax` to `ah`

(3) Delete 2-stresses

Once the preceding step was done, we could distinguish 0-stress from 2-stress syllables by the choice of vowel phoneme. This made 2-stresses largely redundant. In fact, most pronunciations sound better in DECtalk without the 2-stresses, in the opinion of the authors. Compounds (e.g., Newhouse) are occasionally an exception; but lacking a principled way of detecting such cases, we deleted all 2-stresses.

A few technical details about making the cassette tape: DECtalk was set to its default voice (Perfect Paul), pitch, and volume. For increased clarity, the speaking rate was reduced to 140 words/minute from the default of 180. Each name on the tape was read as follows: the number of the name (from 0 to 399), a pause (for the judges to consider how *they* would pronounce the given spelling), the different pronunciations of the name separated by pauses, and a final pause. The length of the pauses varied with the difficulty of the name, where difficulty was gauged by the number of different pronunciations the 8 systems generated. For instance, Chun was easy because all 8 systems agreed on its pronunciation. Loizakes was hard because all 8 disagreed. The exact ranges of pause lengths were 1/2 to 2 1/2 seconds for the post-number pause, 1 to 1 1/2 seconds for the inter-pronunciation pause, and 2 to 2 1/2 seconds for the final pause. These values were arrived at by informal trial and error. The whole tape had 37 minutes of speaking and 52 minutes of pauses.

3.4 Experimental procedure

The judges in the experiment were 13 Stanford undergraduates from the Psych 001 subject pool, plus one subject not from the pool. All subjects were required to be native speakers of American English, and not Linguistics majors. The intent was to get native speaker intuitions. The 13 pool subjects ranged in age from 17–21, and in geographical background from the West Coast of the US to the Northeast. There were 10 female and 3 male pool subjects. The non-pool subject was a 46-year-old male business analyst from the West Coast, with a JD degree. Roughly 2/3 of the subjects had training in at least one foreign language, French and Spanish being the most common. The group was intended to be a representative cross-section of the US population, but clearly the level of education was above normal.

The experiment was run in one group session on all subjects, except that the non-pool subject was run separately. The first author conducted the experiment. The S's were provided with a score sheet of the 400 names. For each name, the sheet gives the number of the name (from 0 to 399), the spelling, and one box for each of its pronunciations. The S's were asked to listen to the tape and score each pronunciation on a scale of 1 to 3:

1 = Clearly acceptable; e.g., as they would say it
2 = Somewhere in between; they could imagine that someone might say it that way
3 = Clearly bad; no one would say it that way.

The rating scheme is taken from Coker *et al.* [1990]. If a pronunciation goes by too quickly for them to score it, the S's were instructed to leave a blank and raise their hand. Time permitting, the experimenter would then rewind the tape and give them another chance. In practice, this happened two or three times; in the end, no subject left any blanks. To help them get the hang of the procedure, the experimenter started with a practice tape of 4 names. During the playing of the actual tape, the experimenter paused the tape briefly every page of the score sheet (once every 40 names) and took a short break every 100 names. Some subject fatigue was apparent by the end of the session.

4 Analysis

The sections below analyze the experimental data in various ways, starting with a summary of each system's performance on the test set, and a brief look at the actual pronunciations generated; continuing with an analysis of the significant differences between Anapron and the other systems, and how this relates to system performance in practice; and finally investigating the reliability of our test subjects' judgements.

4.1 Performance on the test set

As mentioned in Section 3.3, the methodology of this experiment precludes making fine distinctions in pronunciation quality among systems; instead, we focus on the gross differences. We do this by lumping scores of 1 and 2 together into *acceptable* scores, and counting a score of 3 as *unacceptable*. Tables 1 and 2 summarize the results. Each table gives the percentage of acceptable scores, out of a total of 5600, awarded to each system (5600 = 14 judges times 400 pronunciations). The scores are broken down by name frequency in Table 1, and by judge in Table 2. The tables include an imaginary ninth system, labelled Ubound. This system generates for each name the pronunciation that received the greatest number of acceptable votes from the judges. It measures the degree to which all judges can be pleased simultaneously, using just the pronunciations available from the eight systems tested. This represents an upper bound on the scores achievable in this experiment. Two points about the interpretation of the scores in these tables: first, as pointed out in Section 1, although they are valid for comparing Anapron with the other systems, they do not represent full, unbiased text-to-speech performance, and thus should not be used for comparing one commercial system with another. Second, the "overall" scores in these tables rate the systems on our 400-name test set, not on the full distribution of names in Donnelley. The extrapolation to all of Donnelley will be discussed in Section 4.4.

