TWENTE WORKSHOP
on LANGUAGE TECHNOLOGY

TWLT 8

Speech and Language Engineering

L. Boves & A. Nijholt (eds.)

Speech and Language Engineering

Proceedings of the eighth
Twente Workshop on Language Technology

L. Boves & A. Nijholt (eds.)

CIP GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Boves, L., Nijholt, A.

Speech and language engineering:
Proceedings Twente Workshop on Language Technology 8 / L. Boves, A. Nijholt,
Enschede, Universiteit Twente, Faculteit Informatica

ISSN 0929-0672

trefw.: natural language processing, speech, language engineering, evaluation

Book orders:
University of Twente
Ch. Bijron
Dept. of Computer Science
P.O. Box 217
NL 7500 AE Enschede
fax: +31-53-315283
Email: bijron@cs.utwente.nl

Druk- en bindwerk: Reprografie U.T. Service Centrum, Enschede

PREFACE

TWLT is an acronym of Twente Workshop(s) on Language Technology. These workshops on natural language theory and technology are organised by Project Parlevink (sometimes with the help of others), a language theory and technology project conducted at the Department of Computer Science of the University of Twente, Enschede, The Netherlands. Each workshop has proceedings containing the papers that were presented. For the contents of these proceedings consult the last pages of this volume.

Previous workshops.
TWLT1, *Tomita's Algorithm: Extensions and Applications*. 22 March 22, 1991.
TWLT2, *Linguistic Engineering: Tools and Products*. 20 November, 1991.
TWLT3, *Connectionism and Natural Language Processing*. 12 and 13 May 1992.
TWLT4, *Pragmatics in Language Technology*. 23 September, 1992.
TWLT5, *Natural Language Interfaces*. 3 and 4 June, 1993.
TWLT6, *Natural Language Parsing*, 16 and 17 December, 1993.
TWLT7, *Computer Assisted Language Learning*, 16 and 17 June 1994.

TWLT8 was devoted to speech, the integration of speech and natural language processing and the application of this integration in natural language interfaces. The workshop was organized under auspices of the Dutch NWO Priority Programme on Speech and Language, the Special Interest Group on Parsing Technologies (SIGPARSE) of the Association of Computational Linguistics (ACL) and the Centre of Telematics and Information Technology (CTIT) of the University of Twente. The workshop was sponsored by PTT Research, Leidschendam, Getronics Software, Amsterdam and the NWO Prioriteitsprogramma Taal- en Spraaktechnologie. It took place in the Vrijhof at the campus of the University of Twente in Enschede, The Netherlands. Just as with the previous workshop programs there were presentations by a select group of international researchers and other experts.

A workshop is the concerted action of many people. It goes without saying that we are grateful to the authors and the organisations they represent for their efforts. But in addition we would like to mention here the people whose work has been less visible during the workshop proper, but whose contribution was evidently of crucial importance. Charlotte Bijron and Alice Hoogvliet-Haverkate took care of the administrative tasks. René Steetskamp assisted us in making these proceedings fit for printing. Finally we also wish to thank the participants for being there and for contributing to the discussions.

We hope that TWLT9 on *Corpus-Based Dialogue Modelling*' in May/June 1995 will match the success of this workshop.

November, 1994

Loe Boves
Anton Nijholt

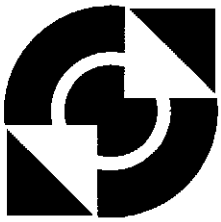
CONTENTS

Workshop Papers:

Sponsors and Support

We gratefully acknowledge help from:

University of Twente, Enschede



PTT Research, Leidschendam

Getronics Software, Amsterdam

Getronics
Software

Prioriteitsprogramma Taal- en Spraaktechnologie, NWO

The North American Business News Task: Speaker Independent, Unlimited Vocabulary Article Dictation.

Chr. Dugast

Philips GmbH Research Laboratories Aachen, P.O. Box 1980, D-52021 Aachen, Germany

ABSTRACT

The North American Business (NAB) news task will be described with its accompanying Hub and Spoke paradigm that allows fair and interesting method comparison along different axes of Speaker-Independent (SI), Continuous-Speech Recognition (CSR) research work. The NAB task allows to train acoustic-phonetic models on more than 60 hours of speech material and to build language models with more than 240 Million words. Different boundary conditions are set up to allow for example non-native speaker adaptation or microphone independency. This years test (november 1994) focused on unlimited vocabulary recognition. The paper presents also the work done at Philips Research to take part at the 1994 evaluation.

INTRODUCTION

The Wallstreet Journal (WSJ) task is dead; Long living to the North American Business (NAB) news task. Time is flying and the border line of what seems feasible with state-of-the-art Speaker-Independent (SI) Continuous-Speech Recognition (CSR) technology is pushed every year a little bit further: From the speaker-dependent, than speaker-independent Resource Management task (1,000 words with word-pair grammar) at the end of the 80's, to the speaker-independent 5,000 words WSJ task (bigram language model) in 1992, further in 1993 with 20,000 words and trigram language model up to the 1994 cru with unlimited vocabulary and penta-gram language models on articles read from 5 different business news sources, CSR technology is also flying.

This paper will present the Hub and Spoke paradigm as it was discussed between participants of the evaluation. A section will be devoted to what makes benchmarking attractive. Then the base technology developed at Philips Research will be presented before describing what has been done this year to participate at the evaluation.

WHAT IS THE NAB TASK?

Since 1986, ARPA (Advance Research Project Agency, USA) has organized periodic formal evaluations of Continuous Speech Recognition (CSR) technology.

From the beginning, these evaluations were distinguished by well-defined tests that were required of all participants within a specified window of time. Most importantly, the test definition remained stable over several years so that sustained effort could be made toward improved performance and so that any improvement made over time could be demonstrated convincingly. These were important features of a series of evaluations based on the well-known Resource Management (RM) corpus. Those evaluations were highly regarded for the competitive stimulus they produced, resulting in the rapid assimilation of new techniques across the CSR community worldwide.

When the ARPA CSR community began in 1992 designing a test bed corpus for large-vocabulary recognition to replace RM, one of the shortcomings addressed was its lack of support for the variety of important research interests that existed within the community at the time. Active research was already underway in adaptation to speaker, domain, dialect, and in compensation for mismatch in microphone, environment, and speaking style but none of this was supported by the RM corpus.

The ARPA-charted CSR Corpus Coordinating Committee (CCCC) was given the task of defining a corpus and specifying an evaluation schema that would take advantage of this diversity and drive it to produce enabling technology for eventual application to real-world CSR problems.

The Hub and Spoke evaluation paradigm was conceived first in 1993 to accommodate the research requirements of this diverse community and produce convincing demonstrations of technological capability. Tests were defined, to exercise the primary interests of all participants, and to include important comparisons needed to

make informed decisions about the efficacy of a particular algorithm or general approach. At the same time, the evaluation preserved the important controlled baseline test, characteristic of past ARPA-sponsored evaluations, that permitted direct comparison of CSR technology across different systems.

After the successful implementation of the paradigm in 1993, the CCCC was again called to organize it anewed for 1994. More data was to be made available to the participants, the task being not only limited to articles of the Wallstreet Journal but extended to 5 different American newspapers, the so called North American Business news (NAB made of Washington Post, New-York Times, Los Angeles Times, Dow Jones Information Services and Reuters North American Business Report). The texts to be read for the evaluation should not go through a vocabulary filter any more, the vocabulary for recognition being unlimited.

In the next section, we describe the general design of the Hub and Spoke evaluation paradigm. Each component test in the 1994 evaluation is then shortly presented.

The Hub and Spoke Evaluation Paradigm

The Hub and Spoke evaluation paradigm [1] implies an array of fairly independent tests (the Spokes) coupled to a central test (the Hub) in some informative fashion. The Hub test is further distinguished by being an abstract representation of a fundamentally important problem in CSR and by being the only test required of all participants in the evaluation. It forms the basis for all informative *inter*-system comparisons.

The Spoke tests, on the other hand, are abstractions of problems of somewhat less central importance in CSR and evaluation on them is optional. The Spoke tests can be informatively compared to the Hub test to calibrate the difficulty of the problem, but they are otherwise independent. The Spoke tests are specifically designed to permit *intra*-system comparisons of algorithms and methods for problems involving mismatches between training and test data.

Each Hub or Spoke test consists of a primary condition (designated the P0 condition) and several contrastive conditions (designated CX, X = 1,2,..). In general, the primary tests are unconstrained with respect to the lexicon and acoustic or language model (LM) training allowed. The purpose of the primary condition in each test is to showcase an algorithmic or procedural solution to a problem in CSR.

The contrast conditions are designed to expose the effectiveness of the algorithms or methods used in the primary test and to calibrate the difficulty of the problem or data. The first contrast test (designated C1) normally specifies that an adaptive or unconstrained feature of the primary test be disabled or constrained so that the effect of the primary feature can be measured in isolation. This contrast is usually required. Additional contrastive tests (either required or optional) may be specified to calibrate the data or evaluate the featured algorithm on additional data.

1994 Hub and Spoke Test Descriptions

The Hub and Spoke evaluation paradigm was first used for ARPA CSR evaluation in November, 1993. The entire test suite for this evaluation consisted of 2 Hub tests and 9 Spoke tests. In 1994, the 2 Hubs were revisited and only 5 Spokes survived and 2 new ones were introduced. Each of these tests is described below.

The abstract problem represented by all the tests in the 1994 evaluation was the dictation of news stories, with an emphasis on financial news stories. Most of the tests in the 1994 evaluation used speech data from subjects reading diverse articles from the 5 different NAB newspapers mentioned above. Typical tests used 20 subjects reading 15-20 sentences each. Each test had equal numbers of male and female subjects. The primary microphone was the Sennheiser HMD-410.

Unless otherwise noted, the default side information given to the system was as follows. Speaking style, general environment conditions (quiet or noisy), and microphone identity were known. Speaker gender, specific environment conditions (room identity), session boundaries and utterance order were unknown.

The 1994 Hub

The Hub for 1994 was split into two tests differing in recording conditions (office quality and telephone quality recordings).

H1. Unlimited vocabulary read NAB news baseline

The paramount Hub test (H1) was designed to measure state-of-the-art performance on an unlimited-vocabulary SI test, using clean test data that was well-matched to the training data. The prompting texts for the H1 test were not

filtered any more like in 1993 where they were selected from a pre-defined 64K-word text pool which excluded articles that contained words outside the 64K most frequent from the WSJ0 corpus. In 1994, the prompting texts were extracted from NAB articles, in roughly equal portions from the 5 newspapers previously mentioned.

The primary H1 (H1-P0) test allowed any language model (LM) or acoustic training data to be used. In addition, the temporal order of the utterances and the location of subject-session boundaries in the utterance sequence was given to encourage the use of unsupervised incremental adaptation techniques.

To permit direct comparisons of acoustic modeling technology between different systems, the H1 test contained a required contrastive test (H1-C1) that controlled the amount of training data and specified the LM statistics. This contrast was run as a static SI test, so utterance order and session boundaries were not given to the system.

For H1-C1, the acoustic training data was limited to 37.2K utterances (62 hours of speech) drawn from one of two segments of the combined WSJ0 and WSJ1 corpora. One segment was made up of speech data from 284 subjects (designated the short-term speakers) who produced 100-150 utterances each. The other segment had 37 subjects (long-term) who produced either 600 or 1200 utterances each. Participants were free to choose which acoustic training corpus to use.

The common required LM specified for the H1-C1 test was produced by Rosenberg at Carnegie Mellon University. It was a 3-gram backoff LM estimated from 247M words of text from a 3-year WSJ0 text corpus (1987-1989) of 121M words added by 115M words from Agency Press and 11.6M words from the San Jose Mercury. Its lexicon was defined as the 20K most frequent words in the corpus, hence, the test contained some words outside the vocabulary.

An optional contrast test, H1-C2, was specified as an extension of the H1-P0 where supervised adaptation was allowed.

The 1994 Spokes

There were 7 Spoke tests in the 1994 evaluation that were designed to support the major interests of the participating sites at the time.

Spoke S0 as a 5K word test is intended for calibration of systems used in other 5K Spokes (S3, S4, S5 and S10). Spoke S2 supported problems in LM adaptation primarily. Stories on non business news (for example AIDS) are to be recognized. A small corpus of 10,000 words on the same subject is given to adapt LM. Spokes S3 and S4 were targeted at speaker adaptation methods, S3 being particularized to non-native speaker adaptation. Adaptation to different microphones was the focus of Spoke S5. Noise recorded in a car traveling at 55 mph with closed windows and air-conditioning turned on has been digitally added to read speech to allow noise reduction in Spoke S10. Spoke S9 looked at data from a potential application for large-vocabulary CSR – spontaneous dictation of news stories from print-media journalists.

All Spokes except S2 and S9 used read-speech from the WSJ0 5K-word prompting texts. All Spokes except S5 used data from the Sennheiser microphone.

WHY BENCHMARKING?

In the first trial of the Hub and Spoke evaluation paradigm in November, 1993, 11 research sites participated, including 3 sites from Europe. In November 1994, 15 sites participated from them 4 European laboratories. In 1993, 5 sites from the 11, in 1994, 7 sites from the 15, did not receive any funding from ARPA to evaluate their technology. These figures show an increasing interest for participating at the evaluation, although, participation implies, for each site, to accept a very important work load.

What is so attractive in comparing technologies? The aim of such a benchmarking is not to win a competition but to compare methods to solve problems related to speech recognition. The Hub and Spoke paradigm in giving very strong constraints for the different test conditions allows a *glass box* comparison of methods.

The result is a rich array of comparative and contrastive results on several important problems in large vocabulary CSR, all calibrated to the current state-of-the-art performance levels. A complete listing of the numerical results for 1993 can be found in [19]. For interpretive results, the interested reader should consult the contemporary papers of the participating sites.

It is important to remember that the only tests for which fair and informative comparisons can be made across systems (and sites) are the controlled C1 contrasts for either of the two Hub tests. All other tests are designed to produce informative comparisons only within a given system run in two contrastive modes. So in general, only within-system comparisons should be made on the Spoke tests.

The Hub and Spoke evaluation paradigm appears to have met the competing requirements of

supporting the variety of important research interests within the ARPA CSR community while providing a mechanism to focus that work into well-defined and competitively charged evaluations of enabling technology. If it is to be successful, however, it will need to sustain that focus over time in a manner analogous to the very successful Resource Management based evaluations of the late 1980's.

PHILIPS BASE TECHNOLOGY

This section is going to present the base technology used for as well our products as for benchmarking. It will be reduced to the main points, references are given for the reader in need of more information. This section may also be skipped.

ARCHITECTURE OF THE RECOGNITION SYSTEM

Fig. 1 presents a block diagram of the system architecture. In the pre-processing step of acoustic analysis, the speech signal is transformed into a sequence of acoustic vectors $x_1, ..., x_T$ (over time $t = 1, ..., T$). As the speech signal, and thus this sequence of observations, is highly variable, a statistical approach is used to model its generation. Statistical decision theory tells that in order to minimize the probability of recognition errors, one should decide for the word sequence $W = w_1, ..., w_N$ (of unknown length N) that maximizes [2]

$$Pr(w_1, ..., w_N | x_1, ..., x_T) = \frac{Pr(x_1, ..., x_T | w_1, ..., w_N) \cdot P(w_1, ..., w_N)}{Pr(x_1, ..., x_T)} \quad (1)$$

As the denominator is constant for a given observation, this amounts to finding $w_1, ..., w_N$ that maximizes

$$P(w_1, ..., w_N) Pr(x_1, ..., x_T | w_1, ..., w_N) \quad (2)$$

The first term, the a-priori probability of word sequences $Pr(w_1, ..., w_N)$, is independent of the acoustic observations and is completely specified by the language model. It reflects the system's knowledge of how to concatenate words of the vocabulary to form whole sentences and thus captures syntactic and semantic restrictions.

The acoustic-phonetic modeling is reflected by the second term. $Pr(x_1, ..., x_T | w_1, ..., w_N)$ is the conditional probability of observing the acoustic vectors $x_1, ..., x_T$ when the words $w_1, ..., w_N$ were uttered. These probabilities are estimated during the training phase of the recognition system.

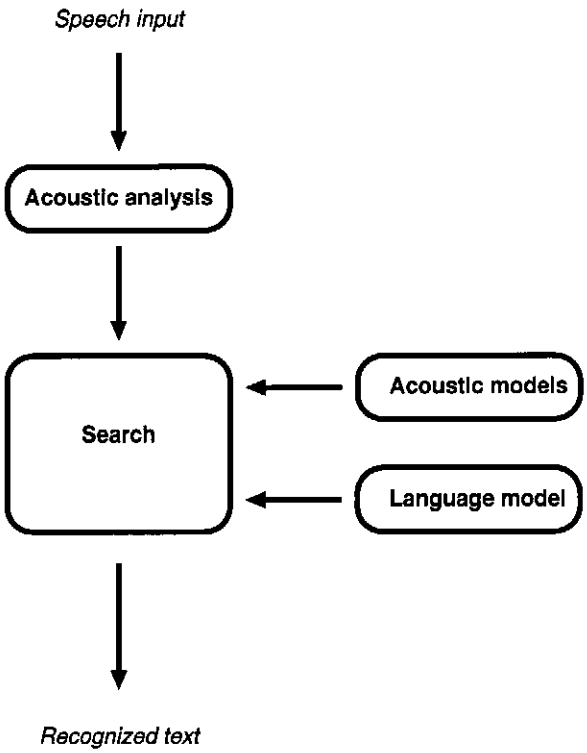


Figure 1: Architecture of a Continuous Speech Recognition System.

A large-vocabulary system is typically based on subword units like phonemes, which are concatenated according to the pronunciation dictionary to form word models.

The decision on the spoken words must be taken by an optimization procedure which combines information of the language model and of the acoustic model, the latter being based on the phoneme models and the pronunciation dictionary. The optimization procedure is usually referred to as search in a state space defined by the knowledge sources.

ACOUSTIC-PHONETIC MODELING

The acoustic conditional probabilities $Pr(x_1, ..., x_T | w_1, ..., w_N)$ are obtained by concatenating the corresponding word models, which again are obtained by concatenating phoneme models according to the pronunciation lexicon. We use inventories of 40-50 phoneme symbols including symbols for silence and glottal stop (For English, triphones are used as basic units). As in many other systems, these subword units are modeled by stochastic finite-state automata, the so-called Hidden Markov Models (HMMs) [2][3][4].

For each state s of the HMM, there is an emis-

sion probability density $q(x_t|s)$ of generating the vector x_t. No pronunciation variants are used in the pronunciation lexicon, such that the emission distributions have to model deviations from the standard pronunciation as well as coarticulatory effects. Our best results were obtained for continuous mixture densities

$$q(x_t|s) = \sum_k c_k(s) b_k(x_t|s) \qquad (3)$$

$$with \quad 0 \leq c_k \leq 1 \quad and \quad \sum_k c_k(s) = 1$$

where the so-called component densities $b_k(.|.)$ are unimodal densities such as Gaussians or (as in our system) Laplacians:

$$b_k(.|.) = \prod_n \left(\frac{1}{2v_n}\right) exp\left(-\sum_n \frac{|x_t(n) - r_{k,s}(n)|}{v_n}\right) \qquad (4)$$

n is the index of the vector components. Each density is completely specified by its location vector $r_{k,s}$. The vector of absolute deviations, $(v_1, ..., v_N)^t$, is assumed to be independent of both the component densities and the states and thus serves as an overall scaling for the acoustic vectors.

In contrast to other systems, the Viterbi criterion is used both in training and recognition. This applies even to the level of mixture components, such that the sum over the component densities in eq. 3 is replaced by their maximum [5][6].

Language Modeling

The language model provides, for each word sequence, an estimate of probabilities $Pr(w_1, ..., w_n)$ usually expressed by m-gram models which have established themselves as both a good way to reliably estimate the parameters and to keep them limited so they can be stored and retrieved. In view of the sizes of available corpora, we typically use word bigram models or a category-based bigram models (bigram class models) with automatically generated classes [7]. An overview about more general techniques in language modeling can be found in [8].

While maximum-likelihood estimation would suggest to take relative frequencies of bigram counts, it is common knowledge that these are particularly bad as estimates and that smoothing is important. The smoothing method that we use is different from those used in other systems and is explained in [9] and [17]. With this method, we achieve better results than with backing-off [10] or linear interpolation.

The Search Procedure

Time-synchronous beam search has been successfully used in the Philips continuous speech recognizer for several years [11]. We found that it is efficient also for 10K or more words [12]. First, all knowledge sources are available at the same level in the integrated search. Second, all hypotheses refer to the same acoustic vector sequence in time-synchronous search. These two key points allow a drastic reduction of the actual search space by pruning less promising hypotheses. As an example, in connection with the WSJ benchmarking, we could recently increase the vocabulary size up to 45K words with a minimal increase of the search space [17].

Tree Lexicon: When the lexicon is large, e.g. above 1K words, it is more efficient to arrange the pronunciation lexicon as a tree of phonemes (tree lexicon). The compression factor for the tree lexicon as compared to a linear lexicon is even surpassed by the reduction in the number of active states, because most of the active states are located in the word beginnings (near the tree's root).

Unfortunately, the tree organization of the lexicon has an undesired consequence for the organization of the search space. In contrast to a linear lexicon, the word identities are unknown at the word beginnings. Particularly for a bigram language model, this means that separate tree copies have to be held, depending on the predecessor word.

Different techniques (language model look-ahead, histogram pruning) have been implemented in the search algorithm to overcome this drawback and to use efficiently the tree structure of the lexicon [17]. A detailed discussion with experiments is also given in [12].

NAB SYSTEM

Training data (as well speech as text) is made available to all partners wishing to participate at the benchmarking. For the base contrastive test within the Hub1 (H1-C1), the amount of speech data allowed for training is controlled, either 284 speakers (SI-284), each speaker having spoken at most 150 sentences, or 37 speakers (SI-37), each speaker having spoken up to 1200 sentences. In both cases, the amount of available recordings sums up to about 62 hours of speech and ... 15 hours of silence.

For language modeling, the amount of text made available totalized 247 Million words, making a vocabulary of 476K words (number of dis-

tinct words in the corpus); 69 Million different trigrams and 16 Million different bigrams occur in the corpus.

ACOUSTIC-PHONETICS

State-tying [21] has been pointed out a good technique to take advantage of the 9552 within-word triphones identified in the training corpus (SI-284). Actually, 4644 different within-word triphones, covering about 99.5% of all triphones pronounced have been modeled. Without state-tying, training 4644 different triphones would have meant to train 13932 mixtures. State-tying allowed to reduce the number of mixtures to be trained by a factor of 3, so that *only* 4000 mixtures were trained. Each mixture had in average 60 densities, making a total of 250,000 sex-dependent densities.

It is known that between-word triphones play an important role, especially for the recognition of short words. To take the decision to model between-word triphones in addition to within-word triphones leads to the discussion on how complex the recognition software should be. At Philips, where we have to think of our PC-based dictation system, the discussion is not closed and between-word triphone modeling still not implemented.

LANGUAGE MODELING

The 1994 NAB news evaluation was characterized by two new factors: the availability of a tremendous amount of texts coupled with no vocabulary filtering. What would be the quality of the new corpus (200 Million words collected in less than a year)? What would be the reduction of the perplexity with so much new data, increasing the corpus size from 40 Million words to 247 Million words? What would be the out-of-vocabulary (OOV) rate?

Concerning the quality, it appeared that although the data collection has been processed automatically (what indubitably leads to errors), the whole corpus was tractable. The bigram perplexity could be reduced by 10%, the trigram perplexity by 20%; For a vocabulary of 20K, the OOV rate increased only from 1.7% to 2.7% (that means that 20,000 words from the 476K different words cover 97.3% of the corpus, the remaining 2.7% being distributed within 456K different words!). Furthermore, quadri-gramms as well as penta-gramms could be envisaged.

A key point was then to decide on the size of the vocabulary. All sites came out with a vocabulary of about 60K words. Philips decided for a 64K vocabulary with an OOV rate of 0.6%. Most of the OOV are proper names: for example Clinton would have not appear in our 64K lexicon if we were to make tests on 1989 articles. Decision on which words had to belong to the vocabulary was a weighted measure of frequency and period of appearance, a word just under the frequency limit but appearing only during the weeks before articles were selected for the evaluation had good chances to come in the vocabulary.