Table 1 shows that for all systems, performance degrades as the names get rarer. Also, although the eight systems seem to hit a performance asymptote at 93%, it is likely that humans with more exposure to names than MJW and TJG (e.g., telephone operators) would score higher; the Ubound system demonstrates that it is possible to score at least 97%. This suggests that there is room for improvement in all systems.

Table 2 shows that there is considerable variability among judges. For instance, the scores for BP-legal range from 51% to 78%. This variability was confirmed by calculating interrater agreement, kappa [Fleiss, 1981, p.218]. Kappa was found to be 0.357, indicating poor agreement beyond chance among judges on their scores. On the other hand, the judges agreed closely in their *rankings* of the eight systems. For example, despite the discrepancies in the scores assigned by the judges to BP-legal, they all ranked it in last place. We measured the agreement among judges on rankings by calculating Kendall's coefficient of concordance, W [Kendall and Gibbons, 1990]. We found W=0.934, indicating almost perfect agreement.

System	Name frequency				Overall
	H	M	L	UL	
Ubound	98	98	98	96	97
TJG	97	93	93	88	93
MJW	98	94	94	86	93
Orator	97	95	93	90	93
DECvoice	96	90	87	86	90
TTS	96	94	89	78	89
Anapron	91	88	85	80	86
BP-block	84	83	77	69	78
BP-legal	78	72	66	52	67

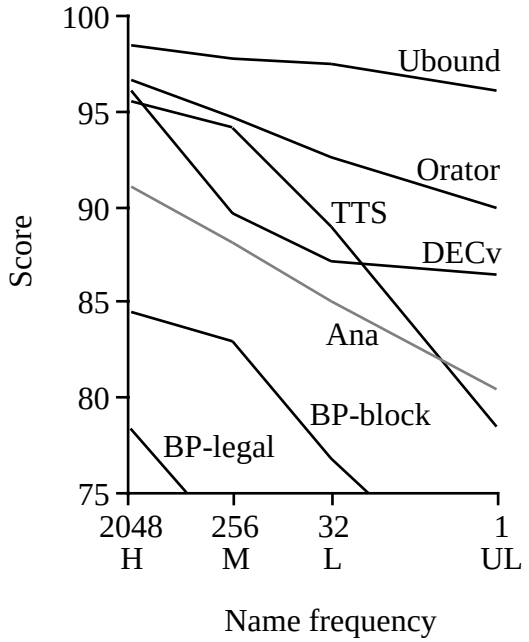

Table 1: Percentage of acceptable scores for each system, broken down by name frequency. The data are shown as a table and as a graph. For readability, the humans are omitted from the graph. The curves for the BP systems are truncated when they fall below a score of 75. The frequency axis is scaled logarithmically. As mentioned in the text (Section 4.1), these scores are for comparison with Anapron only; they are not valid for comparing one commercial system with another.

System	Judge													
	J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12	J13
Ubound	96	97	97	98	96	96	100	93	99	99	100	100	96	98
TJG	93	92	90	97	89	92	98	79	98	95	97	95	91	95
MJW	92	92	92	97	88	91	98	83	97	96	97	95	90	92
Orator	92	92	91	98	90	91	98	82	97	97	98	98	91	94
DECvoice	85	88	87	96	87	84	96	78	95	96	95	93	86	91
TTS	90	87	86	95	83	85	95	79	96	94	93	91	86	90
Anapron	83	85	81	94	80	81	94	74	92	93	90	91	80	87
BP-block	80	70	73	92	68	71	88	62	88	88	85	78	71	82
BP-legal	66	59	62	80	60	58	78	51	74	77	76	66	59	69

Table 2: Percentages of acceptable scores for each system, broken down by judge.

This value was highly significant (χ^2=91.5, df=7, p<0.001). Given this consensus on *relative* strengths, we are on firm ground when drawing conclusions about Anapron's performance relative to the other systems. We should not, however, put too much stock in the *absolute* acceptability scores, as these vary depending on whom you ask.