SPEAKER ADAPTATION

The primary test of the Hub (A1-P0) allowed unsupervised adaptation. Although each speaker read in average only 15 sentences (around 400 words) a very simple acoustic-phonetic incremental adaptation scheme has been successfully developed at Philips. After having recognized each sentence, the optimal path given by the recognition procedure is taken to evaluate new densities which are linearly interpolated with the old ones. A weight of 20% has been allocated to the newly evaluated density. This adaptation scheme is incremental, ie after each sentence the knowledge on the speaker is increasing. Even though each speaker read in average only 15 sentences, comparing word error rates with a truly speaker independent system, the relative word error rate decrease due to speaker adaptation was as high as 5%.

SUMMARY

The Wallstreet Journal (WSJ) task has been extended to 5 other news sources what let rebaptize the task to North American Business (NAB) news. In doing so, the 64K word filtering of the WSJ has been given up. The task has gained in generality to come closer to a truly open task. No official results for the 1994 evaluation are known at printing time. Most of them will be published at ICASSP 1995 in Detroit, where each participating site will give their interpretation of the results.

From the research point of view, the HUB and Spoke Paradigm allows contrastive comparisons of systems along different research axes like speaker adaptation or microphone independence. Most importantly, the definition of the Hub with a contrast (H1-C1) and a primary (H1-P0) test allows glas box comparisons of the different technologies involved in speech recognition.

The increasing number of non funded sites competing on the task shows the real interest of

research groups to take part to the discussion that goes along with participation. It is in our opinion the only way to push the technology.

What will be the next test? This will be discussed, but telephone quality, spontaneous speech, domain independency might be axes along which the NAB community is willing to go.

REFERENCES

1. F. Kubala and CCCC: *The Hub and Spoke Paradigm for CSR Evaluation*, Proceedings of the ARPA Human Language Technology Workshop, Morgan Kaufmann Publishers, Mar. 1994.
2. F. Jelinek (1976): *Continuous Speech Recognition by Statistical Methods*, Proc. of the IEEE, Vol. 64, No. 10, pp. 532-556, April 1976.
3. J. K. Baker (1975): *Stochastic Modeling for Automatic Speech Understanding*, in D. R. Reddy (ed.): 'Speech Recognition', Academic Press, New York, pp. 512-542, 1975.
4. S. E. Levinson, L. R. Rabiner and M. M. Sondhi (1983): *An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition*, The Bell System Technical Journal, Vol. 62, No. 4, pp. 1035-1074, April 1983.
5. H. Ney (1993a), *Search Strategies for Large-Vocabulary Continuous Speech Recognition*, Proc. of NATO Advanced Study Institute on Speech Recognition and Understanding, Bubion, Spain, 1993, in print.
6. H. Ney (1993b): *Modeling and Search in Continuous Speech Recognition*, Proc. Europ. Conf. on Speech Communication and Technology, Berlin, pp. 491-500, Sep. 1993.
7. R. Kneser and H. Ney (1993): *Improved Clustering Techniques for Class-Based Statistical Language Modeling*, Proc. Europ. Conf. on Speech Communication and Technology, Berlin, pp. 973-976, Sep. 1993.
8. H. Ney and U. Essen and R. Kneser: *On Structuring Probabilistic Dependencies in Stochastic Language Modeling*, Computer Speech Language, Vol. 8, pp. 1-38, 1994.
9. H. Ney and U. Essen (1991): *On Smoothing Techniques for Bigram-Based Natural Language Modeling*, ICASSP, Toronto, pp. 825-828, May 1991.
10. S.M. Katz: *Estimation of probabilities from sparse data for the language model component of a speech recognizer*, IEEE Trans. on Acoustics, Speech and Signal Proc., Vol. ASSP-35, pp. 400-401, March 1987.
11. H. Ney, D. Mergel, A. Noll and A. Paeseler (1992b): *Data Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition*, IEEE Trans. on Signal Processing, Vol. SP-40, No. 2, pp. 272-281, Feb. 1992.
12. H. Ney, R. Haeb-Umbach, B.-H. Tran and M. Oerder (1992a): *Improvements in Beam Search for 10000-Word Continuous Speech Recognition*, ICASSP, San Francisco, CA, pp. I-9 - I-12, March 1992.
13. M.Oerder, H.Ney: *Word Graphs: An Efficient Interface between Continuous Speech Recognition and Language Understanding*. In *Proc. ICASSP 93*, pp. II-119–II-122, Minneapolis, MN, 1993.
14. M. Oerder, H. Aust (1994): *A Real-Time Prototype of an Automatic Inquiry System*. In *Proc. ICSLP 94* pp. 703–710, Yokohama, 1994.
15. K.S.Fu: *Syntactic Pattern Recognition and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1982.
16. F.Jelinek, J.D.Lafferty, R.L.Mercer: *Basic Methods of Probabilistic Context Free Grammars*. In *Speech Recognition and Understanding*, pp. 345–360, NATO ASI Series F, Springer-Verlag, Berlin Heidelberg, 1992.
17. V. Steinbiss, H. Ney , U. Essen-Willemsen, B.-H. Tran, X. Aubert, Chr. Dugast, R. Kneser, H.-G. Meier, M. Oerder, R. Haeb-Umbach, D. Geller (1994): *Continuous Speech Dictation - From Theory to Practice*, to appear in Speech Communication.
18. Bernstein, J., D. Danielson, *Spontaneous Speech Collection for the CSR Corpus*: Proceedings of the DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, Feb. 1992, pp. 373-378.
19. Pallett, D., J. Fiscus, W. Fisher, J. Garofolo, B. Lund, and M. Pryzbocki: *1993 Benchmark Tests for the ARPA spoken Language Program*, Proceedings of the ARPA Human Language Technology Workshop, Morgan Kaufmann Publishers, Mar. 1994.
20. Paul, D., J. Baker: *The Design for the Wall Street Journal-based CSR Corpus*, Proceedings of the DARPA Speech and Natural Language Workshop, Morgan Kaufmann Publishers, Feb. 1992, pp. 357-362.
21. C. Dugast, P. Beyerlein, R. Haeb–Umbach: *Application of Clustering Techniques to Mixture Density Modeling for Continuous-Speech Recognition*, to appear in the proceedings of the ICASSP 95 in Detroit.

CREATION AND ANALYSIS OF THE DUTCH POLYPHONE CORPUS

P. van Alphen[1], C. in 't Veld[2], W. Schelvis[1],
[1]PTT Research, [2]SPEX,
e-mail P.vanAlphen@research.ptt.nl

ABSTRACT

In this paper the linguistic design, speaker selection, and the recording and transliteration procedures for the Dutch POLYPHONE corpus are described in some detail. Over 5,000 have been recorded. The paper gives details of the distributions of the speakers according to regional and socio-economic background, sex and age of the speakers. Also, first results of the analysis of the linguistic contents of the recordings are reported.

1. INTRODUCTION

Large corpora of speech recorded over the public switched telephone network have become of crucial importance for the progress of Research & Development in speech technology. In the fringe of ICSLP-'92 COCOSDA defined guidelines for such a corpus, which should be recorded for as many different languages as possible. In the course of 1993 these guidelines were made more concrete by the Linguistic Data Consortium, who specified a recording protocol for American English and Spanish. Taking those protocols as a point of departure, PTT Research and the Speech Processing Expertise Centre SPEX set out to record the Dutch POLYPHONE corpus.

This paper starts with a summary description of the Dutch POLYPHONE corpus. First, information is given about the distribution of speakers over cells defined according to regional, social and personal characteristics. It appears that we have not been able to reach our goal of essentially uniform distribution and we explain why this happened. Next we give data about the linguistic contents of the corpus. One of the aims that we had in mind was research into the way in which speakers of Dutch express concepts, like answers to *yes/no*-questions, telephone numbers, dates and times, etc. In this paper first results of in-depth analysis of the spoken replies are presented. The paper ends with some conclusions and recommendations for future speech corpus recording projects.

2. THE DUTCH POLYPHONE CORPUS

The recording workstation used for POLYPHONE was based on an Aculab MVIP/PEB E1/G703 PC Card with 1TR6 isdn-30 signaling (i.e., a primary rate German ISDN 2MB/s connection) for the telephone interface, a Rhetorex Voice Card and driver software, Show-'n-Tel application development software, and a 16 port operational license, in an OS/2 PC. Data files were copied to a Unix network for transliteration and archiving.

The recording platform is set up to store the speech signals in 8 bit A-law coded samples at a sampling rate of 8 kHz. In the Dutch PSTN it is guaranteed that a speech signal with an ISDN connection as its destination remains in an A-Law coded digital form after the first major network switch that it encounters. Thus, the acoustic quality of the recordings is completely determined by the characteristics of the caller's local loop and the background noise in the caller's location. Post-Processing was done at the Dept. of Language & Speech, Nijmegen University, using software running on a PC under MS-Windows, equipped with a Pro-Audio board. The software supporting transliteration was developed jointly by PTT Research and SPEX.

When post-processing for a new speaker begins, the transcriber must first listen to the speaker's identification code, and enter that code into the program. The server maintains a data base with information about the prompts which each speaker has answered. Whenever the answer is predictable (i.e., in all cases where the caller is supposed to read preprinted material), the expected answer is displayed on the screen.

3. SPEAKER SELECTION

Prospective callers were sent a personalized letter. We contracted with a direct mail company, that made an initial random selection of addresses from their data base. In principle, this company would be able to select callers from neighbourhoods which are known to be occupied by a specific socioeconomic group with overrepresentation of certain age groups. However, we never got round to asking them to select such neighbourhoods.

It appeared that the response in this way of approaching prospective respondents was only 15%. Since each letter costs approximately $ 1, we decided that we could not stay within the project budget unless we could increase the response rate considerably. Therefore, we approached a number of acquaintances, who were asked to provide us with address lists of people they knew, so that we could send these persons extra personalized invitations to participate. Although the request for addresses was successful, the plan failed, because the direct mail company that processed these addresses did not include the invitation letter signed by the person who provided the names.

In the final stage of the project subjects were recruited from the personnel of the Dutch PTT. Thanks to intensive publicity and wide press coverage, this increased the response rate to well over 30% in the last group of subjects. In the very last stage of the sampling process we were able to oversample the regions which until then were heavily underrepresented.

Originally, we aimed at collecting 5000 speakers, uniformly divided over a large number of cells, defined according to four criteria, viz. (1) geographical region, (2) socioeconomic status, (3) sex, and (4) age. Geographical region, operationalized as the province in which the speaker lives, is the best practically feasible approximation to regional accent and dialect background. By sampling provinces, we sidestep the unsolved problems of how many different regional accents should be distinguished and how these should be defined. However, the Dutch population is quite mobile; this adds to the difficulty of predicting dialectal background of callers from their present location. Because we think that knowledge about dialectal background is important, we decided to ask the callers to tell us in what part(s) of the country they grew up. Due to the very uneven distribution of the population over provinces it appeared to be practically impossible to get equal numbers of speakers from each province.

Socioeconomic status is difficult to define, and even more difficult to assess reliably from what respondents are willing to say. We decided to approximate status on the basis of the highest education level of the respondents. To avoid complicated and confusing questions, we settled for a division into three education classes, only elementary school, secondary school, and college/university level education. Using hindsight, this decision proved to be somewhat unfortunate: since the mid-fifties youngsters in the Netherlands are obliged to follow classes up to the age of 16, so that practically everybody under the age of 55 has had more than just elementary school. Thus, it is not surprising that we were able to recruit only 256 speakers who said that they had no more than elementary school. 2600 callers said that they have secondary level education, while 2194 claimed college level education.

We distinguish four age classes, i.e., under 20, between 21 and 40, between 41 and 60, and 61 and older. Information about age is acquired by asking the respondents for their year of birth. Since we also set a minimum age of 16 for participation, the under 20 group is necessarily much smaller than the other groups (168 respondents). For similar reasons, the group of 61 and older is underrepresented with 457 respondents. The group between 20 and 40 comprises 2686 speakers, whereas the group between 40 and 60 consists of 1739 callers.

Information about speaker sex is obtained by asking the respondents to say whether they are male or female. The distribution of the speakers with respect to sex and regional background, based on a total of 5,050 respondents, is shown in Table 1.

Table 1: Distribution of respondents over geographical regions

Province	inhabitants (× 1,000)	respondents	
		males	females
Groningen	555.2	139	212
Friesland	601.8	135	164
Drente	445.6	140	120
Overijssel	1,032.4	160	169
Flevoland	232.8	8	88
Gelderland	1,828.8	302	291

Utrecht	1,037.3	215	206
Noord Holland	2,421.7	349	311
Zuid-Holland	3,271.5	464	379
Zeeland	359.2	157	110
Noord Brabant	2,225.3	291	259
Limburg	1,115.5	184	125
Totals	15,128.1	2,616	2,434

4. THE SPEECH MATERIAL

The speech material recorded in the POLYPHONE project consists of 32 read items, 14 extemporaneous answers to printed questions, and 4 extemporaneous answers to questions not printed on the response sheet.
The material **to be read** consists of the following items:
5 digit strings (one telephone number, two bank accounts or credit card numbers, one string of isolated digits, and the participation number)
3 natural numbers
3 guilder amounts
2 city names
4 application words
3 spelled words
1 date
1 time
1 amount
1 postal code
4 sentences with an application word
5 phonetically rich sentences

The following list of *printed* questions is asked:

- Is Dutch your native language?
- Did you ever live outside the Netherlands?
- Would you willing to participate in another study like this one?
- What is your last name?
- What is your house number?
- What is the name of the street you are living?
- What is your postal code?
- In which city do you live?
- In which cities did you grow up?
- Are you a man or a woman?
- What is your age?
- Which code (1, 2, or 3) represents your education level (1=primary school, 2=high school, 3=college/university)?
- Please, say a familiar phone number.
- Please, give your comments about this recording session.

The following *unprinted* questions are asked:

- Please, spell your name.
- Are you calling from your home phone?
- Are you using a cordless telephone?
- What time is it now?

4.1 Construction of the texts

The text material to be read was carefully constructed in order to maximize the coverage in linguistic sense. Numbers have been constructed in such a way that all digits appear approximately with equal frequency, with one restriction: in order to balance the number of *teen* and *ty* forms, the digit *1* had to be overrepresented. Care has been taken to prevent unreasonable combinations of amounts and units, like *1,567,329* mm.
In order to obtain approximately equal numbers of tokens for all letters of the alphabet, a greedy search algorithm was used to selects words from an electronically readable dictionary (provided by CELEX) in such a way that the least frequent letters would occur at least 120 times, while the frequency of occurrence of more frequent would be minimized. In doing so, we ended up with a list of 797 words. Only the words shorter than 11 and longer than 5 characters are selected.
We have created a list containing the names of all train stations in the Netherlands, to which the names of all Dutch communities with more than 5,000 inhabitants were added. The lists of city names was completed by adding all European capitals, and the biggest cities on other continents. Foreign cities are represented by their Dutch names, whenever such a form exists.
We have designed a list of over 1000 words that may be used in applications based on isolated word recognition systems. To enable us to study the effects of context, all these words are also be embedded in sentence. All callers read the application words twice: once isolated and once embedded in a sentence.
There are many ways to write and express dates in Dutch. We want to catch all variations. Therefore, we have printed the dates using a range of different notations, viz. (a) (Monday) 1 August 1996; (b) (Monday) 01-08-96; and (c) (Monday) 1 August '96.
Times can also be printed and expressed in different ways, for example as (literally translated into English): 10:15 a quarter past ten, ten hours fifteen, fifteen past ten; 18:40

eighteen hours forty, six hours forty, ten past half seven, twenty to seven. Here too, the response sheets show a number of different formats.

For each application word a short sentence (minimally four words, totalling less than 80 characters, including blanks) was constructed. Care has been taken to create sentences that would seem reasonable in an Interactive Voice Response application.

We decided to try to record all phonemes of Dutch in as many different phonetic contexts as possible. At the same time, we want to record all phonemes from each speaker. To that end, a large number of sets of five sentences was constructed, in such a way that each set contains all phonemes at least once. Since the frequency of occurrence of phonemes in Dutch is heavily skewed, large numbers of sentences had to be scanned in order to fulfil the requirement. In fact, we have not yet succeeded in constructing enough such sets to be able to offer a different set to each caller. By scanning an electronic newspaper (Trouw) and adding by hand sentences containing the least frequent phonemes, we succeeded so far in collecting 2500 x 5 = 12.500 sentences.

All sentences consist of at least four words, with a maximum of 80 characters. Moreover, all sentences have been checked for possibly offending contents or words. The resulting set of sentences was processed by a grapheme-to-phoneme converter, in order to be able to compose sets of five with all phonemes. The output of our grapheme-to-phoneme converter was checked by hand and corrections were made when necessary.

5. POSTPROCESSING

Postprocessing consists of four steps, viz.

1. word-by-word transliteration of all items,
2. transliteration of extra sounds and noises,
3. collecting demographic data,
4. assessing the quality of all items.

The students who carry out the work are instructed to do the tasks in exactly this order.

5.1 Transliteration

The transliterators are presented with a best guess (in most cases the prompt text printed on the response sheet) of what the speaker has spoken. If the real speech deviates from that text, an efficient editor can be invoked to make all necessary corrections. In transliteration only lower case spellings of conventional lexicalized forms are used. No attempt is made to represent pronunciation differences on a phonetic or phonemic level.

5.2 Extra Sounds

A closed set of extra sounds has been defined, following the guidelines in the American MACRPHONE project. Extra sounds that can be located in time relative to the words are placed accordingly in the transcript. Under this heading background noise accompanying the speech is also marked, as long as the speech is clearly audible; if that is not the case, the response in classified as *Noise* (cf. 5.5).

5.3 Quality Assessment

The last thing the transliterators do is to select a quality indication for each item from a fixed menu, offering the options *O.K.*, *Other*, *Garbage*, and *Noise*. The verdict *O.K.* is given to each response that contains only relevant speech, without overt hesitations, etc. *Other* is assigned to relevant responses that do contain hesitations, self-repairs, stutters, etc. The verdict *Garbage* is given when the caller did respond, but with meaningless speech. Finally, *Noise* was assigned to the items which contained only background noise.

In total, 96.55 of all items was judged *O.K.*, 3.15% got the label *Other*, 0.2% was judged as *Garbage*, and 0.09% as *Noise*. Thus, it can be seen that on average the quality of the recordings is quite high.

6. RESULTS

In this section a number of results of analysing the material actually spoken are presented.

6.1 Telephone Numbers

Two items related to telephone numbers were analysed. The first pertains to numbers read from the response sheet. All these numbers were printed in the same format, i.e., area code, dash, subscriber number (e.g. 020 - 5252183). The second item consists of answers to the question *Please, say a familiar telephone number.* In discussing the results we will use the term *digit* for the words *zero*, *one*, ..., *nine*; the term *number* will denote numbers between \$10\$ and \$99\$.

Presently, the Dutch PTT's number plan has two groups of area codes, one comprising three digits (like 020 in the example above) and one comprising five digits (e.g. 08894). The initial 0 is equivalent to the 1-prefix for area codes in the American telephone network. Subscriber numbers can have from four (only with five digit area codes) to seven digits (with a small number of three digit area codes). Because transliteration does not include intonation markers, it is not possible to discriminate between three and five digit area codes. It is not evident that the transliterators would have been able to disambiguate all answers to the request to give a familiar number.
The format of the read numbers is quite different from the format of spontaneously produced familiar numbers, as can be seen from Table 2.

Table 2: Distribution of responses for telephone numbers

Answer type	read	spontaneous
Only digits	34%	23%
... plus Numbers	92%	91%
... plus Hundreds	100%	99%

Table 3 gives an overview of the size and type of material that could be used for training a connected digit recognizer on the telephone numbers only. In addition to the response types shown in that table a lot of other words were used, but none of these words occurred sufficiently often to make it worthwhile to explicitly account for it in a telephone number recognizer.
It is worth mentioning that 18% of the read and 23% of the spontaneous numbers contain extra sounds, far more often preceding the number than following it.

Table 3: Frequency of occurrence of digit types in the telephone numbers

Number type	read	spontaneous
Digits	29,440	22,977
Numbers	7,593	8,806
Hundreds	529	439
"Double n"	41	36

6.2 Yes/No Expressions

For this paper we analysed the responses to four *yes/no* questions; for two we expected affirmative, and for the remaining ones negative responses. There was a large difference between the two putative affirmative items: Almost 93% of the subjects used a single word *ja, jawel, jazeker* to confirm the fact that Dutch was their native language; that proportion dropped to 75% for the question whether the caller was willing to participate in another recording session. Very few callers said "no" to the latter question, but the way in which they expressed their confirmation was much more varied. Almost all persons who said that Dutch was not their native language explained that their first language was Frisian or a regional dialect. We decided to accept these talkers as effectively native speakers.
83% Of the subjects used a single word *nee, neen* to say that they never lived abroad for an extended period of time. The most obvious explanation for this relatively low number is the large proportion of callers who gave an affirmative answer to this question. 80% of the callers used a single word to deny that they were using a cordless phone; here too, the low proportion of single word answers is mainly due to the large proportion of affirmative replies (over 13% of the callers said they were using a cordless phone).
Since the transliterators had to code the answers to three of the four *yes/no* questions we could check how often an affirmative answer contained a negation and how often the reverse was true. In our data these cases that make the life of a recognizer very difficult were virtually absent: on a total number of 10,702 affirmative responses eight contained a single negation, another eight responses contained two "no" expressions and a single reply contained no less than three negative expressions. Out of 4,525 clearly negative responses only three contained an affirmative expression.
Another observation that is worth mentioning is that politeness forms like *yes, sir; no ma'am* were virtually absent. This may be due to the fact that the *yes/no* questions were located in the last part of the recording session, when the caller should be fully aware that they were talking to a recording machine.

6.3 ZIP-Codes

In the Netherlands ZIP-codes consist of four digits followed by two letters. In principle, all letter combinations can appear. Specifically, a large number of letter pairs which are known as acronyms (e.g. NS: "Nederlandse Spoorwegen "*Dutch Railways*, ME: "Mobiele Eenheid" *Riot Police Squat*) or for which ad hoc acronyms or spelling alphabets are easily invented occur. It was not known in which ways people express ZIP-codes.

For POLYPHONE two ZIP-codes were recorded, one arbitrary which was read and the ZIP-code of the respondent which was given in reply to the prompt "*Please, say your ZIP-code.*" It appears that the way in which ZIP-codes are read differs considerably from the way in which people say familiar ZIPs. In the read ZIPs slightly over 65% of the numeric parts are expressed in the form of two numbers; for the familiar ZIPs this proportion is 74%. Only in approximately 11% of the read forms people use expanded acronyms (e.g. *Nederlandse Spoorwegen* for *NS*), while this proportion is as high as 30% in familiar ZIPs. Less than 1% of the read forms had two letters followed by an expanded acronym, while this proportion was over 2% for familiar ZIPs. Somewhat surprisingly, the number of disfluencies does not differ between read and spontaneous ZIPs.

6.4 Sentences

The nine sentences recorded in POLYPHONE are divided into two groups, i.e., four sentences constructed around application words and five constructed so as to contain all phonemes of Dutch. The first observation that must be made is that the number of *Other* judgments for both sets of sentences is over 7%, whereas that proportion is less than 3% for all other items. Apparently, our subjects had considerable difficulty in reading the sentences aloud, even though they had been encouraged to study the texts before calling the recording platform. The proportion of disfluencies in reading sentences seems to be larger than what was obtained in "Voice Across America" (Wheatly & Picone, 1991). Yet, considerable effort has been spent in selecting and designing sentences for easy readability. Some 12% of the sentences contained extra sounds preceding the first word.

The modal length of the application sentences was 12 words; for the phonetically rich sentences the modal number of words was 11. There were approximately 1,100 different application sentences containing 4000 different words, and 12,500 different rich sentences containing 17,000 different words. The relatively large number of different words is certainly due to the way in which the sentences were constructed or selected.

REFERENCES

Godfrey, J., Graff, D. & Martin, A.(1994) "Public databases for speaker recognition and verification." *Proc. ESCA Workshop Automatic Speaker Recognition, Identification and Verification*, Martigny, April 5-7 1994, pg. 39-42.

Bernstein, J., Taussig, K. & Godfrey, J. (1994) "Macro-phone: An American English Telephone Speech Corpus for the Polyphone Project." *Proc. ICASSP-'94*, Adelaide 19-22 April 1994, pg. I-81 - I-83.