4.2 The pronunciations behind the numbers

To get an idea of where the acceptability scores come from, we now take a brief look at the actual pronunciations generated by the systems. We do this for the best and worst systems. We start with the best ones, namely the humans. It is perhaps surprising that each human scored only 93%. In other words, they were judged by their peers to generate pronunciations "that no one would say" for 7% of the names in the test set. Table 3 lists the lowest-scoring pronunciations for each human. All pronunciations receiving acceptability scores of under 70% (i.e., fewer than 10/14 judges said they were acceptable) are included. In a few cases, it seems unclear why the judges were so harsh. For example, on Witucki, only 9 out of 14 judges accepted MJW's pronunciation, `wiht´ahkiy`, or TJG's pronunciation, `waxt´uwkiy`, both of which seem plausible to the authors. In fact, *no* system got more than 10/14 for Witucki. Names like this are the reason that even the Ubound system did not score 100%. The fault could have been in DECtalk's synthesis of the pronunciations, but again it seemed unobjectionable to the authors. On the whole, though, the judges seemed to be justified in their scoring. For instance, although they gave low marks to TJG for his pronunciations of Lieszewicz and Sweitlowicz, TJG admitted during debriefing that he did not remember how to pronounce the -wicz ending. Thus it would appear that the humans scored only 93% simply because they are fallible name pronouncers. This is especially true for rare names; in Table 3, 14 out of 19 of MJW's names are ultralow, as are 11 out of 19 of TJG's names.

At the opposite end of the performance spectrum from the humans are the two incarnations of NETtalk, BP-legal and BP-block. These systems suffer from two special pronunciation problems. The first one, which afflicts BP-legal only, is a predilection to insert the sound `waa` into its pronunciations. This happens because the compound phoneme `waa` (as in bourgeois), acts as a sort of default phoneme in BP-legal — BP-legal generates it whenever it encounters an unusual pattern of input letters for which it has not learned any strong response. This results in such quaint pronunciations as `sw´aytlwaawihwaa` for Sweitlowicz. The second problem, which affects both BP-legal and BP-block, is a tendency to assign illegal stress patterns — patterns containing zero or multiple 1-stresses. This happens because NETtalk assigns stress to a vowel based on the vowel's local environment (a 7-letter window), making it hard to enforce the global constraint of a unique 1-stress. The incidence of these two errors is summarized in Table 4. The rows of the table give: (i) the overall acceptablity score for the system (for comparison), (ii) the percentage of names in the test set affected by each error, (iii) the acceptability score for the system for just the names that were affected by the error, and (iv) a hypothetical overall acceptability score for the system, had all names with the error been rated as 0% acceptable. The table shows that both errors are widespread:

MJW			TJG		
Name	Pronunciation	%	Name	Pronunciation	%
Van Scyoc	vaensk´ahch	14	Lieszewicz	l´ayzaxwihsk	21
Osorioleal	axs`oriyowliy´aal	21	Osorioleal	aas´oriyowliy`ael	29
Crawemeyer	kr´eywaxmayrr	36	DelPrete	dehlpr´eytey	36
Cincilus	chihnch´ihlaxs	43	Mielcarke	miylk´arkey	43
Hamidana	hxaem´ihdaxnax	43	Pantinople	paentaxn´owpliy	43
Cherundolo	ch´ehraxndaalow	43	Le Comte	laxk´ownt	50
Harroun	hxaer´awn	50	Crawemeyer	kr´aowaxmayrr	50
Mijotovich	miyjhaxt´owvihch	50	Van Scyoc	vaensk´ayaak	57
Taglialavoro	taegl`iyaxlaxv´orow	50	Zackery	z´aek`rriy	57
Mazzacapa	maxz´aekaxpax	50	Gadatia	gaxd´aedxiyax	57
Le Comte	laxk´ahmt	57	Sochin	s´owchaxn	64
Agorillo	axjhrr´ihlow	57	Sweitlowicz	sw´aytlaxvihsk	64
Karadjich	kaxr´ihjhihch	57	Ripani	raxp´aeniy	64
Gilanfar	jhaxl´aanfar	57	Lantinov	l´aentaxnaxf	64
Munyasya	myuniy´aasiyax	64	Witucki	waxt´uwkiy	64
Barngover	b´arnxowvrr	64	Wiegand	w´aygaxnd	64
Chikwana	chaykw´aanax	64	Maquedang	m´aakdaanx	64
Witucki	wiht´ahkiy	64	Calefate	kaelaxf´aatey	64
Bourimavong	b´orihmaxv`aanx	64	Bazane	baxz´aaney	64