Boves, L., Boogaart,T. & Bos, L. (1994) "Design and recording of large data bases for speaker verification and identification." *Proc. ESCA Workshop Automatic Speaker Recognition, Identification and Verification*, Martigny, April 5-7 1994, pg. 43-46.

Wheatly, B. & Picone, J. (1991) "Voice Across America." *Digital Signal Processing*, Vol. 1, pg. 45-63.

ASSESSMENT OF SPEECH RECOGNITION SYSTEMS:
An overview of performance measures for systems ranging from word recognition to large-vocabulary speaker-independent recognition

H.J.M. Steeneken and D.A. van Leeuwen
TNO Human Factors Research Institute
Soesterberg, The Netherlands

1. INTRODUCTION

Various methods are developed to assess speech recognition systems. These range from methods using a representative data-base to artificial speech generation with highly diagnostic results. These extremes fit into a continuum, assembled by Moore[1990], which ranks assessment methods into five groups based on the use of:

(a) representative data-bases
(b) reference systems
(c) specific calibrated data-bases
(d) diagnostic methods with a specific vocabulary
(e) artificial test signals.

The most frequently used methods belong to group (a) and make use of data-bases collected under conditions representative for the application. In general no control over the specific parameters is obtained.
Reference methods (b) can be based on human recognition or a reference recognizer. Moore [1977] introduced the "Human Equivalent Noise Ratio" (HENR), a measure which expresses the performance of a recognizer in a noise level necessary to achieve comparable human performance. A reference recognition algorithm is proposed by Chollet et al.[1981] and later by Hieronymus et al.[1985]. The Effective Vocabulary Capacity (EVC, van de Vegte and Taylor[1990]) gives the maximum size of a vocabulary that a recognizer can handle with a tolerable small error rate.
These methods make use of a fixed reference: noise, vocabulary or recognition algorithm. However, this limitation may lead to a specific relation with a certain input variable and does not offer a unique comparison. In general no diagnostic information is obtained.
The use of a set of calibrated data-bases (c) is proposed by Peckham et al.[1990]. Each data-base is tailored to a specific environmental condition which affects the recognizer performance. The procedure is based on seven parameters related to the fundamental frequency and spectral aspects. Also the hub and spoke paradigm (Kubala[1994]) in which different parameters are studied separately and the results compared with a reference, is an example of the use of calibrated data bases.
A diagnostic method (d), which was earlier used for intelligibility evaluation, is proposed by Steeneken[1987]. From the analysis of confusions between phonemes, a multi-dimensional representation can be obtained which is related to a set of reference conditions (Steeneken and van Velden [1989a,b]).
A comparable approach used for the evaluation of recognizers is introduced by Simpson et al.[1987a,b]. This test is based on Phonetic Discrimination with one hundred test-words (PD-100). The test-words are constructed on minimal difference phonetic aspects as used with the two alternative rhyme test. Both the correct responses and the false responses are considered.
There is a need to specify the performance of a recognition system or recognition algorithm in a more general way. This can be obtained by estimating the error rates *and* the confusions, as a function of the variation of specific speech production and speech transmission parameters.
For this purpose a test was proposed which is also based on a minimal difference concept but not limited to single phonetic features. It has an "open response" design which includes all possible confusions. This is similar to a common method used for the intelligibility evaluation (Steeneken[1987]). The test-words are of the CVC-word type (Consonant-Vowel-Consonant). The combination of the diagnostic aspects of the CVC-word vocabulary and the systematic, parameter related, data-base generation leads to the

Recognizer Assessment by Manipulation Of Speech (RAMOS, Steeneken and van Velden [1989a,b], and van Velden and Steeneken [1989]).
No assessment methods based on artificial (non-speech) test signals (e) are known for the assessment of speech recognition systems.

The present state-of-the-art of automatic speech recognition technology is focused on systems tailored for use with a large vocabulary (over 20,000 words) and speaker independent. This makes the application of these systems very universal and suitable for Linguistic Research and Engineering projects (Aubert et al.[1994], Ney et al.[1992], Lamel and Gauvain[1992], Woodland and Young [1993]). The underlying technology makes use of language-specific models of the language and lexicon and are therefore focused on a particular language. The majority of these systems are trained for American English based on texts from the Wall Street Journal. The acoustical training is based mainly on speech read from this news paper. The technology push for these large-vocabulary systems is concentrated in an annual assessment session in which all participants test their in-house systems with the same training and test material (Pallett et al.[1994]). The responses of the various evaluated systems are scored by an independent coordinator. This test paradigm, which can be considered as a mixture of (a) and (c), is included in the ARPA (Advanced Research Programme) on Speech and Natural Language programme. ARPA organizes each year a test session in which 15-20 laboratories and industries participate. Application for different languages, such as required within the multilingual European Community, demands separate training for each individual language. The objective of the SQALE project (Speech recognizer Quality Assessment for Linguistic Engineering) is to experiment with establishing an evaluation paradigm in Europe for the Assessment of large-vocabulary, continuous speech recognition systems in a multilingual environment. SQALE is sponsored by the European Commission (DGXIII) within the Third Framework Programme.

2. SELECTION AND ASSESSMENT CONSIDERATIONS

An optimal choice of a recognition system for a certain application requires an inventarization of available systems and of the variables which may affect the recognition performance. It is obvious that the type of recognition (isolated words, connected words, etc.), the vocabulary, and the use (speaker dependency, environment, etc.) play an important role in the performance.
Selection of a system is not an easy job as the market is continuously changing. The NATO research study group on speech processing has established a list of commercial available systems on speech recognition and speech synthesis (Néel[1994]) which can be used as a first step to access the speech technology system market. This list is frequently updated and is freely available.
Selection of an appropriate system requires an inventarisation of the task and system related parameters which lead to one or more candidate systems.
Assessment of a selected system is a major step in the possible integration of a recognition system for an application. Assessment can be performed either in a field experiment under realistic conditions or in the laboratory under artificial conditions. Both methods have their advantages and draw backs such as:
field evaluation: representative vs. uncontrolled, expensive,
laboratory evaluation: reproducible conditions, inexpensive vs. artificial.

Consequently the experimental design of the evaluation test should be based on these aspects.
For an optimal choice a description of the application in a human-system structure is required. We can identify three groups in this structure related to: (a) the speaker, (b) the system, and (c) the application.

(a) With respect to the speaker aspects have to be considered such as: gender, number of users, speaking rate, dialect, native non-native speakers, speech quality, and user related learning effects.

(b) The system also includes the transmission path of the speech signal to the recognizer. Hence, the following aspects in this chain have to be considered: microphone position, environ-mental noise, transmission quality (telephone speech), level variations, and recognizer parameters.

(c) Related to the application the following aspects are relevant: vocabulary size, vocabulary selection, speech type (isolated, connected, spontaneous), syntax rules, acceptable error rate, feedback after recognition, error correction, training, and stability reference patterns.

As a function of all these parameters one can determine the percentage of correctly recognized words. However, we also need to know the number of confusions, rejections, and insertions separately.

For a simple isolated word recognizer the following performance measures, normally expressed as a ratio or percentage, can be determined:
words inside vocab: correctly recognized, rejected, incorrectly recognized,
words outside vocab: rejected (which is correct), false alarms,
confusions: between words.

For connected-word recognizers additional scores for insertions and deletions are obtained.

There are several methods to determine these percentages. For connected word recognition it requires an advanced scoring program which can detect false alarms, insertions, and deletions. In general a dynamic time warping algorithms is used.

Scoring method
From these separate scores a single score can be calculated. This may be the accuracy figure defined as:

$$\text{Accuracy} = \frac{\text{Words Correct} - \text{Insertions}}{\text{Total}} \cdot 100\%$$

or alternatively:

$$\text{Accuracy} = \frac{\text{Total} - \text{Misclassifications} - \text{Deletions} - \text{Insertions}}{\text{Total}} \cdot 100\%$$

A discussion on the scoring is given by Hunt[1989].
Significance between scores of the performance of different recognizers or test conditions can be tested by means of statistical tests such as the analysis of variance ANOVA or the McNemar test[1947]. As for some vocabularies a very low error rate is obtained, the application of a statistical test requires a very high number of trials to get accurate results. In our opinion a more difficult vocabulary or more difficult test conditions would be more adequate.

The performance measures as given above are very dependent on the vocabulary, number of speakers, training etc. A more general measure, independent of the vocabulary, is to determine how human listeners recognize the same vocabulary with the same recognition score but for the condition that the test words are masked by noise. The level of the noise required for an identical score as the recognizer is called the human equivalent noise level (Moore[1977]). Such a noise level opens the possibility to compare results for different vocabularies.

International standardization of assessment methods is a necessity for getting comparable results. Some years ago the already mentioned NATO research study group RSG-10 has established a data-base for isolated and connected digits and for native and non-native talkers. This data-base has been used for many experiments at different locations (Steeneken and Varga[1993], Varga and Steeneken[1993]).

A predictive method was developed (Steeneken[1989a]) where the recognizer performance is specified as a function of the variation of specific speech parameters and environmental conditions. The method uses a small test vocabulary with minimal-difference word-sets of CVC-type words. Training and scoring are according to an open response experimental design. This results in a test environment with a high resolution of the recognition rate versus system performance is different. Additional to this a confusion matrix which quantifies the relative discrimination difference between individual phonemes, providing a valuable diagnostic tool.
This is demonstrated by a practical example given below.

3. Example of the Assessment of a Connected Word Recognizer

The main goal of the study described below relates to the effect of an oxygen mask on the performance of automatic speech recognition systems in the cockpit of a fast jet aircraft (Steeneken and van Velden[1993]). A number of the relevant parameters were investigated: (1) oxygen mask, (2) vocabulary, and (3) noise.

The experiments were carried out with two (commercially available) recognition systems. Both systems are equipped with a noise cancelling system. During the recordings of the speech material, the speaker had a side-tone (feed-back of the speech signal through the helmet-headphones) at a level of approximately 75 dBA. This is a representative condition for aircraft use. It may reduce the so-called Lombard effect which models the increase of the vocal effort of a speaker in the presence of ambient noise at a sufficient level.
A recognizer test bed was used to perform the experiments.

In Table I the mean scores and the corresponding standard errors are given for the investigated combinations of oxygen mask, vocabulary, and noise condition. The results for the words obtained from the AFTI vocabulary (25 control words for a fast jet cockpit) indicate only a small effect of the oxygen mask on the recognition performance. However, the scores for the CVC words (17 identical words with a different initial consonant Cvc, or 15 identical words with a different vowel cVc) show an important effect. As shown in earlier studies the AFTI-test word scores saturate at moderate conditions while the scores for the CVC words are more sensitive to a small deterioration of the speech.
A comparison of conditions with the noise recorded directly (hence, the speaker in ambient noise introducing a Lombard effect) and with additive noise (with the speaker in a silent environment and the noise separately) shows a similar recognition performance.
Hence, even at high ambient noise levels (100 dBA, in combination with the sound attenuation of the pilot helmet and the oxygen mask) no strong Lombard effect occurred and the scores for the direct and additive conditions are quite similar. It should be noted that the effective noise level at the speaker's ear was 80 dBA and that the speaker heard also his own voice by side-tone.
The confusions between CVC test words reflect the confusions between individual phonemes. Depending on the vocabulary, either confusions between vowels or confusions between consonants are obtained. These confusions can be considered as a measure of similarity between the phonemes.
A high similarity between two phonemes indicates that the recognizer could not discriminate between these phonemes.

Table I. Mean recognition scores (m %) and standard errors (se %) based on one speaker, ten test runs, three vocabularies, two recognizers, seven noise conditions and with and without the oxygen mask. The noise was introduced in two ways, directly with the speaker placed in a noise environment (direct) and by addition after the recordings (additive).

			Without oxygen mask			With oxygen mask			
			no noise	direct 80 dBA	add. 80 dBA	no noise	direct 100 dBA	add. 100 dBA	add. 110 dBA
AFTI-sub	recognizer A	mean	95.0	96.0	91.0	96.4	99.6	98.0	88.0
		se	1.0	0.0	3.8	1.6	0.4	0.6	1.3
	recognizer B	mean	100.0	98.0	84.0	93.6	83.6	86.6	61.2
		se	0.0	2.0	5.9	0.7	3.1	2.7	3.6
Vowel in cVc-words	recognizer A	mean	**96.7**	53.4	38.7	**58.7**	**77.3**	**68.9**	22.0
		se	1.1	4.8	4.0	5.5	4.3	3.7	2.2
	recognizer B	mean	**90.0**	68.7	58.7	**64.0**	**50.8**	**70.0**	33.3
		se	2.0	2.0	3.3	2.8	3.1	3.7	3.1
C_i in Cvc-words	recognizer A	mean	86.8	56.4	64.7	45.9	55.4	56.1	30.6
		se	2.8	2.5	2.4	6.3	6.1	4.1	2.6
	recognizer B	mean	72.7	55.0	38.0	67.1	41.9	35.3	11.2
		se	8.4	4.0	3.0	2.4	2.6	2.2	1.6

For a number of conditions the corresponding confusion matrices were used to calculate the corresponding two dimensional representation. This procedure was described earlier (Steeneken[1992]) and is based on a multidimensional scaling described by Caroll and Chang[1970]). In Fig. 1 (left panel) the relative position for the 15 vowels is given. This mean representation is based on six conditions with and without oxygen mask, two noise conditions, and two recognizers (conditions are printed **bold** in Table I). The figure indicates for instance a close relation between /e/, /y/, /I/ (IPA notation). These phonemes are also perceptually very close.
The fit of the confusion matrices for each individual condition is given In Fig. 1 (right panel). The labels correspond with the type of recognizer (**A**, **B**) , oxygen mask (+/-), and noise condition (**no**, **direct**, or **additive**).
This graph represents the relative weight for each condition on each dimension. An equal weight on both dimension (such as for An- and Bn-) means that for these particular conditions the representation in panel A (average of all conditions) corresponds quite well.
A low weight on one of the two dimension means that the optimal representation for that condition on that particular dimension is different. This occurs with the conditions An+, Aa+, and Bn+, Ba+.

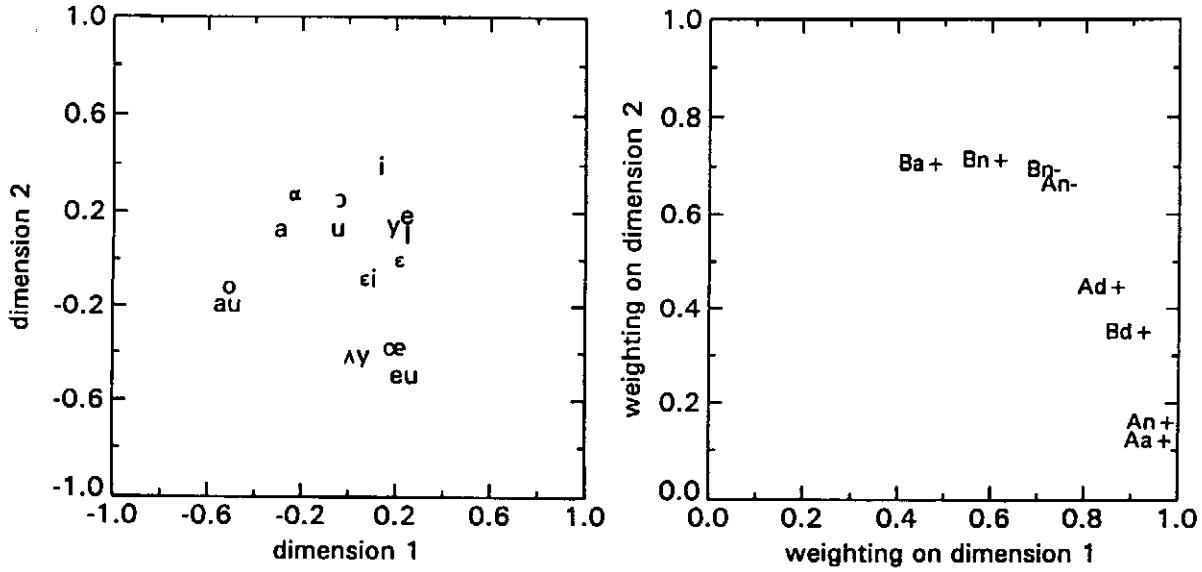


Fig. 1 Stimulus (left) for 15 Vowels (IPA notation) based on six conditions. The right panel represents the fit of each condition in the stimulus space of the left panel. Dimension 1 accounts for 62% of the variance, dimension 2 accounts for 29% of the variance. The conditions A, B refer to two recognizers, a, d to added and direct noise condition and + vs. - to the use of the oxygen mask.

4. Large-vocabulary Recognition

Large vocabulary recognition is based on collecting statistics from many hours of acoustic training material (speech signals) and many millions of words from a text file. This results in a set of acoustic phone models, a lexicon, and a language model.

The majority of the present large vocabulary recognizers are laboratory models, still under development. Only a few systems (speaker dependent or adaptive) are commercially available.

For the assessment of these systems reproducible test are scheduled for which many parameters must be defined in advance. In the SQALE project two aspects of comparison of systems are included: (1) compare different systems with the same language, and (2) compare the same system with different languages. The first aspect serves as a bench mark for comparison of the relative performance of the systems which is required if not all the languages are tested with all systems.

As the effect of some of the parameters can not be controlled for the different languages (e.g., same native speakers, selected utterance) and estimation of the effect on the final score by these parameters must be obtained. Therefore, two additional aspects will be included: (3) the effect of speaker variation, (4) the effect of the test material selected. The aspects 2, 3 are not studied before and are research issues raised by the SQALE project. The project results should allow a comparison of different recognition algorithms and the relative difficulties in speech recognition across languages.

Additional to the comparison of systems, human recognition will be included in the experiments and used as a bench mark.

For the comparison between languages, comparable training material for the languages is required. This training material can be divided in three parts:

- Acoustic speech data. All databases have 8-10 hours of speech.
- Lexicon. A set of roughly 50 phones will be used. The vocabulary size will be typically 20,000 words. The coverage of the language by the lexicon is estimated to be 95-97%.
- Language Model. The models are usually based on trigrams trained on backed-off trigram counts of newspaper texts having a total length of typically 37M words.

For the SQALE project three languages,

English, French and German are included and additionally American English as reference. The corresponding corpora used are respectively:
- the American English database based on the Wall Street Journal (WSJ0),
- the British English version of WSJ0,
- the French "BREF-80",
- the German "Phondat".

The British English database: WSJCAM0
In analogy to the efforts of the American WSJ project, researchers of the Cambridge University Engineering Department (Fransen et al.[1994]) assembled a large speech database called WSJCAM0, the Wall Street Journal recorded at CAMbridge university. All utterances were spoken by native British English speakers. Some 90 speakers uttered 90 sentences each.

The French database: BREF-80
A large corpus of speech recordings (120 speakers, over 100 hours of material) was assembled at LIMSI (Gauvain et al.[1990], Lamel et al.[1991]) to provide continuous spoken French to researchers on acoustic-phonetics and speech recognition. The spoken sentences were excerpts of the French newspaper "Le Monde", thus covering over 20k words in a wide range of phonetic environments.
The BREF-80 sub-corpus contains over 10 hours of speech.

The German database: Phondat
The training database for the German language consists of a set of phonetically balanced sentences and data recorded for the railway information system. The number of speakers is 194. Recordings were made in Bochum, Munchen, Bonn and Kiel. For the purpose of SQALE, only the recordings of 100 speakers will be used, this amounts to 10 hours of speech. Additionally, data will be used from German Railway Information System recordings, consisting of train inquiries, adding an extra half hour of speech.

After training the large vocabulary recognition system with the training material and supplying the language model a dry run is performed, preceding the real assessment test. This dry run includes the same procedures as the test runs but are used as a final check to make sure that the preceding steps work well. For a dry run speech material is used from 20 speakers, each speaker uttering 10 sentences. Because of the assessment of speaker-independent recognition systems it is essential that the speakers in the test have never been used before, 40 distinct persons per language (20 for the dry run and 20 for the actual test) are required.
The dry run and test material are selected according well defined criteria:
- A fraction of out-of-vocabulary words is allowed. Words not in the recognizer's vocabulary are not just impossible to be recognized, they also cause additional errors because they confuse the language model as well.
- Sentence perplexity. This is a measure how well the words of a sentence matches the language model. The higher the perplexity, the more difficult the recognition task. The sentence perplexity should (more-or-less) match that of the language model itself.
- Diversity. This is an effort to cover as many possible words with a limited number of test sentences.
- Sentence length. Sentences can be selected with lengths within a certain range. The length of the sentence has consequences for the sentence-score and beam-search algorithm.
- Phonetic balance. The distribution of the phonemes used in the test material must match with the distribution representative for the language.

Of these criteria the percentage of out-of-vocabulary words, the perplexity, and the sentence length are the most important ones. An acceptable fraction of out-of-vocabulary words is 1-2%. Therefore, a set of sentences is normally selected which contains approximately 1.5% OOV-words. The sentence length is normally taken in the range of 10-30 words, with an average around 20. Thus, some 4000 words per language will be used for the test material.

Testing
After completion of the dry run, an evaluation test will take place. In a comparative test with various participating laboratories a coordinator will supply the test-data on a convenient medium such as a CD-rom, and

within a certain time-frame (e.g., three weeks). The test laboratories have to return the recognizer output data.

The test-material may also consist of 20 speakers, each uttering 10 sentences. Half of the speakers should be female. For the SQALE project an extension of the test is planned. Additionally to the 200 test sentences, 100 additional sentences will be supplied for statistical analyses. Therefore, some of these extra utterances may contain the same sentence produced by the same speaker, but in that case the recordings will be different. For the benefit of the statistics it is required that the recognizer does not use information about replicas of sentences. Although these 100 sentences are in addition to the original experimental design, all participating laboratories volunteered to perform a test on these data as well.

Scoring of the Results

In general recognition errors are counted on a word level. There are many parameters that vary over a total test, for the SQALE project these include language (4), recognition system (4), gender (2), speaker (4 times 20).

The description given above is focused on system and language. It is obvious that other parameter such as telephone speech, spontaneous speech, the effect of back ground noise, etc., should be included as well (Kubala[1994]). It is expected that the effect of these parameters on the recognition rate will be evaluated in the future.

5. Conclusion

Assessment of speech recognition system can be focused either on a application oriented or on a laboratory oriented approach. The first is representative but is expensive and the conditions are often uncontrolled, the laboratory oriented methods offer reproducable conditions and the possibility to study individual parameters. For development normally the latter method is choosen.

6. References

Aubert, X., Ch. Dugast, R. Knesser, V. Steinbiss, S. Besling & H. Ney [1994]. The Philips Large Vocabulary System. *Proceedings of the ARPA Spoken Language Program Workshop*, Morgan Kaufmann Publishers.

Carroll, J.D. & Chang, J.J. [1970]. Analysis of individual differences in multidimensional scaling via an N-way generalization of the "Eckhart-Young" composition. *Psychometrika* 35, 283-319.

Chollet, G.F. & C. Gagnoulet [1981]. On the Evaluation of recognizers and data bases using a reference system. *IEEE Proc. ICASSP*, Atlanta.

Fransen, J., D. Pye, P. Robinson, P. Woodland & S. Young [1994]. WSJCAM0 Corpus and Recording Description. *CD-rom documentation for the SQALE project.*

Gauvain, J.L., L.F. Lamel & M. Eskenazi [1990]. Design Considerations and Text Selection for BREF, a large French read-speech corpus. *Proc. ICSLP-90*, Kobe, Japan.

Hieronymus, J.L. & W.J. Majurski [1985]. A reference speech recognition algorithm for benchmarking and speech data base analysis. *IEEE Proc. ICASSP.*

Hunt, M.J. [1990]. Figures of merit for assessing connected word recognizers. *Speech Communication* 9, 329-336.

Kubala, F. [1994]. The Hub and Spoke Paradigm for CSR Evaluation. *Proc. of the ARPA Human Language Technology Workshop, Morgan Kaufmann Publishers* [Mar. 1994].

Lamel, L.F. & J.L. Gauvain [1992]. Large Vocabulary Speech Recognition at LIMSI. *Final review Darpa Speech Program*, Stanford, CA.

Lamel, L.F., J.L. Gauvain & M. Eskenazi [1991]. BREF a Large Vocabulary Spoken Corpus for French. *Proc. Eurospeech '91*, Genoa, Italy.

McNemar [1947]. Note on the sampling error of difference between correlated proportions or percentages. *Psychometrika* 12, 153-157.

Moore, R.K. [1977]. Evaluating Speech Recognizers. *IEEE Trans. ASSP* 25, 2, 178-183.

Moore, R.K. [1992]. Esprit project 2589 Multi-Lingual Speech Input-Output Assessment, Methodology and Standardization. Final Report (Revised Edition June[1992]). Ref: SAM-UCL-G004.

Néel, F. [1994]. NATO RSG.10 Overview on Speech Technology products. Retrieval (ftp://m29.limsi.fr/pub/cpprod/prod92a).

Ney, H., D. Mergel, A. Noll & A. Paeseler [1992]. Data Driven Organization for Continuous Speech Recognition. *IEEE Tran. Signal Proc.* P-40, 272-281.

Pallet, D.S., J.G. Fiscus, W.M. Fisher, J.S. Garofolo, B.A. Lund & M.A. Pryzbocki [1994]. *1993 Benchmark Test for the ARPA Spoken Language Program.*

Peckham, J. & T.J. Thomas [1990]. Recogniser sensitivity analysis: A method for assessing the performance of speech recognisers. *Speech Communication* 9, 317-328.

Simpson, C.A. & J.C. Ruth [1987a]. The Phonetic Discrimination Test for Speech Recognizers: part I. *Speech Technology* March/April.

Simpson, C.A. & J.C. Ruth [1987b]. The Phonetic Discrimination Test for Speech Recognizers: part II. *Speech Technology* Oct/Nov.

Steeneken, H.J.M. & T. Houtgast [1980]. A physical method for measuring speech transmission quality. *J. Acoust. Soc. Am.* 67, 318-326.

Steeneken, H.J.M. [1987]. Diagnostic information from subjective and objective intelligibility tests. *IEEE Proc. ICASSP*, Dallas.

Steeneken, H.J.M. & J.G. van Velden [1989a]. Objective and diagnostic assessment of (isolated) word recognizers. *IEEE Proc. ICASSP*, Glasgow.

Steeneken, H.J.M. & J.G. van Velden [1989b]. RAMOS - Recognizer assessment by means of manipulation of speech. *European Speech Conference ESCA*, Paris.

Steeneken, H.J.M. & T. Houtgast [1991]. On the mutual dependency of octave-band-specific contributions of speech intelligibility. *European Speech Conference ESCA*, Genova.

Steeneken, H.J.M. & A. Varga [1993]. Assessment for Automatic Speech Recognition: I. Comparison of Assessment Methods. *Speech Communication* 12, 241-216.

Varga, A. & H.J.M. Steeneken [1993]. Assessment for Automatic Speech Recognition: II. Noisex-92: A Database and an Exeriment to Study the Effect of additive noise on speech recognition systems. *Speech Communication* 12, 247-251.

van de Vegte, J.M.E. & M.M. Taylor [1990]. Testing the effective vocabulary capacity method of evaluating speech recognizers. *Speech Communication* 9, 337-347.

van Velden, J.G. & H.J.M. Steeneken [1989]. RAMOS I: Recognizer assessment by manipulation of speech. *Proc. ESCA workshop on Speech Input/Output Assessment and Speech Databases*, Noordwijkerhout.

Woodland P.C. & S.J. Young [1993]. The HTK Continuous Speech Recognizer. *Proc. Eurospeech* '93, Berlin, 2207-2219.

THE ROLE OF PROSODY IN HUMAN SPEECH RECOGNITION

James M. McQueen
Max Planck Institute for Psycholinguistics,
Wundtlaan 1, 6525 XD Nijmegen, The Netherlands
james@mpi.nl

ABSTRACT

Prosodic information is used by human listeners in spoken language comprehension. Different types of prosodic information, however, are used at different stages in the comprehension process. It is possible to distinguish between information that is used prelexically, in lexical access; information that is used at the lexical level, in word recognition; and information that is used postlexically, in syntactic and semantic analysis. Several aspects of prosodic structure will be discussed. For example, lexical stress information is not used prelexically, probably because its use would delay lexical access. Rhythmic information, on the other hand, is extracted prelexically and is used to guide the segmentation of continuous speech into words. This segmentation procedure has been successfully implemented in the Shortlist model of human spoken word recognition.

INTRODUCTION

Prosodic information can be defined as the variation over an utterance of three acoustic parameters: fundamental frequency, duration, and intensity. It specifies suprasegmental structures in the utterance and therefore must have an important role to play in spoken language understanding. In order to define this role, however, it is necessary to establish the stages of processing at which different sources of prosodic information are brought to bear. At least three distinct levels of processing are usually identified in models of human speech understanding: prelexical processing (the processing of information prior to and for lexical access); lexical processing (the selection and recognition of individual words); and postlexical processing (the syntactic and semantic integration of words into a higher-level interpretation of an utterance).

Prosodic information has a role to play at each of these broadly-defined levels. In this paper, I will focus on the prelexical use of prosodic information. In the first section, I will discuss how prosodic information can be used predictively, so that the listener can focus on important parts of the speech signal. In the following section, I will ask whether lexical stress information is used prelexically, to constrain lexical access. In the third section, evidence will be presented which shows that rhythmic information is used in the segmentation of continuous speech into individual words. In the final section, I will describe how a segmentation procedure using rhythmic information can be implemented in a new, connectionist model of continuous spoken word recognition, the Shortlist model (Norris, 1994).

ATTENTIONAL FOCUS

In attempting to delineate the level or levels of processing at which a particular type of prosodic information is used, it is important to consider not only the obvious domain of that information, but also how it could be used at other stages of processing. For example, consider sentence accent. Obviously, information specifying that a particular word bears sentence accent will be crucial to the postlexical processes which extract a semantic interpretation of that sentence. But this information could also be of value to other processes. Since accented words are more important to the overall understanding of a sentence, perhaps prelexical and lexical processes also make use of sentence accent information. It appears that this is indeed the case: listeners use the prosodic contour of a sentence predictively, so that they can focus attention on words in to-be-accented positions.

Shields, McHugh and Martin (1974) found that listeners were faster to detect a target phoneme at the onset of a nonsense word when the first syllable was stressed (e.g. *BENkik*) than when the second syllable was stressed (*benKIK*). But this was only when the target-bearing words were in predictable locations in sentence contexts, not when they were presented in a sequence of nonsense words. Shields et al. (1974) therefore argued that the effect was not due to acoustic differences between stressed and unstressed syllables, rather that it was due to listeners' predictive use of the prosodic contour of the sentences. The difference between sentential and nonsense contexts could however be due instead to the unnaturalness of the strings of nonsense words.

A more convincing demonstration of listeners' ability to use prosodic information predictively was provided by Cutler (1976), who used identical acoustic tokens of words in two different sentence contexts. In one context, the word appeared in a position where the preceding prosodic contour predicted a sentence accent; in the other, the word was in an unaccented position. Listeners were faster to detect the initial phonemes of these words in the accented than in the unaccented positions. Subjects can predict sentence accent location, and still be faster to detect phonemes in these locations than in unaccented locations, when fundamental frequency variation in the sentences is removed (Cutler & Darwin, 1981).

Listeners use the prosodic contour of a sentence to predict the location of accented words, so that they can then focus attention on information at these locations. It thus appears that this type of prosodic information can be used at several different levels of processing. Attentional focus on an accented word benefits the prelexical processing that allows a target phoneme to be detected; attentional focus is also likely to speed the recognition of the accented word; and the accent information is of course essential for the extraction of a semantic interpretation.

Other types of prosodic information can also be used predictively. Pitt and Samuel (1990) tested subjects' ability to detect target phonemes in words from minimal stress pairs (e.g. *PERmit* and *perMIT*). They presented listeners with lists of disyllabic words which all had stress on the first syllable, or all had stress on the second syllable, followed by acoustically neutral versions of the stress-pair words (e.g. PERMIT, made by splicing together the two stressed syllables). Listeners were faster to detect a target phoneme in these words when the phoneme appeared in the syllable which could from the preceding list be predicted to be stressed than when it appeared in a syllable which was predicted to be unstressed. Listeners can thus take advantage of lexical stress information, in the same way as with sentence contours, to focus attention on particular aspects of the signal. Prosodic information, via attentional focus, can therefore be used to benefit prelexical processing.

LEXICAL STRESS

Is lexical stress information used at the prelexical level to constrain the process of lexical access? Words which have been mis-stressed are harder to recognise than correctly stressed words (Bond & Small, 1983). Cutler and Clifton (1984) compared mis-stressing of words which have schwa in their unstressed syllable (e.g. *WISdom* as *wisDOM*, and *deCEIT* as *DEceit*) with mis-stressing of words which have two full vowels (e.g. *NUTmeg* as *nutMEG*, and *tyPHOON* as *TYphoon*). Mis-stressed words of the former type were harder to recognise, but this may have been due to a vowel quality mismatch (schwa swapping with a full vowel) rather than a lexical stress mismatch *per se*. In the latter category, however, mis-stressing also had an effect, such that, for example, *nutMEG* was harder to recognise than *NUTmeg*. This appears to be a genuine effect of lexical stress mismatch. Note however that Cutler and Clifton (1984) found that mis-stressings like *TYphoon* were not significantly harder to recognise than the correctly stressed forms. This may be because stress shift rules can require stress to shift to an earlier full syllable (for example, "the UNknown SOLdier", even though the citation form is *unKNOWN*). Pronunciations like *TYphoon* and *UNknown* may therefore be stored in the lexicon, such that supposed mis-stressings of this type do not create a mismatch.

These results might suggest that lexical stress information is used prelexically, to select the access of particular lexical entries. But these results are also consistent with the claim that lexical stress information is *not* used prelexically – they could simply be due to a failure to find a complete match between the mis-stressed input and the lexical entry which is nonetheless activated by that input. A means of establishing whether lexical stress information is used prelexically is provided by minimal stress pair words like *forbear*. The fact that such pairs are rare in stress languages like English and Dutch itself suggests that lexical stress information may not be very useful prelexically. Cutler (1986) asked subjects to listen to sentences with, for example, *FORbear* or *forBEAR* in them. At the offset of these words, subjects saw items and were required to decide if they were real words. The subjects were faster to perform this decision to semantic relatives of both members of the stress pair after hearing either form (both *FORbear* and *forBEAR* primed decisions to both *ancestor* and *tolerate*). Both meanings of *forbear* appear to be accessed irrespective of the stress pattern of the input, suggesting that lexical stress information is not used prelexically to constrain the lexical access process.

There is a very good reason why lexical stress information is not used to constrain lexical access. By definition, the stress pattern of a word can only be determined by comparison of syllables over a word. It is impossible to tell whether a strong first syllable bears primary or secondary stress until the following syllables are available for comparison. In other

words, lexical stress information cannot be used to constrain lexical access, unless lexical access is delayed until the end of the word. This would be a very inefficient procedure. In any case, a large body of research indicates that lexical access is rapid and immediate, and many models of spoken word recognition thus assume that lexical access is not delayed until word offsets (e.g. Cohort, Marslen-Wilson, 1987; TRACE, McClelland & Elman, 1986; and Shortlist, Norris, 1994).

LEXICAL SEGMENTATION

In contrast to lexical stress information, other prosodic information is used prelexically, for speech segmentation. In normal continuous speech, the boundaries between words are not reliably marked, and yet listeners are able to divide this stream of speech sounds into a sequence of individual words. Prosody plays a role in solving this segmentation problem. Specifically, lexical segmentation appears to be based on language rhythm.

In English and Dutch, for example, rhythmic structure is stress-based, and can be characterised in terms of feet (where a foot is a strong syllable plus optional following weak syllables). Metrical structure thus depends on the alternation of strong and weak syllables. Various lines of research suggest that strong syllables have an important role to play in lexical segmentation. One line of enquiry is computational. Cutler and Carter (1987) have shown that the majority of English content words begin with strong syllables, and furthermore that most strong syllables in conversational English are word-initial. Vroomen and de Gelder (1995) have shown that most Dutch content words are also strong-initial. A segmentation procedure which assumed that strong syllables are likely to be word onsets would therefore work well for both of these languages. A second line of enquiry has looked at misperceptions of word boundaries. When listeners make the mistake of inserting a word boundary when there is not one in the signal, they do so more often before strong than before weak syllables; when listeners delete a boundary that was in the signal, they do so more often before weak than before strong syllables (Cutler & Butterfield, 1992; van Zon & de Gelder, 1993).

Another experimental line of enquiry has used the word-spotting task (Cutler & Norris, 1988). Listeners hear a list of nonsense words, some of which have a real word embedded within them. The task is to detect these real words. Cutler and Norris (1988) found that it was harder to detect a word like *mint* in a bisyllabic item with two strong syllables (/mɪntɛɪv/) than in a bisyllabic item with a strong-weak pattern (/mɪntəv/). If lexical segmentation is triggered by strong syllables, the strong-strong string will be segmented at the /t/, making it hard to detect *mint*, which will have to be assembled over a segmentation position; no such segmentation should take place in the strong-weak string, so detection of *mint* should be easier.

Not all languages have stress-based rhythm: French, for example, has a syllable-based rhythm, and Japanese rhythm is mora-based. Cross-linguistic investigations have shown that in these languages, segmentation is also based on rhythm. Thus, segmentation in French appears to be syllable-based (Mehler, Dommergues, Frauenfelder & Segui, 1981), while that in Japanese appears to be mora-based (Otake, Hatano, Cutler & Mehler, 1993). Segmentation based on linguistic rhythm appears to be a language universal phenomenon, with different, language-specific instantiations.

Models of spoken word recognition, including TRACE (McClelland & Elman, 1986) and Shortlist (Norris, 1994), have an alternative mechanism for lexical segmentation, one based on competition between lexical hypotheses. In Shortlist, for example, the words best matching the available speech input are wired into a small interactive activation network. This "shortlist" of words then competes for recognition, via inhibitory connections between word nodes. The parse of words which best matches the input will win in the competition process and these words will be recognised. Given *mad captive* as input, for example, words like *mad*, *madcap*, *cap* and *captive* would enter the shortlist as the information arrived. To begin with, *madcap* would have a higher degree of activation than *mad* (since there is more bottom-up evidence in favour of the longer word), but as *captive* won out over *cap*, it would inhibit *madcap*, allowing *mad* to be recognised, and the correct parse of the input to be obtained. Segmentation can thus be achieved by a competition process which allows words straddling different parts of the input string to compete with each other.

An important question is therefore whether competition is a sufficient mechanism to account for lexical segmentation, or whether both competition and prosodically-guided segmentation are required. In a series of experiments, we have examined this issue (McQueen, Norris & Cutler, 1994; Norris, McQueen & Cutler, 1995). In McQueen et al. (1994), we tested for effects of both competition and metrical segmentation in the same word-spotting experiment. Subjects were required to detect real words (such as *mess*) embedded in bisyllabic nonsense strings which

were either the beginnings of longer real words (such as /dəmɛs/, the onset of *domestic*) or not (such as /nəmɛs/, which cannot be continued to form any real words). Subjects found it harder to detect the target words in the strings which could be continued to form real words than in the strings which could not. This is a clear demonstration of the lexical competition effect predicted by the Shortlist model: in the word onset case, *domestic*, for example, competes for recognition with *mess*, making it harder to detect the target word. No such strong competition is present in the nonword onset case.

Shortlist also predicts a competition effect for items like *sack* embedded in word onsets (/sækrəf/, the beginning of *sacrifice*) and nonword onsets (/sækrək/, which cannot be continued to form a real word). This competition effect was also found, although it was smaller than that found for the items like /dəmɛs/. This difference in size of the competition effects is also predicted by Shortlist: the effect should be greater when the target starts later than the competitor (*mess* versus *domestic*) than when the words are aligned (*sack* versus *sacrifice*). This is because in the first case the competitor can be a more effective inhibitor, since its activation will have built up before the target becomes activated.

This experiment also tested for an effect of prosodically-guided segmentation. All of the items with a target in the second syllable (e.g. /dəmɛs/) had a weak-strong stress pattern, while all the items with a target in the first syllable (e.g. /sækrəf/) had a strong-weak stress pattern. If strong syllables are taken to be likely word onsets, segmentation should take place at the onset of any strong syllable. For the weak-strong strings, this segmentation should occur, and furthermore, it should occur at the onset of the target word, making the target easier to detect. For the strong-weak strings, there should be no segmentation. Overall, aside from any competition effects, listeners indeed found target detection to be easier in weak-strong strings (e.g. *mess* in /nəmɛs/) than in strong-weak strings (e.g. *mess* in /mɛstəm/).

The results of McQueen et al. (1994) show that both prosodically-guided segmentation and lexical competition are required to account for continuous speech recognition. In Norris et al. (1995), we asked how these two mechanisms interact. We again used the word-spotting task, but used materials similar to those from Cutler and Norris (1988), where the target word was followed either by another strong syllable or by a weak syllable (like /mɪntelv/ and /mɪntəv/).

The basic prediction from prosodically-guided segmentation is that target detection should be harder in strong-strong than in strong-weak strings. In Norris et al. (1985) the number of words beginning at the onset of the second syllable was systematically manipulated. For example, many words begin with /k/, including *cup, custard, company,* and so on, and many words begin with /kə/, including *collide, cathedral, cadet,* and so on. In comparison, far fewer words begin with /taʊ/ (only *town, tout, towel,* and *tousle*) and /tə/ (such as *tomorrow, toboggan,* and *tonight*). If lexical competition modulates prosodically-guided segmentation, the segmentation effect (the advantage for targets in strong-weak versus strong-strong strings) should be larger for targets like *mask* in strings like /maskʌk/ and /maskək/ (where there are many words competing from the /k/) than for targets like *mint* in strings like /mɪntaʊp/ and /mɪntəp/ (where there are few words competing from the /t/). This is exactly what was found.

In Dutch, using cross-modal identity priming, Vroomen and de Gelder (1995) have also shown effects of the number of competitors in the second syllable of bisyllabic strings. Listeners were faster in visual lexical decision to *melk* (milk) after hearing *melkem* (a strong-weak string with no competitors beginning from the /k/) than after hearing a control word like *lastem.* They were also faster on visual *melk* after hearing *melkem* than after hearing either *melkeum* (a strong-strong string with few second syllable competitors) or *melkaam* (a strong-strong string with many second syllable competitors). In addition, subjects were faster in lexical decision to *melk* after *melkeum* (few competitors) than after *melkaam* (many competitors). Vroomen and de Gelder (1995) argue that lexical competition is required to account for the recognition of Dutch words.

SHORTLIST IMPLEMENTATION

These results (McQueen et al., 1994; Norris et al., 1995; Vroomen & de Gelder, 1995) show that prosodically-guided segmentation and lexical competition have a role to play in continuous speech recognition. In Norris et al. (1995), we took these findings one step further, by implementing a metrical segmentation procedure in the Shortlist model.

Shortlist is a two-stage model. The first stage can in principle be instantiated as a recurrent net (Norris, 1990), but this becomes very computationally expensive with a large lexicon. Since it is important that the model simulates real human data, it must be run with a lexicon that is a realistic approximation of the adult human lexicon. To achieve this, the recurrent net is itself simulated by an exhaustive search of a machine-readable dictionary. We can thus run simulations using lexicons of over 25,000 words.

The search procedure consists of a computation of the degree of match between the acoustic-phonetic information in the input and the words in the lexicon. Matching information increases the evidence in favour of a word, and mismatching information decreases it. An activation score is computed for words which are roughly consistent with the input, and a small set of these words, those which best match the input, the "shortlist", are entered into the second stage of processing. This consists of an interactive activation network, where words compete, via inhibitory connections, until the model settles on a best parse of the input.

There are two obvious ways in which prosodically-guided segmentation could be implemented in Shortlist, corresponding to the match and mismatch computations that are based on segmental information. Words with a metrical match could receive a boost in activation level, or those with a metrical mismatch could receive a penalty in activation level. Specifically, words aligned with a strong syllable in the input could have their activation boosted, or words misaligned with a strong syllable in the input could have their activation penalized. In either of these two possibilities, metrical information available prelexically in the signal would be used to compute the initial activation score of candidate words. Those matching the input prosodically would have a higher activation level, and would therefore be more likely to win out in the competition process and be recognised.

Are both the boost and the penalty required, or can the available data be accurately simulated with only one of these two procedures? Over a large number of simulations of both the McQueen et al. (1994) and the Norris et al. (1995) data, we found that both the boost and the penalty procedures were required (Norris et al., 1995). Interestingly, the boost and penalty mirror the original formulation of prosodically-guided segmentation (Cutler & Norris, 1988). In that proposal, it was suggested that strong syllables cause the input to be segmented, and that lexical access is then initiated at these segmentation positions. The penalty instantiates the segmentation component: a strong syllable like /teɪv/ in /mɪnteɪv/ causes *mint* to be penalized, since *mint* has no strong syllable beginning at the /t/, as indicated in the input; it is as if the input is segmented before the /t/. The boost instantiates the lexical access component: strong syllables like *mess* in /dəmɛs/ are more likely to be word onsets, so they are given an activation boost during lexical access; although in Shortlist access can be initiated at every syllable, words at strong syllables are at an advantage.

The combined boost and penalty procedures allowed the data from McQueen et al. (1994) to be simulated accurately. The mean activation level of a target words was consistently higher in weak-strong strings (e.g. *mess* in /nəmɛs/) than in strong-weak strings (e.g. *mess* in /mɛstəm/), in line with the result that target detection was easier in weak-strong strings. Prior to the instantiation of the segmentation procedures, Shortlist was unable to simulate this result. It had, however, been able to simulate all of the competition results. It was still able to do so after the segmentation procedures were added. Thus, in line with the human data, activation of words like *mess* was higher, at the offset of the target, in /nəmɛs/ than in /dəmɛs/ (due to competition from *domestic*), and activation of words like *sack* was higher, at the end of the whole string, in /sækrək/ than in /sækrəf/ (due to competition from *sacrifice*).

The prosodically-enriched Shortlist model was also able to simulate the Norris et al. (1995) data. In the second syllables of strings like /maskʌk/, /maskək/, /mɪntaup/ and /mɪntəp/, the activation levels of e.g. *mask* and *mint* were higher in the strong-weak strings than in the strong-strong strings. This result simulates the basic advantage found in the human data for the targets in strong-weak strings. Furthermore, the model simulated the modulation of this stress-pattern effect by number of second syllable competitors. The difference in activation level for words like *mask* in /maskʌk/ and /maskək/, where there were many competitors, was greater than the difference in activation level for words like *mint* in /mɪntaup/ and /mɪntəp/, where there were fewer competitors.

These simulation results show that prosodically-guided segmentation can be implemented in the Shortlist model. All the simulations were performed with large lexicons, which allowed the actual words from the experiments to be used in the simulations. In other words, the simulation results are based on the mean activations of sets of target words, so they are not likely to be due to idiosyncrasies of particular test words. More importantly, Shortlist thus provides a much more plausible simulation of human performance than models with very limited lexicons.

CONCLUSIONS

Prosodic information serves several roles in human speech recognition. I have argued that preceding prosodic contour or rhythm can be used predictively, and hence that prosodic information can be used to focus attention on important parts of the signal, which can thus be processed more easily. In addition, I have argued that lexical stress information

is not used prelexically, to constrain the process of lexical access: this information only appears to play a role at the lexical level of processing.

I have also described how prosodic information is used in the segmentation of speech. Prosodically-based segmentation (using strong syllables in a stress-timed language like English) complements segmentation based on lexical competition, as instantiated in the Shortlist model. Furthermore, the prosodically-guided procedure has been built into the activation/competition mechanism of Shortlist. Although the final segmentation of the input is based on a competition process operating at a lexical level of processing, it is important to emphasize that the instantiation of the prosodic segmentation procedure depends on the availability of prosodic information (whether a syllable is strong or weak) at a prelexical level of processing (during the access and selection of the shortlist members). The prosodically-enriched Shortlist model provides an accurate account of the performance of human listeners during continuous speech recognition.

Acknowledgements

This work was done in collaboration with Anne Cutler and Dennis Norris. Part of it was supported by the Joint Councils Initiative in Cognitive Science and HCI, grant no. E304/148, and was carried out when James McQueen and Anne Cutler were at the MRC Applied Psychology Unit, Cambridge, U.K.