Table 3: Lowest-scoring pronunciations generated by each of the humans, MJW and TJG. Each entry gives the spelling of a name, the human's pronunciation, and the percentage of acceptable scores that the pronunciation received from the judges.

`waa` occurred in about 1 of 6 names, while illegal 1-stresses occurred in about 1 of 3. The `waa` error clearly degraded BP-legal's overall score. Performance on the `waa` names was a paltry 14%, as compared with 67% for BP-legal overall. However, things could have been a little worse — if performance on the `waa` names had been 0%, then BP-legal would have scored 65% overall. For the stress error, NETtalk's performance (for both BP-legal and BP-block) was only about 15% lower on names with the error than overall. Thus NETtalk may actually have gotten off easy. If the judges had been uncompromising about illegal stress patterns, BP-legal would have scored 52% overall, and BP-block would have scored 58%.

4.3 Significant differences between systems

The main question addressed in this experiment is how Anapron performs relative to the other systems. This was tested statistically in two phases. First, we ran an ANOVA to see if there was a significant difference in scores between *any* systems. The result was positive.

	waa-insertion	Illegal 1-stress	
	BP-legal	BP-legal	BP-block
Overall score	67	67	78
Percent of names affected	16	30	31
Score on affected names	14	50	65
Hypothetical overall score	65	52	58

Table 4: Incidence of the waa-insertion and illegal 1-stress errors for BP-legal and BP-block.

We then ran a planned comparison to localize the significant differences — in particular, to see if there was a significant difference between Anapron and any of the other seven systems.

The data for the ANOVA consist of an $8 \times 14 \times 4$ array of cells. The three dimensions are system, judge, and name frequency. Each cell contains 100 observations, for the scores awarded to a particular system by a particular judge for the 100 names in a particular frequency category. Each observation is an integer from 1 to 3. As is, these categorical observations do not satisfy the ANOVA assumptions, because they do not follow a normal distribution. We therefore normalized each cell in the 3D array by first collapsing its 100 observations into a single value, the proportion of *acceptable* scores; and then applying the double-arcsine transformation to assure equal variances across cells [Freeman and Tukey, 1950, p.607]. We then performed a 3-way, fixed-effects ANOVA. The ANOVA says whether differences between systems, between judges, or between name frequencies account for a significant portion of the variance between cell values. It also says whether interactions among these three dimensions are significant.

The results were that there were in fact significant differences between systems ($F=649$, $p<0.001$), between judges ($F=183$, $p<0.001$), and between frequency categories ($F=575$, $p<0.001$). All pairwise interactions were significant at the 0.001 level as well, although these effects were quite minor compared to the main effects. The interaction between system and name frequency indicates that some systems are affected more than others by changes in name frequency. This shows up in the graph of system score versus name frequency (Table 1), where the curve of the TTS system slopes steeply enough to cross the curves of DECvoice and Anapron.

Having established that significant differences exist, we then tested which particular pairs of systems were significantly different. We used the Bonferroni multiple comparison procedure [Miller, 1981, p.68]. We made seven comparisons, for Anapron versus each of the other systems. This design was preferred over the blanket approach of comparing all pairs of systems, because (1) the fewer comparisons we make, the more powerful each comparison can be, without increasing the risk of detecting spurious differences; and (2) given that our goal is to evaluate Anapron, comparisons that do not involve Anapron are superfluous. The results of the Bonferroni procedure appear in the last column of Table 5. The table shows

System	Name frequency				Overall
	H	M	L	UL	
TJG	+	+	+	+	+
MJW	+	+	+	+	+
Orator	+	+	+	+	+
DECvoice	+	+?	+?	+	+
TTS	+	+	+	–?	+
BP-block	–	–	–	–	–
BP-legal	–	–	–	–	–

Table 5: Differences in performance between other systems and Anapron, broken down by name frequency. A plus sign (+) means higher acceptability than Anapron, and a minus sign (–) means lower acceptability. All differences are significant at the 0.01 level, except those marked with a question mark (?), which are not significant even at the 0.10 level.

that the overall differences between all other systems and Anapron are significant at the 0.01 level. In particular, Anapron outperformed the two versions of NETtalk, but the commercial systems and humans did better than Anapron.

Because there was an interaction between system and name frequency, saying that a system performs better or worse than Anapron overall does not tell the whole story. We also need to compare the systems on each individual frequency category. This was done in analogous fashion to the overall comparison: for each frequency category, we ran an ANOVA (2-way, this time) to test for any significant differences, and — since the result was positive in all cases — we followed it up with a Bonferroni comparison to localize the inter-system differences. The results are included in Table 5. Some of the differences suggested by the overall results cannot be detected as significant for individual frequencies, partly because we only have one fourth of the data to work with. In particular, we could not detect significant differences from TTS for ultralow-frequency names, nor from DECvoice for mid to low-frequency names.

4.4 Extrapolated performance

The previous section identified the significant differences between Anapron and other systems, both across the whole test set, and within particular frequency ranges. What the analysis did not cover, however, was the impact of these differences on system performance in practice. In practice, differences on commonly-occurring names are correspondingly more important than differences on rarely-occurring names. To get an idea of how the observed differences translate into real-world behavior, we now estimate the performance of each system on the distribution of names in the full Donnelley corpus.

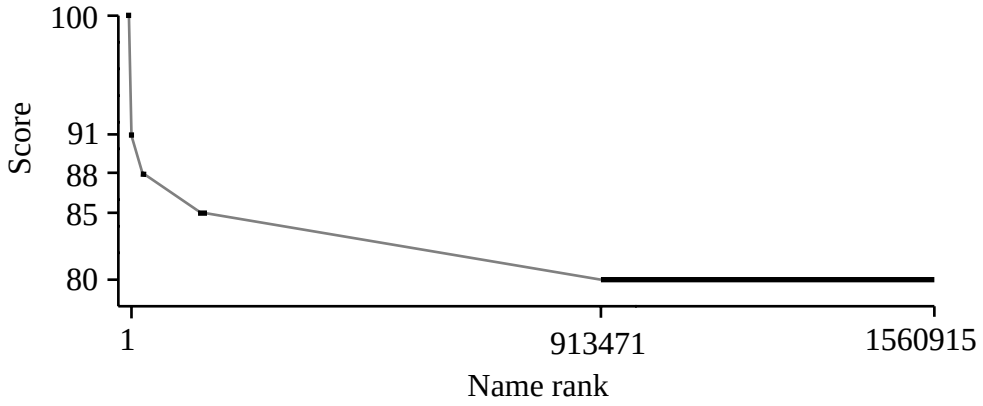

Figure 2: Estimate of Anapron's performance as a function of name rank in Donnelley. The black segments correspond to measured (or assumed) performance. The gray segments are linear interpolations between the black segments.

The estimates were obtained as follows: for each system, we start with four (F, S) data points. The value F is a name frequency, and S is the system's score on names of that frequency. The four points correspond to the four frequency categories tested. To cover the very high frequencies, we add a fifth point, $(676080, 100\%)$, which assumes that every system's performance approaches 100% for the top-ranking names. This is realistic to the extent that system designers will have done whatever is necessary to make their systems produce acceptable pronunciations for these often-encountered names. Given these five data points, we can construct a partial function from name rank to system performance. Each (F, S) point gives rise to a horizontal segment in this function, connecting (R_1, S) and (R_n, S), where R_1 and R_n are the ranks of the first and last names in Donnelley of frequency F. To fill in the gaps between these segments, we do simple linear interpolation. Figure 2 shows the complete function for Anapron.