REFERENCES

Bond, Z.S. and Small, L.H. (1983). Voicing, vowel and stress mispronunciations in continuous speech. *Perception & Psychophysics*, **34**, 470–474.

Cutler, A. (1976). Phoneme monitoring reaction time as a function of preceding intonation contour. *Perception & Psychophysics*, **20**, 55–60.

Cutler, A. (1986). *Forbear* is a homophone: Lexical prosody does not constrain lexical access. *Language and Speech*, **29**, 201–220.

Cutler, A. and Butterfield, S. (1992). Rhythmic cues to speech segmentation: Evidence from juncture misperception. *Journal of Memory and Language*, **31**, 218–236.

Cutler, A. & Carter, D.M. (1987). The predominance of strong initial syllables in the English vocabulary. *Computer Speech and Language*, **2**, 133–142.

Cutler, A. and Clifton, C.E. (1984). The use of prosodic information in word recognition. In H. Bouma and D.G. Bouwhuis (Eds.), *Attention and performance X: Control of language processes* (pp. 183–196). Hillsdale, NJ: Erlbaum.

Cutler, A. and Darwin, C.J. (1981). Phoneme-monitoring reaction time and preceding prosody: Effects of stop closure duration and of fundamental frequency. *Perception & Psychophysics*, **29**, 217–224.

Cutler, A. and Norris, D. (1988). The role of strong syllables in segmentation for lexical access. *Journal of Experimental Psychology: Human Perception and Performance*, **14**, 113–121.

McClelland, J.L. and Elman, J.L. (1986). The TRACE model of speech perception. *Cognitive Psychology*, **18**, 1–86.

McQueen, J.M., Norris, D. and Cutler, A. (1994). Competition in spoken word recognition: Spotting words in other words. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **20**, 621–638.

Marslen-Wilson, W.D. (1987). Functional parallelism in spoken word-recognition. *Cognition*, **25**, 71–102.

Mehler, J., Dommergues, J.-Y., Frauenfelder, U.H. and Segui, J. (1981). The syllable's role in speech segmentation. *Journal of Verbal Learning and Verbal Behavior*, **20**, 298–305.

Norris, D.G. (1990). A dynamic-net model of human speech recognition. In G.T.M. Altmann (Ed.), *Cognitive Models of Speech Processing: Psycholinguistic and Computational Perspectives* (pp. 87–104). Cambridge, MA: MIT Press.

Norris, D. (1994). Shortlist: A connectionist model of continuous speech recognition. *Cognition*, **52**, 189–234.

Norris, D., McQueen, J.M. and Cutler, A. (1995). Competition and segmentation in spoken word recognition. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **21**, in press.

Otake, T., Hatano, G., Cutler, A. and Mehler, J. (1993). Mora or syllable? Speech segmentation in Japanese. *Journal of Memory and Language*, **32**, 258–278.

Pitt, M.A. and Samuel, A.G. (1990). The use of rhythm in attending to speech. *Journal of Experimental Psychology: Human Perception and Performance*, **16**, 564–573.

Shields, J.L., McHugh, A. and Martin, J.G. (1974). Reaction time to phoneme targets as a function of rhythmic cues in continuous speech. *Journal of Experimental Psychology*, **102**, 250–255.

Vroomen, J. and de Gelder, B. (1995). Metrical segmentation and lexical inhibition in spoken word recognition. *Journal of Experimental Psychology: Human Perception and Performance*, **21**, in press.

van Zon, M. and de Gelder, B. (1993). Perception of word boundaries by Dutch listeners. *Proceedings of the 3rd European Conference on Speech Communication and Technology*, Berlin, 689–692.

The potential role of prosody in automatic speech recognition.

Louis F.M. ten Bosch

Institute for Perception Research/IPO
P.O. Box 513, 5600 MB Eindhoven
e-mail: tenbosch@prl.philips.nl

Abstract

In this paper, we study the potential role of prosody in the context of automatic speech recognition. It will be argued that prosodic information may play an essential role in the recognition performance by reducing the search space of acoustic hypotheses, but also on the interface between the acoustic analysis and the linguistic model part of the recognition system.

1 Introduction

Presently, systems performing automatic speech recognition show the best recognition scores while they are based on purely statistical methods to derive essential cues from the speech signal. Systems based on the nowadays widely used HMM-approach perform better than systems based on 'acoustic knowledge', but the recognition rates (word-recognition rates, or word-spotting rates, etc.) crucially depend on many factors: the required vocabulary, speaker-dependent, speaker-adaptive or speaker-independent, the recording environment, speaking style, speaking rate, isolated words versus connected speech, read-aloud versus spontaneous versus elicited, etc.

In this paper, we study the role that prosodic information may play in automatic speech recognition (ASR). In fact, we then ask for the usefulness of prosodic information within the context of probabilistic classification methods. In the next section, we firstly consider three major possible contributions of prosody to ASR: (probabilistic) stress prediction, the (probabilistic) detection of prosodic boundaries, and the desambiguation of ambiguous sentences. The first two issues deal with the (probabilistic) reduction of the lexical or grammatical search space within ASR. The third issue deals with pragmatic questions.

In the second section, we go into detail with respect to the automatic classification of (Dutch) pitch movements. We briefly discuss the role of several statistical features of the training data that have shown to play a role in the actual classification results.

2 The role of prosody in ASR

Basically, an ASR-system is nothing but capable to convert an incoming speech signal into a text, i.e. a string of characters, in a readable form. An algorithm perform-

ing ASR is an example of a very broad class of classification algorithms. These algorithms can be characterized by two major model parts: (A) a representation of the input signal, and (B) the actual classification of the input. In the representation within the ASR-context, the speech signal is transformed into a sequence of vectors, each referring to an 'acoustic state' or an 'acoustic event', depending on the precise implementation. The representation phase aims at the extraction of the most 'essential' information from the acoustic signal, where 'essential' means 'with respect to the required classification'.

In step (B), the actual classification (recognition) takes place. In recent ASR-systems, the classification consists of two parts: (B1) the creation of a list of so-called 'acoustic hypotheses', based on the acoustic information obtained in the representation, and (B2) a linguistic component, often in the form of a word bigram or a more sophisticated form of grammar. Step (B1) involves a lexical search, and the (time-consuming) construction of candidate word sets.

The role of prosodic information in ASR is primarily to facilitate the search by limiting the search space. Words (phoneme sequences) such as (Dutch) 'regent' ('rains') and 'regent' ('governor') can be desambiguated by indication of the word stress position. That means that one of these word candidates (both matching the acoustic hypothesis) can be skipped from the candidate list before a grammar is invoked. Such a reduction can take place if the stress information is fully reliably detectable from the speech signal. This, however, is not true in general: information about stress and other prosodic functions will in most cases have a statistical character. This means that the reduction of the search space is probabilistic, and the search reduction must be effected by adequate pruning of a tree of candidate outcomes.

Basically, there are three levels where prosodic information can play a role in ASR-systems: restriction of the lexical search space (by e.g. probabilistic stress prediction), by prosodic grouping (e.g. probabilistic detection of boundary tones), and by desambiguating different semantic interpretations.

Stress

Hieronymus (1989) and Campbell (1992) studied stress in relation to acoustic correlates such as duration and energy. Both these acoustic cues have to be normalized in order to be useful as a classification cue (to allow comparison across segments, utterances, speaking rates, speaking styles, speakers and so on). If this normalization is taken to be segment dependent, this implies that the recognition of prosodic features from the signal is secondary to the recognition of segments, and this is not the appreciated role of the prosodic information. One way out is to have a prior segmentation of the speech signal into broad phonemic categories. With respect to the boundaries between broadly defined segment categories, a segmentation result of about 80 percent (that is, 80 percent of all predicted segment boundaries within 20 ms from manually set segment boundaries) is feasible today. A vowel onset detection algorithm already performs more than 85 percent with an accuracy of 20 ms (Hermes, 1990). On the basis of the normalized segment durations and energy, an automatic stress prediction is feasible with an accuracy of at least about 70 percent (cf. Bagshaw, 1993).

Grouping

With respect to grouping, Lea (1979) proposed to use prosodic information to

detect boundaries of grammatical constituents. He tested several algorithms using F_0 and energy. There was an agreement of about 90 percent between predicted and actual major constituent boundaries with about 5–10 percent false alarms and about 10–15 percent misses. Later, this algorithm was improved by using additional information from the pitch contour. Since that time, many other authors studied elations between intonation patterns, acoustic pauses, prosodic boundaries, and grammatical boundaries (cf. the Proceedings of the ESCA-workshop on prosody, sept. 1993, Lund, Sweden, pp. 32–61; also Swerts, Bouwhuis, and Collier, 1994; De Pijper and Sanderman, 1994). In general, prosodic cues give information on prosodic phrasing and indirectly on grammatical structure (cf. Wichmann, 1993), but these effects may be language-dependent (cf. Nagano-Madsen, 1993).
Waibel (1986) used automatically derived prosodic cues (phrase boundaries, pitch, intensity, lexical stress levels) to reduce the search space (word candidate set) during lexical access. Hirschberg (1989) automatically determined prosodic cues to predict whether a question is a yes/no question or a wh-question. Butzberger (1990) used an HMM system to classify intonation contours for isolated words (into six categories) as well as boundary tones in continuous speech. A technical limitation has been the limited size of the training data set, but boundary tones were predicted with an accuracy of about 86 percent.

Both stress prediction as well as grouping can be used in so-called parse scoring. Probabilistic parse scoring on the basis of prosodic information is studied by Veilleux and Ostendorf (1993). For American-English, Veilleux (1994) provides information about computational models of the prosody-syntax mapping for speech. Basically, the question here is how the lexical search can be optimized by using prosodic information.

Semantics/pragmatics
The third application of prosodic information is in the semantic/pragmatic domain. Certain ambiguous sentences can be desambiguated by prosodic information. Well known examples are:
- vannacht is de vorstin gevallen (with second reading 'vannacht is de vorst ingevallen')
- 'Chinese teacher' (teacher of Chinese) versus 'Chinese teacher' (teacher from China)
- 'Does flight US 604 leave San Francisco on Friday or Thursday?', which is a yes/no question or a wh-question, depending on its intonation.

Also, certain types of disfluencies in spontaneous speech may be desambiguated by prosodic information. Hesitations and self-corrections are well known examples. Furthermore, prosodic information contains cues for discourse structure (Hirschberg, 1992; cf. Swerts *et al.*, 1994).

3 Automatic classification of pitch movements

In ten Bosch (forthcoming), it is studied how the acoustic-phonetic realizations of pitch movements in Dutch correspond to intonological categories, given by human transcribers within the framework developed by 't Hart, Collier and Cohen (1990). Within this framework, an intonation grammar is defined that defines the admissible sequences of pitch movements

for well-formed Dutch utterances.
A speech corpus have been transcribed by intonation experts. On the basis of a consensus transcription, an inventory of acoustic realizations of the most frequent intonation categories has been set up in order to study their statistical properties. It appears that the confusions between human transcribers can partially be explained by the acoustic overlap between realizations of different intonation categories. The mean overlap (in acoustic terms) between the five categories the accent-associated rise (labelled '1'), the non-accent-associated rise ('2'), the accent-associated fall ('A'), the non-accent-associated fall ('B'), and the rise-fall ('P'), is about 80 percent. That means that a classifier is able to separate these five classes with a classification rate of 80 percent. This result crucially depends on the representation. The input of the classifier consisted of the pitch contour over time and the instants of the vowel onsets. It has been shown in ten Bosch (forthcoming) that the classification of the pitch movement at a syllable requires a three-syllable window for the representation of the pitch movement. The results drops to about 70 percent if the classification is to be performed on a set of realizations associated with the five categories mentioned before, including a set of null-realizations (realizations that did not receive any of the five labels). A typical confusion matrix in the latter case is given below. This matrix has been obtained by using a three-syllable window in order to define the representation of the pitch contour, and a MultiLayer Perceptron (with one hidden layer) was used to transform the representation into a vector of a posteriori class probabilities. The matrix shows the summation of the responses of all data points used in the test set.

Resp.	stimulus					
	null	'1'	'2'	'A'	'B'	'P'
null	18	3	9	6	12	3
'1'	3	39	5	0	0	10
'2'	10	5	31	0	5	5
'A'	5	0	0	40	8	1
'B'	12	0	5	8	22	2
'P'	3	8	5	0	2	30

The categories '2' and 'B' are statistically less well defined that are the other categories (excluding the nulls) – as a consequence, many of the nulls tend to be classified as either '2' or 'B'. If two super categories {null, 2, B} and {1, A, P} are formed, the first one being not accent associated, the second accent associated, the separation between these super categories is over 88 percent (test set performance). This means that the pitch contour, together the vowel onsets contain enough information to predict sentential stress with a classification rate of over 88 percent.
Furthermore, it is shown that, although most of all realizations meet the properties as described in 't Hart, Collier and Cohen (1990), about 10 percent does not, even with respect to the pitch slope. The 10 percent of outlayers indicate that the relation between perceptual categories and acoustic realizations is not straightforward. The underlying cause is that the labelling of a training set is almost automatically subject to criterial noise, since the perception of intonation depends (in most cases) on intonation details that are not limited to the range of a syllable. Furthermore, grammatical constraints introduce noise in the classification of the individual pitch movements. As a consequence, the success of automatic classification of pitch movements, and of prosodic features in general, strongly depends on the theoretical framework behind the (human) labelling process.
Apart from the theoretical framework,

there are three other major issues to be addressed in the automatic classification of pitch movements. These are
(a) the bias in the database. Suppose that pitch movements have to be classified as belonging to one out of k categories $C_1, \ldots, C_k$. It makes an essential difference whether the number of learning samples in each category is equal or not. Unbalanced training sets introduce bias in the classification.
(b) The error measure. Statistically, a number of error measures are equally interesting and worth to be considered. In ordinary speech recognition systems, one mostly optimizes the likelihood $P(data \mid model(\lambda_1, \ldots, \lambda_n))$, n denoting the number of free model parameters, but this is only one of the interesting options (cf. Fukunaga, 1972)
(c) The classification power. If the number of model parameters in increased, the performance on an independent test set saturates to a level dependent on the model topology and the dataset itself.

4 Conclusion

Prosodic information may play a crucial role in automatic speech recognition systems. The probabilistic prediction of stress and of prosodic boundaries reduce the lexical and grammatical search spaces and therefore can be used in probabilistic parsing algorithms. A third domain of application is related to the semantic/pragmatic desambiguation of utterances, and the detection and interpretation of disfluencies in (spontaneous) speech.
We have shown in the automatic classification of pitch movements in terms of the framework developed by 't Hart *et al.* may reach a performance of about 80 percent in the case were only five (major) categories are to be separated; this result drops to 70 percent if so-called nulls are included.

Acknowledgement
This study has been sponsored by the Dutch Foundation for the Advancement of Science (NWO).

References

Bagshaw, P.C. (1993). 'An investigation of acoustic events related to sentential stress and pitch accents, in English.' Speech Communication, 13. pp. 333–342.
Ten Bosch, L.F.M. (forthcoming). 'On the automatic classification of pitch movements.' To be submitted to J. Acoust. Soc. Amer.
Butzberger, J.W. (1990). *Statistical methods for analysis and recognition of intonation patterns in speech.* Thesis, Boston university.
Campbell, W.N. (1993). 'Automatic detection of prosodic boundaries in speech.' Speech Communication 13. pp.343–354.
't Hart, J., Collier, R., and Cohen, A. (1990). *A perceptual study of intonation. An experimental-phonetic approach to speech melody.* Cambridge University Press, Cambridge.
Fukunaga, K. (1972) *Introduction to statistical pattern recognition.* New York: Academic Press.
Hermes, D.J. (1990). 'Vowel-onset detection.' J. Acoust. Soc. Am. 87, 866–873.
Hieronymus, J. (1989). 'Automatic sentential vowel stress labelling.' (Tubach, J.P. and Mariani, J.J., eds.) Proceedings of the first European Conference on Speech Communication and Technology (Eurospeech), Paris. pp. 226-229.
Hirschberg, J. (1989). 'Distinguishing questions by contour in speech recognition tasks.' Proceedings of the DARPA Workshop on Speech and Natural Language, October 1989. pp. 22–34.
Hirschberg, J. (1992). 'Intonational fea-

tures of local and global discource structure.' Proceedings of the DARPA Workshop on Speech and Natural Language, Feb. 1992, pp. 441–446.
Kraayeveld, J., Rietveld, A.C.M., and Heuven, V.J. van (1993). 'Speaker specificity in prosodic parameters.' In: Proceedings of the ESCA-workshop on prosody. Working Papers 41, Dept. of Linguistics, Lund Univ. pp. 264–267.
Lea, W.A. (1979). 'Prosodic aids to speech recognition.' In: Trends in Speech Recognition (Lea, ed.). Prentice-Hall, Englewood Cliffs, NJ. pp. 166.
Nagano-Madsen, Y. (1993). 'The grouping function of F_0 and duration in two prosodically diverse languages – Eskimo and Yoruba.' In: Proceedings of the ESCA-workshop on prosody. Working Papers 41, Dept. of Linguistics, Lund Univ. pp.46-49.
De Pijper, J.R. and Sanderman, A.A.(1994). 'On the perceptual strength of prosodic boundaries and its relation to suprasegmental cues.' J. Acoust. Soc. Amer., 96(4), 2037-2047.
Swerts, M., Bouwhuis, D.G., and Collier, R. (1994). 'Melodic cues to the perceived 'finality' of utterances.' J. Acoust. Soc. Amer., 96(4), 2064-2075.
Veilleux, N. and Ostendorf, M. (1993). 'Probabilistic parse scoring with prosodic information.' In: Proceedings of the ICASSP '93, Minneapolis. Vol II, p. 51.
Veilleux, N. (1994). *Computational models of the prosody-syntax mapping for spoken language systems.* PhD thesis, Boston University, 1994.
Waibel, A. (1986). *Prosody and speech recognition.* PhD thesis, Carnegie Mellon University.
Wichman, A. (1993). 'F_0 troughs and prosodic phrasing.' In: Proceedings of the ESCA-workshop on prosody. Working Papers 41, Dept. of Linguistics, Lund Univ. pp. 50-53.

SPONTANEOUS SPEECH PHENOMENA IN NAIVE-USER INTERACTIONS

P. Baggia, E. Gerbino, E. Giachin and C. Rullent

CSELT - Centro Studi e Laboratori Telecomunicazioni
Via G. Reiss Romoli 274 - 10148 Torino, Italy
paolo.baggia@cselt.stet.it

ABSTRACT

Dealing with spontaneous speech in a man-machine dialogue system rises a range of new problems, such as the presence of extra-linguistic phenomena (filled pauses, blows, coughs, etc.), restarts, out of dictionary words, and a large linguistic variability. The effects of those phenomena are evaluated on a corpus of 1,111 dialogue sessions acquired from 212 naive users who were let to talk to a (completely developed) dialogue system over the telephone. The application domain is voice access to train timetable information.

1. Introduction

A spoken dialogue system for voice access to train time-table information was developed at CSELT during the past few years. Other similar prototypes were developed in Europe for instance in the SUNDIAL ESPRIT Project or in the US in the ATIS ARPA project. Our system runs in real-time on a PBX telephone line and it was tested with a large number of *naive users*.

The prototype itself was used to acquire a large corpus of spontaneous speech during human-machine interactions. That made it possible to effectively study and analyse the presence of peculiar phenomena hardly found in read speech. In fact read speech has been initially used for research on speech recognition and understanding because it is easily acquired and more controlled. In a realistic environment such as naive user interactions, on the other hand, there are many peculiar phenomena that have to be dealt with. The presence of extra-linguistic phenomena such as filled and unfilled pauses, blows, coughs, clearings and so on, is higher than in read speech. There may be restarts and interruptions inside an utterance with self-correction or rephrasing. Finally, the user vocabulary and the linguistic structures are not constrained.

The acquisition of spontaneous speech is important in order to study naive user interactions (speech, vocabulary, and linguistic expressions), to evaluate speech and language technologies, and to train acoustic and language models for improving the recognition performances.

The goal of this paper is to describe the distribution and the effects of these phenomena on the CSELT corpus acquired with the spoken dialogue prototype over a one-year period. Section 3 briefly illustrates the modalities of the acquisition campaign and the prototype architecture. The following sections describe the most relevant spontaneous speech phenomena encountered (extra-linguistic phenomena, restarts and out-of-vocabulary words) and give a survey of the major techniques used for coping with them.

2. What is spontaneous speech?

The term *spontaneous speech*, as opposed to *read speech*, is used in the speech community to denote sentences uttered in a natural way in the course of human-human or, in our case, human-machine interactions. An example of a spontaneous speech utterance taken from our corpus, followed by the literal English translation, is:

(2.1) *"Allora **<pausa>** io vorrei sapere il primo treno che parte **<soffio>** da Milano **<soffio>** e arriva a Torino entro le ore **<ehhh>** otto entro le venti."*

"Then <pause> I would like to know the first train which leaves <blows> from Milano <blows> and arrives to Torino by the hour <ehhh> eight by twenty"

The main phenomena in (2.1) are the presence of an unfilled pause at the beginning, two blows in the middle of it, and a hesitation. The arrival time is repeated twice to avoid a possible confusion between a.m. and p.m. time. This kind of phenomena are very common in a human-human communication, that we hardly recognize their presence in our

everyday experience, but they are evident in a transcription of a spoken dialogue.

The most frequent ones are:

1. Filled and unfilled pauses, which may be further classified following their sound, such as a very long 'e' vowel, 'eh', 'uhm' and 'mmm'.

2. Noises produced by the speaker, such as blows, breathes, coughs, throat clearings, lip smacks, and so on.

3. Noises from the environment or, in our case, from the telephone line.

4. Restarts and interruptions.

5. Truncated words, frequent in the case of restarts and out-of-vocabulary words.

6. Partially ungrammatical sentences or sentences having a considerable linguistic complexity.

In the case of a recognition system those phenomena are likely to cause errors, because a recognizer has to hypothesize something even in the case of extraneous input. For instance, a blow or a filled pause may cause the insertion of a spurious word or, in the worst case, it may create a misalignment in the recognized sequence of words. A truncated or an out-of-vocabulary word has to be recognized as one or more words in the system vocabulary.

3. A corpus of spontaneous speech

A large corpus of spontaneous speech interactions was collected at CSELT from naive users over a one-year period. The data were collected using a complete human-machine dialogue system. This kind of acquisition was called "system-in-the-loop"[1] which is an alternative to the widely used "Wizard of Oz" modality, where an expert plays the role of a system, or only a part of a system, during a dialogue interaction. On the other hand, in the "system-in-the-loop" modality, the real system is tested, so that it is more realistic and the acquired data directly match the application conditions.

The prototype under test, partially developed under the Sundial Esprit project, was a spoken dialogue system, which allows accessing a remote DB in continuous speech using the telephone. The application domain consists of the Italian train time-table information.

[1] The "system-in-the-loop" modality has been used for the collection of spontaneous speech in the ATIS domain at MIT, see [Polifroni *et al.* 1991], and more recently in a test of an automatic train time table system at PHILIPS [Aust *et al.* 1994].

The system is composed by four main modules: the acoustic front-end (AFE), the linguistc processor (LP), the dialogue manager and message generator (DM), and the text-to-speech synthesizer (TTS). The AFE and the TTS are interconnected to the PBX through a telephone interface, while the DM is connected to a train timetable DB to obtain the data. For a complete description of the prototype under test see [Clementino and Fissore 1993].

A total of 212 users (52.8% female and 47.2% male) have been testing the system. The large majority of them where people contacted outside CSELT, while 7% of them where people who contributed to develop the system itself. External users had no knowledge about the system or the involved technologies. Users were selected in the age range from 18 to 75, with different levels of education as in Figure 3.1.

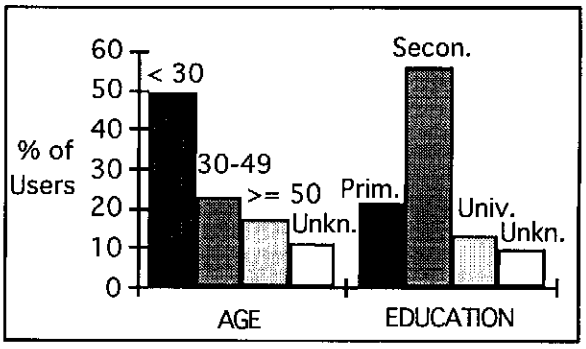


Fig 3.1. Users' age and educational level

Users were invited to our labs and were asked to test an automated telephone service able to provide information about Italian train timetables and train characteristics (type of train, fares, presence of sleeping cars, restaurants, number of classes, extra charges, required reservation, etc.) They received a single printed page of instructions which contained a brief explanation of what type of information could be provided by the system. The users were asked to imagine they had to organize a trip between two cities and collect some information about the trains that matched their requirements. To precisely determine whether the data gathered by the users were correct, predefined scenarios were employed. Each scenario specified the departure and arrival cities, selected among the set of 100 cites of the available railway DB, and the train attributes to be collected during the dialogue, while the user was let free to specify the departure time. In order not to constrain the user vocabulary and linguistic expressions, only pictorial scenarios were used. The

test was unsupervised, since the users were left alone in an isolated room. At the end of all the dialogue interactions, each user had to express his/her opinions about the system and to point out difficulties and problems encountered during the dialogues through an interview or a questionnaire.

Each user was involved in 5 dialogue sessions on the average. Each session took about 5 minutes, with 18 dialogue turns (a turn consists of a user utterance and system answer pair). The dialogues were quite long because during a dialogue a user was instructed to ask 4 or 5 attributes of the train connections of interest. Also, the dialogues were lengthened due to the presence of recognition and understanding errors; in that case the dialogue manager initiates recovery strategies through focused requests and restricted modalities, such as isolated word recognition and spelling. For a more detailed description of the dialogue strategy see [Gerbino and Danieli 1993].

Utterances	No.	Ave.Length
All	8,720	6.2 words
First	1,111	9.6 words

Table 3.1. Summary of the collected data

In Table 3.1 is shown a summary of the acquired data. 8,720 continuous speech user utterances were collected, with an average utterance length of 6.2 words. The first utterance of the dialogue was completely unconstrained, so that it is longer (9.6 words on the average), with respect to the remaining utterances which are more focused and so shorter.

All the speech material acquired, 17 hours of speech,was manually transcribed (1GByte of data - 2 CDROMs).

4. Extra-linguistic Phenomena

The presence of extra-linguistic phenomena in a user utterance is very common, in fact in the course of a dialogue the speaker has to do many things in parallel: to think how to continue the interaction, to interpret the system answer and to properly formulate the next intervention. For those reasons, a pause or a hesitation may be useful for the user to take time, delay the speech, and to be capable of formulating more effectively the request.

During the transcription of the speech material, we have classified the occuring phenomena into 10 classes. In other similar studies many more classes were used for the initial transcriptions, which are however clustered into a smaller set for later operation ([Kubala *et al.* 1992]).

Phenomena	Instances		Utterances		Users	
	No.	%	No.	%	No.	%
E open	217	11.2	197	2.3	72	34.1
Ehs	426	21.9	360	4.3	107	50.7
Ehms	60	3.1	58	0.7	39	18.5
Uhms	87	4.5	80	0.9	55	26.1
Blows	696	35.8	651	7.8	113	53.6
Breathes	99	5.1	99	1.2	45	21.3
Lip Smacks	128	6.6	128	1.5	67	31.8
Coughs	10	0.5	10	0.1	9	4.3
Clearings	34	1.8	34	0.3	23	10.9
Other noises	185	9.5	180	2.2	91	43.1
Totals	1942	100.0	1590	19.0	188	89.1

Table 4.1. Extra-linguistic phenomena in 8720 user utterances

Table 4.1 shows the frequence of occurrence for each phenomenon, the number of utterances[2] and the number of users for which that phenomenon occurred. 19% of the utterances contains at least one phenomenon. The frequence of occurrence in the total number of utterances is high (20%); these utterances are not specific to a restricted number of users, but are spread out in the large majority of them. This result is consistent with similar results for many languages, such as English (ATIS corpus [MADCOW 1992]), German/Spanish (multi-language database [Woszczyna *et al.* 1994]) and Japanese ([Kuroiwa *et al.* 1993]).

It is very hard to predict the position of such phenomena because there is only a vague tendency to occur in a specific point of the utterance. There are some regularities, however: some phenomena most often occur at the beginning or at the end of the utterances. For instance the lips smacks lie at the beginning of an utterance in 86% of the cases, while the 'mmm' filled pauses are likely to occur in the middle of an utterance, and the blows are in the 77% of the cases at the end of the utterance. All of them seem to occur more frequently between syntactic boundaries: they were found to lie 90% before an article and 83% before a preposition.

[2] In Utterances (Users) columns, totals are lower than the sum of the column numbers, because many Utterances (Users) may show more than one phenomenon and percentages refer to the whole corpus.

Regarding utterance length, the extra-linguistic phenomena have a lower incidence on the short utterances (the more frequent in the corpus), as shown in Figure 4.1.

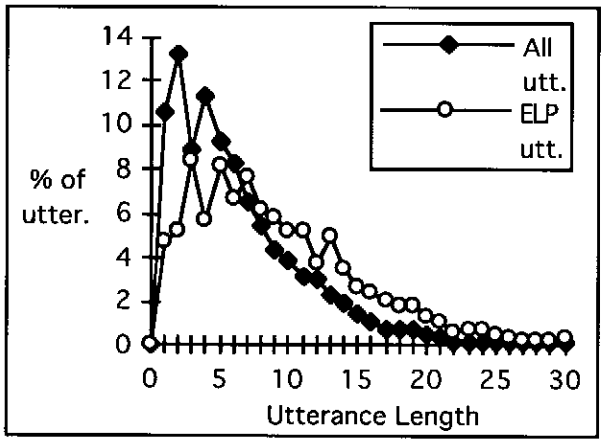


Fig 4.1. Utterances length comparison between all utterances and utterances with ELP

4.1. First experiments

In the literature the most effective approach to cope with extra-linguistic phenomena is to model them explicitly and then to recognize them in parallel with the words in the vocabulary, so that when a phenomenon is present in an utterance the recognizer should be capable of skipping it without affecting the surrounding words. This approach was used in [Ward 1989] and [Kubala *et al.* 1992].

In this case a large corpus of spontaneous speech may be necessary for training specific acoustic models, but for many of them there are very few instances for training a robust model anyway, and the only possibility is to cluster them into a smaller number of generalized models that covers different phenomena. In our case 3 models where trained:

1. a blow model;

2. a hesitation model, which covers all kinds of hesitations (E open, Ehs, Ehms, Uhms);

3. a noise model, which covers all the rest.

In future, using a larger database it will be possible to further specialize them.

The metrics used for the detection of those phenomena are the recall rate (RCL) and the precision rate (PRC). The recall rate is the percentage of phenomena that are correctly identified (n_c) vs. the total number of phenomena really present in the utterances (n_t).

$$RCL = \frac{n_c}{n_t} \bullet 100$$

The precision rate is the percentage of phenomena that are correctly identified (n_c) vs. the total number of phenomena that are identified (n_g), the latter includes also the false alarms possibly generated.

$$PRC = \frac{n_c}{n_g} \bullet 100$$

Both metrics give an idea of the performance of a model: the recall is about how good it matches the real occurrences, while the precision is about how well it does not generate false alarms.

Models	Instances		No_lm		Bigrams	
	#	%	RCL	PRC	RCL	PRC
Blows	72	36.3	84.7	89.7	87.5	78.8
Hesitations	82	41.2	53.7	77.2	78.0	51.6
Noises	45	22.5	57.8	16.2	62.2	13.7
Total	199	100.0	65.8	46.0	77.9	38.0

Table 4.2. Experiment on 858 user utterances

Table 4.2 shows the results of a first experiment for evaluating the performance of the models, on a separate test set of 858 spontaneous speech utterances which has not been used in the training. The experiments were carried out using two configurations: without any language model (No_lm columns) and with bigrams (Bigrams columns).

The blow model shows higher RCL and PRC in both configurations. This is probably due to the fact that it is a specific model, robustly trained on a large set of instances; moreover, it is less confused with the voiced speech. As regards the hesitation and the noise models, they show a good improvement of RCL in the Bigram configuration, but the PRC of the noise model is still very low. This was caused by the attempt to model low-energy noises and so it was easily confused with the silence. Finally the PRC decreases when a language model is used; this is caused by the difficulty to model the position of those phenomena.

5. Restarts

Restarts are another common class of problems found in spontaneous speech corpora. The frequence of restarts in English utterances was evaluated between 5-10% ([Hindle 1983] and [Shriberg *et al.* 1992]); in the CSELT corpus it is 5.4%. A few examples are the following, where a restart was transcribed in the form[3]: " ... [<wrong part> | <interruption part> | <correction part>] ...". In many cases the <interruption part> is missing and the point between the <wrong part> and the <interruption part> is referred to as the interruption site.

(5.1) *"[da | da] Firenze"*
"[from | from] Firenze".

(5.2) *"[vers- | verso] le venti"*
"[nea- | near] twenty".

(5.3) *"L'orario di partenza dei treni [dalla | da] Porta Nuova"*
"The time of departure of trains [from-the | from] Porta Nuova".

(5.4) *"[è un | è un] treno intercity?"*
"[Is it an | Is it an] intercity train?.

(5.5) *"[c'è il serviz- | c'è il servizio] di ristoro?"*
"[Is there the serv- | Is there the service] of restoration?"

(5.6) *"Il prezzo del biglietto [fino a Venezia | fino a Firenze]"*
"The fare of the ticket [to Venezia | to Firenze]".

(5.7) *"vorrei sapere [quando parte il | !eh | a che ora parte il] treno !eh del cinque luglio da Alessandria a Vercelli"*
"I want to know [when it leaves the | <ehh> | at what hour leaves the] train <ehh> on the fifth of July from Alessandria to Vercelli".

(5.8) *"[qual'è la città | !eh | dove bisogna cambiare treno]?"*
"[which is the city | <ehh> | where is required to change train]?"

The examples show that the restarts are a very variable phenomenon. Sometimes in the interruption site there is a fragment of a word, 50% of the restarts in CSELT corpus and 73.3% for English utterances. The fragment of the word is sharp; it does not respect even the syllable boundaries. When the speaker decides to stop it stops immediately. In many cases only a word is repeated (5.1-5.3), in other cases a phrase of different length is repeated (5.4-5.7) and sometimes the interrupted utterance is rephrased form the beginning (5.8). When only a word is repeated, in the large majority it is a preposition or an article.

In the interruption site there is often a small pause[4], but in 17% of the restarts there is also a hesitation and in a small fraction (2%) a cue word, such as "no", "then" or "that is". In long utterances there could be many restarts, possibly nested, as in:

(5.9) *"vorrei sapere se [è un tre- | !eh | è un treno festivo] [o f- | o anche feriale]?"*
"I want to know if [is it a tra- | <ehh> | is it a week-end train] [or w- | or even worker]?"

(5.10) *"!eh per cortesia !e [vorrei | !m | [vorrei prend-] | !eh | vorrei partire da Agrigento]"*
"<ehh> excuse me <e> [I would | <uhm> | [would leav-] | <ehh> | I would leave from Agrigento]".

The role of prosody may be very relevant in the restarts, as shown at the Edinburgh University ([Lickley *et al.* 1991] and [Lickley and Bard 1992]) where listener where able to identify a restart even in the presence of low filtered acoustic signal.

The goal of the on-line identification of restarts is a difficult topic because there is an intersection of different kinds of knowledge that play a role in this process. The first attempts where *text-based* and tried to extend a traditional parser to handle ungrammatical input, such as [Fink and Biermann 1986] and [Carbonell and Hayes 1983]. An attempt to parse restarts is described in [Hindle 1983]; this approach is essentially syntactic, based on a deterministic parser, and it relies on an explicit marker of the interruption site, called *editing signal*. This marker should be of phonetic nature and detected from a lower level. As of yet this marker has yet to be found and many approaches take into account different knowledge sources to identify the interruption site. At SRI Bear, Dowding and Shriberg [Bear *et al.* 1992] use a pattern matching technique to find the utterances that may contain restarts and syntactic-semantic knowledge to rule out wrong candidates. This method was tested on the transcription of more that 10,000 spontaneous

[3] Recently Nakatani and Hirschberg ([Nakatani and Hirschberg 1994]) have proposed the Repair Interval Model (RIM) for describing a repair. In that model the <wrong part> is called "Reparandum Interval", the <interruption part> is called the "Disfluency Interval" and the <correction part> is the "Repair Interval".

[4] We tested a proposal made in [O'Shaughnessy 1992] that pausal duration might be used to identify a repair site. In our corpus the pausal duration in an interruption site is not distinguishable from other pauses.

speech utterances with the results of 76% of recall and 62% of precision. More than half of the identified utterances was properly corrected.

Another approach ([Nakatani and Hirschberg 1994]), called *speech-based*, takes into account even the acoustic and prosodic features of the restarts. The study propose the RIM (Repair Interval Model) to model the restarts in an utterance and a technique based on Classification and Regression Trees (CART) for their localization. This technique uses a large number of features, such as the duration of the pause on the boundary of two words, the presence of word fragments, cue words and hesitations, F0 measures, and even morpho-syntactic features. On an independent test-set of 148 utterances, it gives 86% of recall and 91% of precision, a very interesting result. For the time being, this technique is quite complex to be integrated in a recognizer.

Finally, in the SLT system [Agnas *et al.* 1994] the repair detection was faced in a pragmatic way. On the output of a traditional recognizer, DECIPHER, which generates a restricted number of N-best, a set of pattern-matching rules detects the presence of repairs and hypothesize a correct version of the utterance. The CLE parser, then, generates the meaning representation for all the understood sentences and even of the corrected one. Finally the sentence which scores higher than the other is selected. The score takes into account different aspects such as the acoustic score and the results of the parsing. This method of correcting the repairs, besides its simplicity, has a low value of recall, but a high precision, which means that it does not generates too many false starts.

6. Out-of-vocabulary words

Current recognizers are unable to cope with words not present in the vocabulary of the system, i.e. out-of-vocabulary words, so that, if such words are uttered the recognized will try to decode the best match with other words in the vocabulary. For instance "*hai detto*" ("did say") was decoded as the word "*biglietto*" ("ticket") and "*l'importo*" ("the fare") with the syntactically uncorrect sequence "*in parto*" ("in leave"). In the worst case the presence of out-of-vocabulary words may generate a dis-alignment in the decoded sequence of words.

In our system the vocabulary contained 762 words, selected studying the application domain, but it was not known a priori if the naive users would use that words. Actually the vocabulary used by the naive users only partially matches the vocabulary of the system, because of the 1044 different words present in the user utterances only 53% are in system's vocabulary, the rest are out-of-vocabulary words. The total of words considering the repetitions is 54,634.

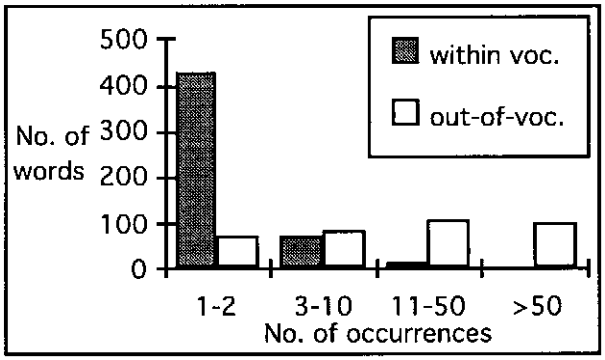


Fig 6.1. Frequence of occurrence of words within and out-of-vocabulary

Although the percentage of out-of-vocabulary words is very high, each of them recurs only a few times, as shown in Figure 6.1, where the words in the system's vocabulary and the out-of-vocabulary words are distributed in classes of frequence.

The growth of the dictionary depends even on the modality of acquisition. Polifroni, Seneff and Zue at MIT ([Polifroni *et al.* 1991]) have compared the growth of the user dictionary in utterances acquired with two different modalities. Wizard of OZ at TI and "system-in-the-loop" at MIT. The growth rate in the first case (about 50 words per 100 utterances) is higher than in the second one (about 20 words per 100 utterances), because the presence of an integrated system constraint much more the user utterances that the Wizard of Oz. The growth rate in our corpus (about 14 words per 100 utterances), depicted in Figure 6.2, is similar to the MIT one and furtherly confirms their observation.

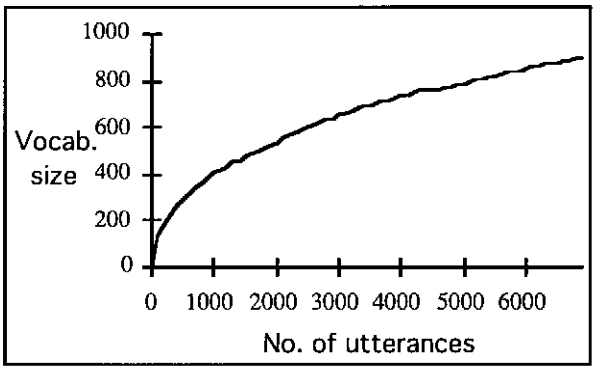


Fig 6.2. Vocabulary growth increasing the user utterances

The conclusion is that for this kind of application domains there exixts a small core vocabulary, widely used by the naive users. The size of this core vocabulary is of a few hundred words (without considering proper names, i.e. railways city and station names in the DB in use). Those words in the core vocabulary covers a large percentage of the words used by the naive users. As in Figure 6.3, where is shown the coverage of the words in the user vocabulary on the total number of words, considering even the repeated ones. The first 200 words covers the 90% of the words in the user utterances.

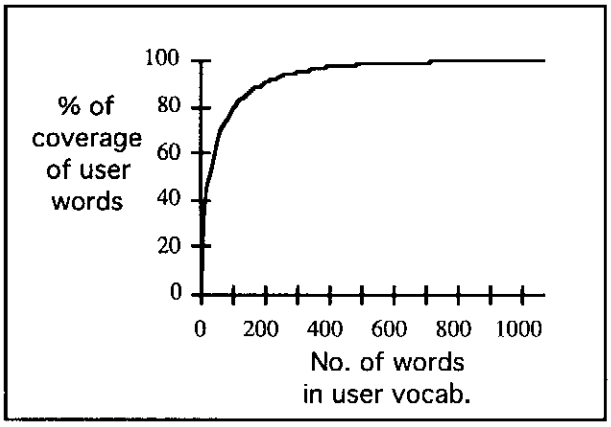


Fig 6.3. Coverage of user words by the user vocabulary

The words used by naive users shares a lot of words, but there is a great number of words that are hardly predictable, so that an application in a restricted domain should have an extremely large vocabulary or a small vocabulary that covers the large majority of words, but a certain amount of them are left outside of it.

The methods proposed in the literature to handle out-of-vocabulary words are summarized in [Itou *et al.* 1992] in four classes:

1. The system recognized the utterances using a fixed vocabulary, but if the acoustic score of the recognized sentence is very low, the sentence may be rejected.

2. The system recognize the presence of out-of-vocabulary words, but it is not able to distinguish them.

3. Like the previous point, but with the possibility of identify the approximate transcription of the out-of-vocabulary words.

4. Like the previous point, but it is able to infer the meaning of the word and the syntactic category.

Those strategies are more ambitious from the point 1 to point 4 and the last one gives the idea of an auto-correcting system, see [Young 1993], but it seems a bit far from the actual technology. The first one is very sharp because it does not distinguish different sources of error, so that a speaker with a poor acoustic match may not use the system.

The strategies of type 2 and 3 were studied at BBN ([Asadi *et al.* 1990]) modeling an out-of-vocabulary word with a generalized model (a free sequence of two or more phonemes). The model was used only in specific classes of words, such as the proper names, with a very constraining language model. Using a similar methodology for modeling an out-of-dictionary word and a bigram language model for allowing the presence of out-of-vocabulary words in different classed was studied ([Suhm *et al.* 1993]).

A different methodology tries to give a confidence measure to all the recognized words, so that it is possible to discriminate which part of an utterance may be wrongly recognized or, eventually, contain out-of-vocabulary words. In a second time is possible to use this cue to reprocess the low confidence part of the utterance or to use it in the dialogue strategy. The idea was realized ([Young and Ward 1993]) using a normalized acoustic score, which is obtained combining the acoustic score of the recognizer and a score of the recognition of a string of phonemes. The latter one should represent a measure independent of the vocabulary in use. The normalized score is then used to obtain a confidence measure using histograms, which represent the distribution of the normalized score of a word when it is correctly recognized and when it is misrecognized. For the longer words it is possible to detect wrong occurrences with a good confidence, but for other classes of words, such as short function words, it is impossible.

7. Linguistic Issues

Studies which analyse the language of spontaneous speech human-machine interactions are very rare. Some experiments performed at Dundee University, described in [Kennedy *et al.* 1988], have compared the language of written communications to either a system or a person. Actually, the system was simulated by an expert. The results show that the language used talking to a system is very simple, with a restricted focus to a few previous interactions and with short utterances. In addition the lexical richness and the use of anaphora is reduced. Although these results are encouraging, in a spoken

human-machine communication there are all the spontaneous speech phenomena previously described, which complicate the interaction.

In [Moore 1993] are described the most common types of non-standard utterances found in 15000 ATIS utterances, which includes sentence fragments, sequence of fragments, fragments combined with complete sentences, repairs of different kinds, and many ungrammaticalities. All those phenomena are commonly found in the CSELT corpus. Parsing and interpreting these utterances require using robust parsing algorithms. In our system a partial parsing strategy ([Baggia Rullent 1993]) is used which relies on the idea of always trying to understand something, eventually incomplete and sometimes partially wrong. The error recovery is left to the dialogue strategy. Other robust parsing methods were recently proposed, such as [Ward 1991], [Seneff 1992], [Dowding *et al.* 1993], and [Hanrieder and Heisterkamp 1994].

Another important topic is the influence between the dialogue context and the next user utterance, in the CSELT corpus the user utterances where classified by the dialogue context. Some preliminary analyses show that in a focused context, such as the request of a specific attribute (departure and arrival places, time of departure, etc.), the user replies are in the large majority very short and elliptical. Only 13% includes a verbal form and 6% are very complex and possibly ill-formed. On the other hand, long and complex sentences easily contain many spontaneous speech phenomena. In fact, there is evidence that extra-linguistic phenomena are correlated with repairs and out-of-vocabulary words. Oviatt shows in [Oviatt 1993] that the increase of the disfluency rate in spontaneous speech utterances is statistically related to the length of the utterance and the lack of presentation format. These data give an idea that a realistic system should include a suitable dialogue strategy, which channels the user to interact with brief sentences, and this should reduce the influence of many spontaneous speech phenomena.

8. Conclusions

Speech recognition and understanding of spontaneous speech requires to cope with a large number of phenomena, such as: extra-linguistic phenomena (filled pauses, blows, coughs, etc.), restarts, out of dictionary words and ill-formed sentences. The presence of these phenomena was verified in a large corpus of spontaneous speech utterances acquired from naive users interacting with an automated system.

The first fact is that each phenomenon is statistically relevant; 19% of the utterances contains extra-linguistics phenomena, 13% out-of-vocabulary words and 5.4% restarts. Each phenomenon can cause errors at different levels in the system.

The extra-linguistic phenomena may be effectively modeled with specific acoustic models which require the acquisition of large number of instances to train many models robust enough. The out-of-vocabulary words are common, but there is a small core of highly used words. The restarts are a difficult problem because their identification requires the interaction of many different levels of knowledge.

Finally it will be useful to better study such phenomena through a continuous use of a spoken dialogue system in the "system-in-the-loop" modality.

References

[Agnas *et al.* 1994] M-S. Agnas, H. Alshawi, I. Bretan, D. Carter, K. Ceder, M. Collins, R. Crouch, V. Digalakis, B. Ekholm, B. Gamback, J. Kaja, J. Karlgren, B. Lyberg, P. Price, S. Pulman, M. Rayner, C. Samulesson, T. Svensson, *Spoken Language Translator: First Year Report*, Technical Report SRI: CRC-043, January 1994.

[Asadi *et al.* 1990] A. Asadi, R. Schwartz, J. Makhoul, "Automatic Detection of New Words in a Large Vocabulary Continuous Speech System", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Albuquerque, NM, p. 125-128, April 1990.

[Aust *et al.* 1994] H. Aust, M. Oerder, F. Seide, V. Steinbiss, "Experience with the Philips Automatic Train Timetable Information System", in *Proc. 2nd Workshop on Interactive Voice Technology for Telecommunication Applications (IVTTA94)*, Kyoto, p. 67-72, September 1994.