We will denote this function from name rank to score as score(R) for a system. We then estimate the performance of the system for all of Donnelley as a weighted average of score(R), where the weights follow the distribution of names in Donnelley. The formula is as follows:

$$\text{Performance} = \sum_{1 \le R \le 1560915} \text{score}(R)\,\text{freq}(R)$$

where freq(R) gives the frequency distribution for Donnelley. The value of freq(R) was not available for each individual rank, but it was known for 48 rank *intervals*.[7] This gave a way of approximating the formula above. The approximation was applied to the systems in the experiment; Table 6 gives the results. The scores on the 400-name test set are included for comparison. The performance scores for Donnelley are markedly higher than the scores on

[7] Cecil Coker, personal communication, 3/18/91, kindly supplied this information.

System	Donnelley	Test set
Ubound	99	97
TJG	97	93
MJW	97	93
Orator	97	93
TTS	96	89
DECvoice	95	90
Anapron	94	86
BP-block	90	78
BP-legal	85	67

Table 6: Estimated system performance for the full distribution of names in Donnelley. Performance on the 400-name test set is shown for comparison. All scores are percentages of acceptable pronunciations.

the test set because the test set was abnormally difficult. The differences between systems are also smaller, as all systems were assumed to perform comparably (near 100%) on the top-ranking names, which are the dominant ones in the score.

4.5 Reliability of data

As mentioned at the end of Section 3.4, it was evident that some of the judges got tired by the end of the experiment. One effect this may have had on the data is that as the experiment went on, subjects may have become less discriminating about which pronunciations were acceptable — they may have started accepting any pronunciation. We investigated this by breaking down the judges' scores as a function of elapsed time in the experiment. Figure 3 shows the results. There are 14 curves, one per judge. Each curve shows the difference between the judge's score in each quarter of the experiment from that judge's score in the first quarter. A judge's score is the percentage of acceptable ratings assigned by the judge. We would expect these scores to be roughly constant as a function of elapsed time, since the order of names was randomized. Thus the curves should hover around 0. But for two judges in particular — J9 and J13 — there seems to be a marked, almost monotonic increase. Although we have no sound basis for simply dismissing the data from these or any other judges, we tried removing them to see how it would affect the results of Section 4.3. In fact, the results were unchanged. We found the same significant effects in the ANOVAs, and the same significant differences in the Bonferroni comparisons. So subject fatigue does not appear to have had an appreciable impact on the results of this experiment.

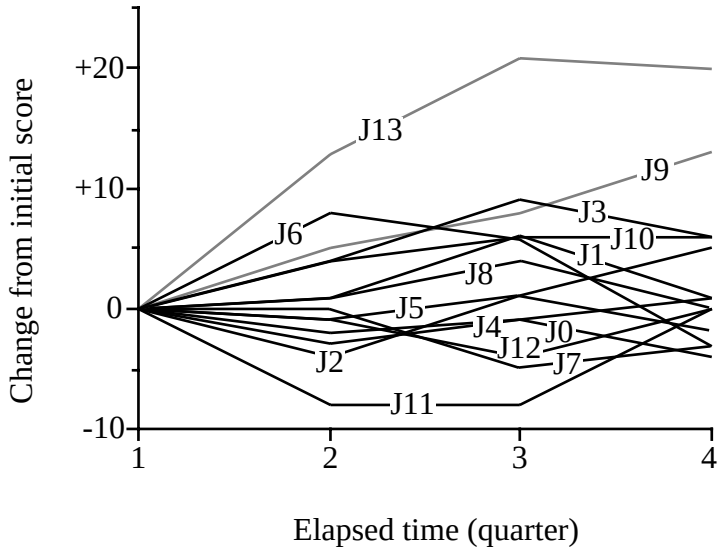

Figure 3: Change in judges' scores as a function of elapsed time in the experiment. There is one curve for each judge, J0 through J13. The curve shows how much the judge's score in each quarter of the experiment differed from that judge's score in the first quarter. A judge's score is the percentage of acceptable ratings assigned by the judge. The curves for judges J9 and J13 are drawn in gray to highlight the fact that they are outliers.

5 Conclusion

An experiment was presented comparing a new name-pronunciation system, Anapron, with seven other systems. Anapron works by a combination of rule-based and case-based reasoning. It is based on the idea that it is much easier to improve a rule-based system by adding case-based reasoning to it than by tuning the rules to deal with every exception. In the experiment described here, Anapron used a set of rules adapted from MITalk and elementary foreign-language textbooks, and a case library of 5000 names. With these components — which required relatively little knowledge engineering — Anapron was found to perform almost at the level of the commercial systems in the experiment. For some ranges of name frequency, a significant difference between Anapron and certain commercial systems could not be detected. Anapron was also found to perform substantially better than NETtalk, even with Dietterich's enhancement of block decoding.

References

Basson, S., D. Yashchin, K. Silverman, and A. Kalyanswamy, 1991. Assessing the acceptability of automated customer name and address: A rigorous comparison of text-to-speech synthesizers. In *Proceedings of AVIOS*.

Coker, C. H., K. W. Church, and M. Y. Liberman, 1990. Morphology and rhyming: Two powerful alternatives to letter-to-sound rules for speech synthesis. In *Conference on Speech Synthesis*, European Speech Communication Association.

Conroy, D., T. Vitale, and D. H. Klatt, 1986. *DECtalk DTC03 Text-to-Speech System Owner's Manual*. Educational Services of Digital Equipment Corporation, P.O. Box CS2008, Nashua, NH 03061. Document number EK-DTC03-OM-001.

Dietterich, T. G., H. Hild, and G. Bakiri, 1990. A comparative study of ID3 and backpropagation for English text-to-speech mapping. In *Proceedings of the 7th IMLW*, Austin, Morgan Kaufmann.

Fleiss, J. L., 1981. *Statistical Methods for Rates and Proportions*. John Wiley and Sons.

Freeman, M. F., and J. W. Tukey, 1950. Transformations related to the angular and the square root. *Annals of Mathematical Statistics*, 21.

Golding, A. R., 1991. *Pronouncing Names by a Combination of Rule-Based and Case-Based Reasoning*. PhD thesis, Stanford University.

Golding, A. R. and P. S. Rosenbloom, 1991. Improving rule-based systems through case-based reasoning. In *Proceedings of AAAI-91*, Anaheim.

Hunnicutt, S., 1976. Phonological rules for a text-to-speech system. *American Journal of Computational Linguistics*. Microfiche 57.

Kendall, M. and J. D. Gibbons, 1990. *Rank Correlation Methods*. Edward Arnold, London. Fifth edition.

McClelland, J. L. and D. E. Rumelhart, 1988. *Explorations in Parallel Distributed Processing*. The MIT Press, Cambridge, MA. Includes software for IBM PC.

Miller, R. G. Jr., 1981. *Simultaneous Statistical Inference*. Springer Verlag, New York.

Sejnowski, T. J. and C. R. Rosenberg, 1987. Parallel networks that learn to pronounce English text. *Complex Systems*, 1.

Spiegel, M. F., 1985. Pronouncing surnames automatically. In *Proceedings of AVIOS*.

Spiegel, M. F. and M. J. Macchi, 1990. Synthesis of names by a demisyllable-based speech synthesizer (Orator). *AVIOS Journal*, 7. Special RHC/RBOC issue.

Vitale, A. J., 1991. An algorithm for high accuracy name pronunciation by parametric speech synthesizer. *Journal of Computational Linguistics*, 17(3).

A DECtalk notation

Following is a description of the relevant portions of the pronunciation notation used by DECtalk [Conroy *et al.*, 1986].

Vowels

aa	father
ae	bat
ah	but
ao	bought
aw	bout
ax	about
ay	bite
eh	bet
ey	bake
ih	bit
iy	beat
ow	boat
oy	boy
rr	bird (stressed)
uh	book
uw	boot
yu	cute

R-Colored Diphthongs

ar	bar
er	bear
ir	beer
or	bore
ur	poor

Syllabic Consonants

el	bottle
en	button
rr	butter

Stress[8]

´	primary stress
`	secondary stress

Consonants

b	bin
ch	chin
d	debt
dh	this
f	fin
g	give
h	head
jh	gin
k	cat
l	let
m	met
n	net
nx	sing
p	pin
r	red
s	sit
sh	shin
t	test
th	thin
v	vest
w	west
yx	yet
z	zoo
zh	measure

Allophones

ix	kisses (reduced ih)
dx	rider, writer (alveolar flap)
lx	bell
q	we_eat (glottal stop)
rx	oration
tx	Latin

[8]Stress values will sometimes be written as numbers above the vowels. A 1 means the vowel has primary stress, 2 means it has secondary stress, and 0 means it is unstressed.