[Baggia and Rullent 1993] P. Baggia, C. Rullent, "Partial Parsing as a Robust Parsing Strategy", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Minneapolis, MN, p. I-126-130, May 1993.

[Bear *et al.* 1992] J. Bear, J. Dowding, E. Shriberg, "Integrating Multiple Knowledge Sources for Detection and Correction of Repairs in Human-Computer Dialog", in *Proc. of the 30th Annual Meeting ACL*, Newark, DE, p. 56-63, 1992.

[Carbonell and Hayes 1983] J. Carbonell, P. Hayes, "Recovery strategies of parsing extragrammatical language", in *Am. Jou. of Computational Linguistics*, vol. 9, n. 3-4, p. 123-146, 1983.

[Clementino and Fissore 1993] D. Clementino, L. Fissore, "A man-machine dialogue system for speech access to train time-table information", in *Proc. of the 3rd EUROSPEECH*, Berlin, p. 1863-1866, September 1993.

[Dowding *et al.* 1993] J. Dowding, J.M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, D. Moran, "Gemini: A Natural Language System for Spoken-Language Understanding", in *Proc. of the 31st Annual Meeting ACL*, Columbus, Ohio, p. 54-61, 1992.

[Fink and Biermann 1986] P. E. Fink, A. W. Biermann, "The correction of ill-formed input using history-based expectation with application to speech understanding", in *Computational Linguistics*, vol. 12, n. 1, p. 12-36, 1986.

[Gerbino and Danieli 1993] E. Gerbino, M. Danieli, "Managing dialogue in a continuous speech understanding system", in *Proc. of the 3rd EUROSPEECH*, Berlin, p. 1661-1664, September 1993.

[Hanrieder and Heisterkamp 1994], G. Hanrieder, P. Heisterkamp, "Robust Analysis and Interpretation in Speeh Dialogue", in H.Niemann, R. de Mori, G. Hanrieder, *Progress and Prospects of Speech Research and Technology*, Infix, Monaco, p. 204-211, 1994.

[Hindle 1983] D. Hindle, "Deterministic Parsing of Syntactic Non-Fluencies", in *Proc. of the 21st Annual Meeting ACL*, Cambridge, MA, p. 123-128, 1983.

[Itou *et al.* 1992] K. Itou, S. Hayamizu, H. Tanaka, "Detection of Unknown Words and Automatic Estimation of their Transcription in Continuous Speech Recognition", in *Proc. of the Int. Conf. of Spoken Language*, Banff, Alberta, p. 799-802, October 1992.

[Kennedy *et al.* 1988] A. Kennedy, A. Wilkes, L. Elder, W.S. Murray, "Dialogue with machines", in *Cognition*, vol.30, p. 37-72, 1988.

[Kuroiwa *et al.* 1993] S. Kuroiwa, K. Takeda, N. Inoue, I. Nogaito, S. Yamamoto, "Online Collection of Spontaneous Speech Using a Voice-Activated Exchanger", in *Proc. of the Int. Symp. on Spoken Dialogue*, Tokyo, p. 25-28, November 1993.

[Kubala *et al.* 1992] F. Kubala, C. Barry, M. Bates, R. Bobrow, P. Fung, R. Ingria, J. Makhoul, L. Nguyen, R. Schwartz e D. Stallard, "BBN BYBLOS and HARC February 1992 ATIS Benchmark Results", in *Proc. DARPA Speech and Natural Language Workshop*, p. 72-77, 1992.

[Lickley *et al.* 1991] R. J. Lickley, R. C. Shillcock, E. G. Bard, "Processing disfluent speech: How and when are disfluencies found?", in *Proc. of the 2nd EUROSPEECH*, Genoa, Italy, p. 1499-1502, September 1991.

[Lickley and Bard 1992] R. J. Lickley, E. G. Bard, "Processing disfluent speeh: recognising disfluency before lexical access", in *Proc. of the Int. Conf. of Spoken Language*, Banff, Alberta, p. 935-938, October 1992.

[MADCOW 1992] MADCOW, "Multi-site data collection for a Spoken Language System", in *Proc. DARPA Speech and Natural Language Workshop*, p. 7-14, 1992.

[Nakatani and Hirschberg 1994] C. H. Nakatani, J. Hirschberg, "A corpus-based study of repair cues in spontaneous speech", in *J. Acoust. Soc. Am.*, vol. 95, n. 3, p. 1603-1616, March 1994.

[O'Shaughnessy 1992] D. O'Shaughnessy, "Analysis of false starts in spontaneous speech", in *Proc. of the Int. Conf. of Spoken Language*, Banff, Alberta, p. 931-934, October 1992.

[Oviatt 1993] S. Oviatt, "Predicting Spoken Disfluencis During Human-Computer Interaction", in *Proc. of the Int. Symp. on Spoken Dialogue*, Tokyo, p. 53-56, November 1993.

[Polifroni *et al.* 1991] J. Polifroni, S. Seneff, V. W. Zue, "Collection of Sponteneous Speech for the ATIS Domain and Comparative Analyses of Data Collected at MIT and TI", in *Proc. DARPA Speech and Natural Language Workshop*, p. 360-365, February 1991.

[Seneff 1992] S. Seneff, "Robust Parsing for Spoken Language Systems", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, San Francisco, p. I-189-192, March 1992.

[Shriberg *et al.* 1992] E. Shriberg, J. Bear, J. Dowding, "Automatic Detection and Correction of Repairs in Human-Computer Dialog", in *Proc. DARPA Speech and Natural Language Workshop*, p. 419-424, 1992.

[Suhm *et al.* 1993] B. Suhm, A. Woszczyna, A. Waibel, "Detection and Transcription of New Words", in *Proc. of the 3rd EUROSPEECH*, Berlin, p. 2179-2182, September 1993.

[Ward 1989] W. Ward, "Modelling Non-verbal Sounds for Speech Recognition", in *Proc. DARPA Speech and Natural Language Workshop*, p. 47-50, October 1989.

[Ward 1991] W. Ward, "Understanding spontaneous speech: the PHOENIX system", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Toronto, p. I-365-367, May 1991.

[Woszczyna *et al.* 1994] M. Woszczyna, N. Aoki-Waibel, F.D. Buo, N. Coccaro, K. Horiguchi, T. Kemp, A. Lavie, A. McNair, T. Polzin, I. Rogina, C.P. Rose, T. Schulz, B. Suhm, M. Tomita, A. Waibel, "JANUS94: Towards Spontaneous Speech Translation", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Adelaide, p. I-345-348, May 1994.

[Young and Ward 1993] S.R. Young, W. Ward, "Learning New Words from Spontaneous Speech", in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Minneapolis, p. II-590-591, April 1993.

[Young 1993] S.R. Young, *Learning New Words from Spontaneous Speech: A Project Summary*, Technical Report CMU: CMU-CS-93-223, Luglio 1993.

Simple Speech Recognition with Little Linguistic Creatures

M.F.J. Drossaers and D.A. Dokter

Computer Science Department,
University of Twente,
P.O Box 217, 7500 AE Enschede,
The Netherlands,
email: mdrssrs@cs.utwente.nl, D.A.Dokter@let.rug.nl

Abstract

In this paper we present a stochastic neural-network architecture, the synchronous-network acceptor, that can approximately simulate nondeterministic finite-state automata, where the precision of the approximation depends on the level of noise in the synchronous-network acceptor. This network learns to simulate finite-state automata by means of an unsupervised learning algorithm. The synchronous-network acceptor is a neurophysiologically plausible model of connected relay nuclei, or connected, coherent, collections of cortical columns. The learning algorithm is a plausible model of a column formation process or, more generally, an axon terminal segregation process.

Complex self-organizing systems can be constructed as ensembles of synchronous-network acceptors. In virtue of the fact that finite-state automata can be simulated, applications of such systems can be implementations of linguistic theories, for instance parsing algorithms. However, we prefer a reductionistic, holistic approach. The approach is reductionist in that linguistic behavior is studied via its neurophysiological substrate. Our approach is holistic because we consider complete linguistic systems. These systems are called little linguistic creatures. They typically contain models of nervous sensory, motivational, and motor subsystems, and unlike in more conventional approaches in which various linguistic faculties are studied separately, we reduce the complexity of the subject by modeling simple linguistic systems. This holistic approach should in the end have the advantage of domain independence over more conventional approaches.

The system of self-organizing synchronous-network acceptors we present in this paper is a very simple linguistic creature for speech recognition. Its state provides the semantics of the linguistic input and the context information for the resolution of nondeterministic choices.

1 Introduction

In this paper we present ideas and results concerning speech recognition in the context of integrated linguistic analysis with neural networks. The approach to language analysis pursued here strongly deviates from what is usually encountered in articles on language analysis, therefore we will first give an exposition of this approach before turning to the technical aspects.

1.1 The Linguistic Starting Point

Define language$_2$ as the set of publicly accessible, or objective, observations of linguistic utterances. This set defines the phenomenon of language as it is manifest in speech and writing. It is, among other things, the language that is studied in the various directions in linguistics that originate from the work of Chomsky. In [4] Chomsky states that:

> A grammar of the language L is essentially a theory of L. Any scientific theory is based on a finite number of observations, and it seeks to relate the observed phenomena and to predict new phenomena by constructing general laws in terms of hypothetical constructs .... Similarly, a grammar of English is based on a finite corpus of utterances (observations), and it will contain certain grammatical rules (laws) stated in terms of the particular phonemes, phrases, etc., of English (hypothetical constructs).

Clearly Chomsky regards here a grammar as a body of natural laws of natural language, not as empirical generalizations. Once one has such a theory, a grammar, one can, as Chomsky did [6], go on and state that if a person A shows behavior that conforms to the grammar, A has knowledge of the grammar. Then the empirical theory is extended to apply to what is inside A's head. Chomsky calls such a theory mentalistic [5].

> The theory of competence is mentalistic, naturally, in that it can at the present stage of knowledge draw no evidence from and make no direct contribution towards the study of the mechanisms that may realize the mental structures that form the subject matter for this theory, or that carry out the mental processes that it studies. Thus the theory of competence (i.e. the theory of grammar) deals with abstract structures, postulated to account for and explain linguistic data.

The lack of knowledge that Chomsky refers to is, incidentally, the mind-brain problem, one of the hardest, if not *the hardest*, philosophical problem known. The point we would like to make here is that a theory of language is nowadays a mentalistic theory of language$_2$; that is, it is not about brain structures or brain activity.

Elements and subsets of language$_2$ can be subjected to integrated linguistic analysis. We define linguistic analysis as the assignment of meaning representations to an element of language$_2$. A meaning representation is in general something like a feature structure, a parse tree, a discourse representation structure, etc. Such an analysis is defined to be integrated if several different kinds of analyses are performed concurrently while constraining each other interactively. The main advantages of integrated linguistic analysis are that a multitude of analyses lead to a richer meaning representation, and that, by constraining each other interactively, they suppress a combinatorial explosion of meaning representations. Usually, the analyses rest on publicly accessible, or objective, knowledge about language$_2$, such as lexical, syntactical, and semantical or domain knowledge.

However, we are not interested in analyzing language$_2$ using publicly accessible knowledge. We define language processing as an activity of people as language users, where processing can be either interpreting or producing language$_2$. The knowledge that language users use for processing is private; that is, inaccessible to others. We conclude then that linguistic analysis is really an operation involving the private knowledge of a language user. Consequently, we are interested in a theory on how language users, people, process language$_2$. This is an interest in a holistic position.

Since the knowledge used for linguistic analysis is not publicly accessible, we cannot make general statements about linguistic analysis. Therefore we switch to a domain that we can have objective knowledge about: the underlying physical structure of language users. More precisely we turn to the nervous system, since it is linguistically the most relevant constituent of language users.

1.2 Reductionism and Holism

This approach presupposes a form of reductionism: a relation between mental processes, among which linguistic processes, and brain activity, see [7] for a discussion of various kinds of reductionism.

The first aspect of our reductionism is that it is in general impossible to refer to mental events, notably by means of language$_2$: a person can refer to his own mental events, but not to anyone else's. Therefore, mental events can appear only connotatively in discourse. Consequently theories on mental events describe "hypothetical constructs", which have to be inferred from overt behavior. So, since they do not at all deal with genuine mental events, theories on mental events cannot be reduced to brain theories.

The relation of mental events, being private, to brain processes is of a quite different nature. In neurosurgical operations Penfield electrically simulated the temporal lobes of the brain in conscious patients. This led to reports of vivid experiences of past events. Repeated stimulation of the same point led to the same response see [13] (page 1006). From this we conclude that brain activity produces mental events. This is, of course, not the conclusion that any brain activity necessarily produces conscious experiences, which ignores the complex functional organization of the brain. The reduction we put forward here is that mental events are cooperative phenomena of brain activity. That is, a mental event is an emergent property of the activity of many neurons, see also [1].

According to McEnery [14] holism (holisticism) explains the phenomena that a system produces, not in terms of the properties of the effects, but in terms of the system that produced those effects.

A holistic philosophical system based on biological premises, that we believe to be reductionistic in the above sense, has been elaborated by Umberto Maturana. *In general* we agree with this position, as it is presented in [15]. We will review some main characteristics of that presentation, in order to derive some engineering principles from these characteristics later in this section.

The first aspect of his work to be discussed here is the closure of the nervous system. This is a noncontroversial issue in neurophysiology, see e.g. [13]. The point is that perceptions are constructed by the nervous system. For instance, the visual cortex contains systems that react to what we perceive as line segments, colored spots, and moving forms. Three dimensional visual experiences have to be constructed from these rudiments. Also the auditory cortex registers frequencies. Specific sound patterns, such as coherent speech have to be distinguished from a noisy background. Etcetera, etcetera.

This means for one thing that one can only perceive what the brain can construct, which depends in turn on its structure. The brain thereby constrains what we can know. It defines the set of things to be possibly known. However, since the structure of the brain may change, by learning, the set of knowable things may change.

The second aspect to be discussed is that neural systems belong to living organisms that try to survive by acting.

The third aspect is that living organisms come to act successfully with respect to survival in a structural adaptation process, which takes the form of evolution and learning. The adaptation proceeds by selection of successful structural configurations, thus maximizing the success in survival of the organism. This entails that the structure of an organism reflects the knowledge it has of its environment. It defines its world with respect to survival. Maturana defines cognition as acting in a way that is relevant to self-maintenance [15].

The fourth aspect is that of mutual and recursive adaptation of an organism to other organisms. The result is a domain of cooperative behaviors that subserves the survival of the organisms involved. Maturana calls such a domain a linguistic domain. According to Maturana, and we subscribe to this position, language is such a domain of behaviors.

It need perhaps be remarked that this position is neither solipsistic nor behavioristic. Solipsistic theories entail that we cannot know our world because we have no perceptual access to it. Maturana escapes solipsism by defining knowledge as the result of adaptive interaction. This includes adaptation of perception: organisms perceive a world that is matched to their survival.[1] Behaviorism claims that the behavior of an organism is a function of its environment. Maturana turns the matter around and claims that the environment is a function of the organism, hence his position is not behavioristic.

To emphasize: Maturana's position on the relation between language and the physical substrate of language is that language is not physical. It is behavior. Nevertheless, the knowledge to use language is embodied in physical structures: the physical structure is a procedural definition of language processing, and the forms language can take depend on the possible structural configurations.

This philosophy entails that language is interpreted and produced in the context of the language user: his or hers other perceptions, his or hers moods, or actions in which he or she is engaged. This requires contributions from sensory, motor and motivational nervous systems. The role of sensory and motor systems will be immediately clear. The motivational system is the seat of the emotions, it regulates instinct behavior, and initiates action. As such, it is the basis of the emotional interpretation of all perceptions, and the basis of intentions, see [13]. Consequently, it provides the basis of the semantics and pragmatics of language, both in the integrated analysis a language user makes of his or hers linguistic perceptions, as in speech production.

1.3 Artificial Language-Users

Our intention is to construct artificial language-users on a reductionistic, holistic basis. We will first discuss a number of aspects of artificial language-users in this and subsequent sections, and later define one in section 6. We start by listing a number of characteristics of artificial language-users, based on Maturana's work.

If a system is an artificial language user, then

1. It is autonomous; that is:

   (a) It has a number of system parameters.

[1] One may wonder whether everything should be related with survival. Aren't we civilized people? Yes, we are. But on the other hand, the numerous wars in former eastern bloc countries, notably Yugoslavia, show very clearly that in order to survive we better be civilized.

(b) It has sensors that assign values to the system parameters.

(c) A relation is defined on the system parameters, possibly implicitly by the structure of the system.

(d) The system optimizes, autonomically, the relation between the parameters by acting.

2. It is adaptive.

People are very complex, Language processing systems. To get a grip on the problem of how to construct artificial language-users, we adopt the research strategy of first constructing very simple systems, and to increase the complexity stepwise.

The metaphorical name for these systems is *little linguistic creatures*. This need not necessarily be animal-like robots with an artificial language-capacity. Any system that satisfies the above principles will do.[2]

What, more precisely, is a little linguistic creature? A little linguistic creature consists of a model nervous system, a model body, and it is confronted with a model environment. Its nervous system consists of a system of neural networks and an unsupervised learning algorithm. The model nervous system will consist of all three of sensory, motor, and motivational systems. The model body of a little linguistic creature could be a robot, or peripheral computer machinery, or a collection of data in a file. For formal convenience we will assume the last option. The environment could be the real world, a computer simulation, or again a data file. At this point we will opt for files too.

The neural network model we use, is a neurophysiologically plausible model. With this we mean that it rests on neurophysiological premises. Also the learning algorithm rests on neurophysiological premises. Of course, being a model, it is to a larger extent implausible than it is plausible from a neurophysiological viewpoint. However, we desire a simple model. We don't care about anatomical details as long as the model is a close functional approximation of the modeled substrate. The reason why we desire simplicity is that we want to have solvable networks. With solvable network and learning algorithm we can have a *theory* of reductionistic holistic language processing, which can be used to construct artificial language-users by means of computation instead of trial and error.

[2]For all we care this may include programs that explore cyberspace on the Internet.

The knowledge of functional aspects of the nervous system is limited, see [7], especially concerning higher cognitive functions. Most neurophysiological knowledge is anatomical. We have chosen the following strategy to deal with this. We can regard the nervous system as consisting of two-layer modules connected by projection neurons, see [10]. A layer is either a relay nucleus, or a coherent collection of cortical columns, where we regard a collection of columns coherent if they receive projecting axons from the same group of neurons, and also project on the same group of neurons. Note that we distinguish these layers on anatomical grounds. Distinct layers may have distinct anatomical properties, such as the kinds of resident cells, and we have tried to synthesize all those properties in our model layers. For instance, Cortical columns contain both excitatory and inhibitory local interneurons. We have modeled this by excitatory local feedback against a strong inhibitory background provided by a multilevel threshold. The threshold increases in case there is activity in a layer, but can be overcome by cooperative excitatory input. The threshold suffices to model both feed forward inhibition and feedback inhibition, see [13].

The organization of the layers, in columns for instance, and the organization of the projections is in the nervous system to a large extend dependent on learning and adaptation. This is functional organization. Cortical columns are functional, not anatomical entities. In our model this functional organization is brought about by self-organization also. The learning algorithm models a number of cellular learning processes: Hebbian column formation, Hebbian association within layers, and classical conditioning.

The merit of this strategy is that we can use anatomical, neurophysiological knowledge to define models that will then have a similar function, in the proper context. The modeling of two-layer modules in the brain by the synchronous-network acceptor defines a lower bound on the modeling resolution. Below that level all structure, such as column formation is due to self-organization.

By evolution people have become the language-users they are now. It seems fruitless and useless to try and redo evolution. Therefore, we borrow from the structure of the human nervous system to construct a little linguistic creature: knowledge about how language is to be processed is embodied in the structure of the nervous system. Also, structures for other tasks can be borrowed from the human nervous system, and other ner-

vous systems. The precise functional definition of a task is then embodied in the system by adaptive interaction.

We thereby map the distinction between structure and function onto the distinction between network and learning algorithm. We bring evolution into the structure of the little linguistic creature while doing this. The implanting of neurophysiological knowledge is restricted to neural structures. This goes down perhaps to the level of the language faculty, but does not specify a language.

The learning algorithm we use is unsupervised. This property is required by the closure of the model nervous system. The unsupervised character of the learning process holds only on microscopic level, that of two-layer modules, not on the level of complete little linguistic creatures. The unsupervised learning process allows the functional structure of the little linguistic creature to be dependent on afferent information only. since we would like to be able to predict the effect of the algorithm deep inside a little linguistic creature also the learning algorithm needs to be solvable.

Another matter is that not much is known about how the brain processes language. However, the synchronous-network acceptor is equivalent to a nondeterministic finite-state acceptor, and the learning algorithm can be used to store one in a synchronous-network acceptor. This fact can be used to hypothesize neural structures that perform some linguistic task. We cannot only work from neurophysiology to linguistic fact, but also the other way around. A drawback is that it will then not be known at which abstraction level the network models the brain: the modeling relation is lost. It could be that a neuron matches a real neuron, or a complete brain organ. Since we hypothesize neural structure, we do not destroy the closure of the nervous system. But because we, at that point, start from mentalistic theory, we do not commit reductionism, but merely implement the mentalistic theory. By the equivalence relation, the synchronous-network acceptor may be expected to do this very well. We have a strong preference for a reductionist approach, but also prefer a practical engineering attitude above principled impotence.

Note in passing that, since we concentrate on artificial language-users, we loose the domain specificity that characterizes language$_2$ systems. Here we deal with closed self-organizing systems, so we don't need to define a problem or an application domain. Descriptions of these domains enter the system via self-organization. In file-based systems, for example, it suffices to collect a number of examples from the domain and to offer them to the system for learning. In a more realistic setting, little linguistic creatures can do their own sampling.

1.4 Contents

In this paper we define systems of synchronous-network acceptors and relate them to the general theory of synchronous-network acceptors. This is done in section 5. Prior to this we first define the synchronous-network acceptor in section 2 and 3, and we define the unsupervised learning algorithm in section 4. Many results where obtained for network acceptors and the learning process. Since detailed presentation of those results does not add to the current discussion, the reader is frequently referred to the original sources. We also provide informal descriptions of the operation of both the synchronous-network acceptor and the learning algorithm that should give the reader an intuitive understanding of both subjects. Systems of synchronous-network acceptors are defined in section 5, where we also explain new details, and relate systems of synchronous-network acceptors to the concept of a little linguistic creature. In section 6 we report on a first experiment with systems of synchronous-network acceptors, and decide whether we have constructed a little linguistic creature. The paper ends with the conclusions in section 7.

2 The Synchronous-Network Acceptor

In this section we will first define the network formally, and then describe its intended operation. We start with the neurons.

Definition 2.1
A *formal neuron* is a state variable $S \in \{0, 1\}$. A formal neuron is usually called a neuron. $S = 1$ models an active neuron, and $S = 0$ models a neuron at rest. □

A collection of neurons forms a network state.

Definition 2.2
A *neural network state* is a set

$$\{S_i \mid i = 1, \ldots, N\}$$

of neurons. This is written as $\{S\}$. Its ordered representation is a vector

$$\boldsymbol{S} \;=\; (S_1, S_2, \ldots, S_N)^T .$$

- A neural network state is usually called a network state.

□

specific network states are activity patterns.

Definition 2.3
An *activity pattern* is a neural network state

$\{\xi_i \mid i = 1, \ldots, N\}$.

This is written as $\{\xi\}$.

- Its ordered representation is a vector

  $$\boldsymbol{\xi} \;=\; (\xi_1, \xi_2, \ldots, \xi_N)^T .$$

- A bit ξ in an activity pattern is a random variable with distribution $\mathcal{D}$.

□

If there are p activity patterns, then this is usually written as $\{\xi^\mu\}$, with $\mu = 1, \ldots, p$, in order to discern the activity patterns. In the context of learning, p is often very large, but finite. Hence the following corollary.

Corollary 2.4
The number and the length of the sequences that are stored in the temporal synapses of the recognition network are finite. □

We have not defined here a specific probability distribution for the bits ξ. This is because many different distributions should be allowed for. However, most properties of the network and the learning algorithm have been established for uniform distributions, which will be defined in general below.

Definition 2.5
Let $\{\xi\}$ be an activity pattern. A *uniform probability distribution* for a bit ξ in $\{\xi\}$ is

$$\Pr(\xi) \;=\; a\delta(\xi - 1) + (1 - a)\delta(\xi),$$

where δ is the Dirac delta and $a \in [0, 1]$ is the *activity level* or *bias*:

$$\frac{1}{N}\sum_{i=1}^{N} \xi_i \to a \quad \text{if} \quad N \to \infty.$$

If $a \neq \frac{1}{2}$ the pattern is biased. □

For reference purposes we give a provisional definition of the synaptic matrices. The synaptic coefficients are obtained by the unsupervised learning process that will be discussed in the next section.

Definition 2.6
The matrices of synaptic interaction are J^{temp} and J^{ext}.

- J^{temp} is the $(N \times N)$ *matrix of synaptic coefficients of the temporal synapses.*
- J^{ext} is the $(N \times N)$ *matrix of synaptic coefficients of the external input synapses.*
- Synaptic coefficients may also be called synapses.

□

Network states produce neural input via the synapses.

Definition 2.7
Given network states $\{S(t)\}$ and $\{S'(t)\}$,

- The *external input* on neuron i at time $t + \delta t$ is

  $$h_i^{ext}(t + \delta t) \;=\; \sum_{j=1}^{N} J_{ij}^{ext} S_j'(t),$$

  where δt is a small constant that models synaptic delay and equals the neuronal cycle time.

- The *temporal input* on neuron i at time $t + \delta t$ is

  $$h_i^{temp}(t + \delta t) \;=\; \sum_{j=1}^{N} J_{ij}^{temp} S_j(t).$$

- The *total neuronal input* on neuron i at time t is

  $$h_i(t) \;=\; h_i^{ext}(t) + h_i^{temp}(t).$$

□

The overlap of the network state with a specific activity patterns is measured by the overlap parameter.

Definition 2.8
The *overlap of a network state with a pattern* $\{\xi^\mu\}$ is

$$m^\mu(t) \;=\; \frac{1}{N}\sum_{i=1}^{N} \xi_i^\mu S_i(t).$$

The *largest temporal average* is

$$m(t) = \max\{m^\mu(t) \mid \mu = 1, \ldots, p\}.$$

□

A related quantity that determines the effect of the noise on the overlap parameter and has a role in the size of the threshold, to be defined below, is the noise correction factor.

Definition 2.9
The *noise correction factor* is a real number $m \in [0, 1]$ such that

$$m = \tfrac{1}{2}\left(1 + \tanh\left[\tfrac{1}{4}m\lambda^{temp}/T\right]\right),$$

where T is the level of noise in the network. □

The threshold comes next.

Definition 2.10
The *threshold U at time t* is

$$U(t) = m(\lambda^{ext} - \tfrac{1}{2}\lambda^{temp}) + m(t - \delta t)\lambda^{temp}.$$

□

This is a multi-level threshold. Its size depends on $m(t - \delta t)$, which registers whether there was a significant overlap of the state of a network with a known activity pattern in the previous neuronal time cycle. If so, the threshold is high.

In case of a network with which no external input synapses make contact, and that has no temporal synapses, we provide virtual input.

Definition 2.11
The *virtual input* on neuron i is

$$h_i' = m(\lambda^{ext} + \lambda^{temp}).$$

□

In such special cases we also use a virtual threshold.

Definition 2.12
The *virtual threshold U'* is

$$U' = m(\lambda^{ext} + \tfrac{1}{2}\lambda^{temp}).$$

□

The magnitude of the virtual threshold is an elaboration of

$$U' = m(\lambda^{ext} - \tfrac{1}{2}\lambda^{temp}) + m\lambda^{temp}.$$

Now we can define the synchronous-network acceptor.

Definition 2.13
The *synchronous-network acceptor* is defined by its architecture, its synaptic interactions, and its neural dynamics.

- The *architecture* of the synchronous-network acceptor consists of two layers, or networks, see figure 1.
  - The recognition network consists of N neurons: $S_1, \ldots, S_N$.
  - The input network also consists of N neurons: $S_1', \ldots, S_N'$.
- There are two types of *synaptic interaction* in the synchronous-network acceptor.
  - J^{ext} defines how neural activity in the input network is relayed to the recognition network.
  - J^{temp} defines the temporal, or recurrent, relation between patterns of activity in the recognition network.
- The neural dynamics are defined by the *neuronal firing probabilities*.

  $$\Pr(S_i' = 1) = \frac{1}{1 + \exp[-(h_i' - U')/T]},$$

  $$\Pr(S_i' = \zeta_i^\mu) =$$
  $$\Pr(S_i' = 1)\delta(1 - \zeta_i^\mu) + (1 - \Pr(S_i' = 1))\delta(\zeta_i^\mu).$$

  When no external input is present, it is assumed that $\{\zeta^\mu\} = \{0\}$.

  $$\Pr(S_i = 1) = \frac{1}{1 + \exp[-(h_i - U)/T]}, \text{ and}$$

  $$\Pr(S_i = 0) = 1 - \Pr(S_i = 1).$$

  Where:
  - δ is the Dirac delta.
  - The neural dynamics depends on parameters λ^{temp} and λ^{ext}, for which holds that $p\lambda^{temp} < \lambda^{ext}$, with p the number of activity patterns that are used to form the temporal synapses.

□

An additional constraint imposed on $\lambda^{temp} \ll \lambda^{ext}$ is that for noise levels below the pseudo-critical noise level, see [9], the relative synaptic strength λ^{ext} is large enough to reduce the firing probability of neurons that receive at most only temporal input, to an insignificant value. The following assumption concerning the network size will be adopted for the rest of this paper, except at places where it is explicitly stated otherwise.

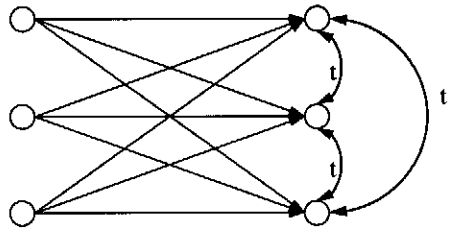


Figure 1: *A schema of the architecture of the synchronous-network acceptor. The three leftmost circles represent the input network and the three rightmost circles represent the recognition network. The straight arrows represent the external input synapses, and the arched arrows represent the temporal synapses.*

Assumption 2.14
N is very large; that is, $N \gg 1$. □

Concerning the timing of the input we adopt the next assumption.

Assumption 2.15
Let $\{\xi^1\}, \ldots, \{\xi^p\}$ be a sequence of activity patterns, let θ be the time interval during which $\{S'\} = \{\xi^\mu\}$, with $\mu \in [1, p]$. Let $k \in \mathbb{N} \cup \{-1\}$, and let $x \in [0, \theta)$, where $[0, \theta) \subset \mathbb{R}$. Then,

$$\{S'(t)\} = \{\xi^\mu\} \text{ implies } \exists k \, \exists x : t = (k + \mu)\theta + x.$$

□

The time interval θ should be chosen to satisfy $\theta \geq 2\delta t$.

3 Synchronous Network Dynamics

Synchronous dynamics is distinguished from asynchronous dynamics, see [9], by the fact that in asynchronous dynamics some neurons may not be updated in one unit period of time, whereas in synchronous dynamics all neurons are necessarily once updated in every unit period of time, and also at the same moment in time.

Definition 3.1
A *unit period of time* is the basic entity in a discrete representation of continuous time. □

Definition 3.2
An *update of a neuron* is a re-computation of its state in accordance with definition 2.13. □

Definition 3.3
The *neuronal cycle time*, δt, is the period of time that is required to update a neuron. In synchronous dynamics the size of the neuronal cycle time satisfies

δt = 1 unit period of time.

□

Network dynamics defines an updating procedure for neurons in both the input network and the recognition network.

Definition 3.4
In *synchronous dynamics* both all N neurons of the input network and all N neurons of the recognition network are updated once within one unit period of time, at integral multiples of the neural cycle time. □

The intended operation of the network is as follows. Assume that a sequence of patterns $\{\zeta^1\}, \ldots, \{\zeta^p\}$ when successively assigned to the input network, produces a sequence $\{\xi^1\}, \ldots, \{\xi^p\}$ in the recognition network, if the threshold is low. Assume also that the temporal synapses provide excitatory temporal input on the active neurons of $\{\xi^{\mu+1}\}$ if the prior network state equaled $\{\xi^\mu\}$. Assume further that starting at $t = \mu\delta t$, $\{S'\} = \{\zeta^\mu\}$ for a period δt, where $\mu = 1, \ldots, p$. Initially, at $t = \delta t$, the recognition network is in a state of coma: $S_i = 0$, for $i = 1, \ldots, N$, and the input network is in a state that equals $\{\xi^1\}$. Assume that prior to that time both networks were silent, then the threshold is low and the recognition network can be driven by the external input neurons alone. Consequently, it will assume state $\{S\} = \{\xi^1\}$. Each neuron in the recognition network that has been active for a period δt provides temporal input on the neurons in the recognition network. Also after a period δt the next external input arrives. In cooperation with the high threshold temporal input performs a control function: any neuron that receives external input but no temporal input, and vice versa, cannot become active. The confidence of the control function depends on the size of the earlier overlap. All neurons that receive both external and temporal input become active, and so on.

4 The Unsupervised Learning Process

There are two main cellular learning processes at work in the brain: Hebbian learning and modulator learning [12]. We argued elsewhere [10] that both learning processes presuppose three groups of neurons: (1) the neurons from which the synapses originate, (2) the neurons onto which the synapses make contacts, and (3) the neurons that activate the neurons in group (2), in case of Hebbian learning, or the neurons that activate the synapses that change, in the case of modulator learning. Note that in both cases the learning process is a binary relation between groups of neurons: a relation between the presynaptic neurons and the postsynaptic neurons in case of Hebbian learning, and a relation between the presynaptic neurons and the modulator neurons in case of modulator learning.

In our model unsupervised learning comprises two distinct learning processes, one concerns projections, the external input synapses, and the other concerns temporal associations. Learning projections is a form of either competitive Hebbian learning or modulator learning and learning temporal associations is a form of (non-competitive) Hebbian association. Each learning process is modeled by one algorithm. This means that we have defined one model for both Hebbian learning and modulator learning, being the algorithm for projection learning. We do this by assuming that in Hebbian learning the third group of neurons is a modulator of the activity in the postsynaptic neurons, and consequently that both kinds of learning are really two kinds of modulator learning. Therefore, the algorithm mentions only the input neurons and the modulator neurons.

What about the synapses from the modulator neurons to the postsynaptic neurons or to the synapses? Here we assume synaptic connections that are accurate without learning, see [13] for details.

Definition 4.1
P_i and P_m are *sets of activity patterns*.

- P_i is the set

  $\{\{\zeta^\nu\} \mid \nu = 1, \ldots, p\}$

  of p activity patterns of the input network.
- P_m is the set

  $\{\{\xi^\mu\} \mid \mu = 1, \ldots, r\}$

  of r activity patterns of the modulator network.

□

Definition 4.2
SUP_i and SUP_m are sets of superscripts.

- SUP_i, the *set of superscripts in* P_i, is

  $\mathrm{SUP}_i \;=\; \{1, \ldots, p\}$.
- SUP_m, the *set of superscripts in* P_m, is

  $\mathrm{SUP}_m \;=\; \{1, \ldots, r\}$.

□

We will also refer to elements of sets of superscripts as *words*. We will refer to a space-separated sequence of words as a *sentence*.

Definition 4.3
Let SUP_y be a set of superscripts. A *sentence* v *over* SUP_y is a finite sequence of words μ such that:

1. $v = \mu$, with $\mu \in \mathrm{SUP}_y$, or
2. $v = w\,\mu$, with w a sentence over SUP_y and $\mu \in \mathrm{SUP}_y$.

The number of words in a sentence v, its length, is denoted by $|v|$. □

Definition 4.4
Let SUP_y be a set of superscripts.

- $\mathrm{SUP}_y^\star$ is the set of all sentences over SUP_y.
- SUP_y^+ is the set of all nonzero length sentences over SUP_y.

□

Definition 4.5
A *learning set* $\mathcal{L}$ is a set of pairs of sentences that satisfies:

$\mathcal{L} \subseteq$
$\{(v, w) \mid v \in \mathrm{SUP}_i^+, w \in \mathrm{SUP}_m^+, \text{ and } |v| = |w|\}$.

□

Definition 4.6
Let $u, x \in \mathrm{SUP}_i^\star$ and $y, z \in \mathrm{SUP}_m^\star$. The *learning procedure* satisfies the following features:

1. At $t = n\delta t$, with $n = 0, 1, 2, \ldots$ the input network and the modulator network adapt their state to a pattern $\{\zeta^\nu\} \in P_i$ and a pattern $\{\xi^\mu\} \in P_m$ respectively. The pair (ν, μ) is such that $u\nu x = v$, $y\mu z = w$, $|u| = |y|$, $|x| = |z|$, and $(v, w) \in \mathcal{L}$.

2. Successive pairs of words (ν, μ) are selected from a pair of sentences (v, w) in a left to right order.

3. Successive pairs of sentences (v, w) are selected from learnset $\mathcal{L}$ in random order.

□

One run of this procedure is called an *epoch*. The learning procedure is repeated a pre-specified number of epochs.

Learning proceeds in two phases. In the first phase the projections are defined. Then the input network can drive the recognition network, and the temporal relations are established. In the first phase we impose an extra condition on a learnset.

Definition 4.7
A *learning set for projection learning* $\mathcal{L}$ is a set of pairs of strings that satisfies:

$$\mathcal{L} \subseteq \{(v, w) \mid v \in \mathrm{SUP}_i, w \in \mathrm{SUP}_m\}.$$

□

The learning process we model presupposes a growth process that results in initial, small synaptic contacts. The size of the initial synapses is a decreasing function of the distance between the presynaptic neuron and postsynaptic neuron.

Definition 4.8
Let M be an $(N \times N)$ matrix. The *initialization function init* is given by:

$$\mathrm{init} : N \times N \to \mathbb{R},$$

such that $\mathrm{init}(i, j)$ assigns a value to each component M_{ij} of M according to:

$$M_{ij} = \begin{cases} \frac{-(k-i)h}{w} + h & \text{if} \quad i \le k \le i + w, \\ \frac{(l-i)h}{w} + h & \text{if} \quad i - w \le l < i, \\ 0 & \text{otherwise.} \end{cases}$$

where $j = k \bmod N$ or $j = (l + N) \bmod N$, depending on the case under consideration. The real parameter $w \in [1, 1/2N]$ is half the size of the support of init and $h \in \mathbb{R}$ is its maximum. Always take $w \times h < \lambda^{ext}$. □

Now we define projection learning.

Definition 4.9
Let $\{S(n)\}$ and $\{S'(n)\}$ be the states of the modulator network and the input network in the nth unit time period respectively. Let $\rho \in [0, 1]$ be the learning rate. The *components of* J^{ext} are defined by infinite iteration to be:

$$\begin{aligned} J_{ij}^{ext}(0) &= \mathrm{init}(i, j), \\ J_{ij}^{ext}(n+1) &= J_{ij}^{ext}(n) + \rho \Delta J_{ij}^{ext}(n), \\ \Delta J_{ij}^{ext}(n) &= J_{ij}^{ext}(n) \langle S_i(n) \rangle \times \\ &\quad \left[\left(\langle S'_j(n) \rangle - \frac{\sum_{k=1}^{N} J_{ik}^{ext}(n) \langle S'_k(n) \rangle}{\sum_{l=1}^{N} J_{il}^{ext}(n)} \right) \right. \\ &\quad \left. + \frac{\langle S'_j(n) \rangle}{\lambda^{ext}} \left(\lambda^{ext} - \sum_{k=1}^{N} J_{ik}^{ext}(n) \right) \right]. \end{aligned}$$

Parameter ρ is the learning rate and $\langle S \rangle$ is the average of S. □

The learning algorithm for the external input synapses is competitive. All neurons in the recognition network learn independently of the other neurons.

The learning algorithm tends to maximize the variance in the distribution of the synaptic strength over the synapses that synapse onto a single neuron in the recognition network. That is, if the P_i is large ($|P_i| \approx 2^N$) and consists of activity patterns that are randomly chosen according to a uniform probability distribution, and the modulating network is kept at full activity during the adaptation of the external input synapses, and if also $w = \frac{1}{2}N$, then a single neuron in the input network will win all the synaptic strength, of size λ^{ext}, and the others will be zero. The neuron to win the competition most likely is the neuron that has initially the largest synapse.

Since the learning process is uniform over all neurons in the recognition network, this result will be the same throughout the recognition network, because the patterns come from a uniform distribution.

The learning algorithm is sensitive to the relative frequency with which neurons in the input network participate in an activity pattern. Consequently, nonuniform distributions may define a strong preference for the winner, or propose a number of candidate winners by assigning a high probability to the active state of one or more neurons.

The learning algorithm is also sensitive to high correlations among the active neurons. Groups of

neurons that are active together with high probability, tend to win the competition with smaller groups an single neurons. Only neurons whose activity is fully correlated can win together.

The above two features define the learning process as a form of feature extraction: neurons that are predominantly active in the input patterns will win most synaptic strength, and consequently determine the activity of most neurons in the recognition network.

A smaller w restricts the competition to a fraction of the neurons in the input network. It is a means to record high activity frequencies or high correlations in local regions of the input network. A smaller w allows a larger number of features to be recorded in the synapses.

The algorithm converges very fast to the maximum variance in the distribution of the synaptic strength over the synapses, see [10]. The learning rate is essentially there to slow the learning algorithm down, and thus prevent convergence before the learning set has been processed once.

After learning, each neuron in the recognition network has synaptic contacts with only a few neurons in the input network. For later reference we will call an activity pattern of the input network that is characterized by activity in those neurons and nowhere else, the feature vector for that neuron.

In the second phase, temporal relations are stored in the temporal synapses. This is done by means of the following learning algorithm.

Definition 4.10
Let $\{S(n)\}$ and $\{S(n-1)\}$ be states of the recognition network in the nth and $(n-1)$th unit time period respectively, with $n \geq 0$ and $\{S(-1)\} = \{0\}$. Let $\rho \in [0,1]$ be the learning rate. The *components of* J^{temp} are defined, by infinite iteration, to be:

$$
\begin{aligned}
J_{ij}^{temp}(0) &= 0, \\
J_{ij}^{temp}(n+1) &= J_{ij}^{temp}(n) + \rho \Delta J_{ij}^{temp}(n), \\
\Delta J_{ij}^{temp}(n) &= \langle S_i(n) \rangle \langle S_j(n-1) \rangle \times \\
&\left[\frac{(J_{ij}^{temp}(n) + \varepsilon)}{\lambda^{temp}} \left(\lambda^{temp} - \sum_{k=1}^{N} J_{ik}^{temp}(n) \langle S_k(n-1) \rangle \right) \right. \\
&\left. + \frac{J_{ik}^{temp}(n)}{\lambda^{temp} C_d} \right] - \frac{J_{ik}^{temp}(n)}{\lambda^{temp} C_d},
\end{aligned}
$$

where ε is a small positive constant, and $C_d = |\mathcal{L}|.\max\{|v| \mid v \in \mathcal{L}\}$. □

This is a form of (non-competitive) Hebbian learning. Unlike in the first phase a standard learning set can be used. The sentences of the learning set are recorded in the temporal synapses in terms of the activity patterns in the recognition network that the input patterns activate.

The decay is chosen such that temporal relations that are not reinforced once an epoch, deteriorate and vanish. This provides a means to prevent the registration of temporal correlations between sentences in the learning set, and it removes small, but undesired, temporal correlations that resulted from the first phase.

The learning algorithm converges only in terms of average changes of the synapses. Due to the ε in the prefix, and extreme activity levels in the input patterns that may occur because of their random character, the temporal synapses will continue to oscillate around either zero or around a value that is a contribution to the sum of the synapses onto a temporal successor that amounts to λ^{temp}, see [10]. So for both learning algorithms average results of learning can be related to the distribution of the activity patterns, and the parameter choices without actually using the learning algorithm. That is, the learning algorithms are solvable.

For each synchronous-network acceptor a set of feature vectors can be defined that generate a set of orthogonal patterns of activity in the recognition network. This definition proceeds by adding a feature vector to the set for each neuron in the recognition network that is not already activated by another feature vector.

To synchronous-network acceptors many results on noisy-network acceptors [9], as well as a number of results on neural-network acceptor [8], carry over, see [10]. This is, however only for synchronous-network acceptors with a set of feature vectors, as defined above, as the external input vectors and a set of suitable modulation patterns. These results allow questions concerning the average behavior of the synchronous-network acceptor to be answered without simulating the network. One special result is that in the limit $T \to 0$ the synchronous-network acceptor is equivalent to a nondeterministic finite-state acceptor. For nonzero T this equivalence holds approximately.

5 Systems of Synchronous-Network Acceptors

In general, systems of synchronous-network acceptors can be formed by replacing the external input network of one synchronous-network acceptor by the recognition network of another synchronous-network acceptor. Next we use the Extended Backus-Naur Form (EBNF) [11] to define a specification syntax for systems of synchronous-network acceptors.

Definition 5.1
The *syntax of the specification language for little linguistic creatures* is

llc	=	*networks connections inputNets parameters*.
networks	=	"Networks" "=" "{" "{" *netName learnMode evolMode* "}" {"{" *netName learnMode evolMode* "}"}"}".
netName	=	*name*.
name	=	char {char}.
char	=	"a" |...| "z" | "A" |...| "Z".
learnMode	=	"AND" | "OR".
evolMode	=	*learnMode*.
connections	=	"Connections" "=" "{" "{" *netFromnetTo anatomy sign* "}" {"{" *netFrom netTo anatomy sign* "}"}"}".
netFrom	=	*netName*.
netTo	=	*netName*.
anatomy	=	*name*.
sign	=	"Excitatory" | "inhibitory".
inputNets	=	"InputNets" "=" "{" "{" *inputNet learnSet inputSet* "}" {"{" *inputNetlearnSet inputSet* "}"}"}".
inputNet	=	*netName*.
learnSet	=	*name*.
inputSet	=	*name*.
parameters	=	"Parameters" "=" "{" "NumNodes" *number* "NetDimensions" "1" | "2" "LambdaE" *number* "LambdaT" *number* "Rho" *number*"."*number* "W" *number*"."*number* "H" *number*"."*number* "Epochs" *number* "Noise" *number*"."*number*"}".
number	=	1|2|....

□

In this definition we have taken the liberty to abbreviate (very) long series of alternatives by means of ellipses. This is not a proper use of the EBNF.

We will now explain a number of features of systems of synchronous-network acceptors going through this definition. A system of synchronous-network acceptors is specified by its networks, the projections between these networks, a declaration of a number of networks as input networks, and a number of parameters that apply to all networks.

Each network is specified by a name, the learning mode and the evolution mode. Suppose the threshold is low. If the evolution mode of a network is OR and there are several networks that project onto it, each of those networks can activate the neurons in the network. If the evolution mode is AND, all projecting networks need to provide excitatory input onto a neuron in order to activate it. The evolution model of a network may be changed after learning. If the learning mode is AND, and there are several networks that project onto a network that learns in AND mode, then the competition is across all projecting networks. If the learning mode is OR, competition is confined to each individual input network.

Each connection; that is, matrix of external input connections, is specified by the names of a presynaptic and a postsynaptic network, the name of an anatomy matrix, and whether the effect on the postsynaptic neurons is excitatory or inhibitory. The anatomy matrix M has only "1" or "0" components. If entry M_{ij} is 0, then the synapse from neuron i to neuron j will be zero. Otherwise its values depends on the function *init*, which is specified by the parameters W and H. The theory of synchronous-network acceptors does not mention inhibitory projections explicitly. However, inhibitory input enters into the neural dynamics via the firing probability, which it reduces, so it will cause no problems. Modulating connections can be specified between arbitrary networks, with the exception of input networks. Their anatomy matrix is the unit matrix, and they are always excitatory.

Input networks are specified by their name, the name of a learning set, and the name of an input set. Input networks cannot be projected on, and have no temporal synapses. Learning sets and test sets are implemented as files. In the context of systems of synchronous-network acceptors the learnset consist no longer of pairs of inputs, the modulator is just another network in the system. Redefining learning sets, we define the collection learning sets for systems of synchronous-network acceptors as follows.

Definition 5.2
A *collection of learning sets for systems of synchronous-network acceptors* is denoted by $\mathcal{L}_1, \ldots, \mathcal{L}_n$ is , where:

$$\mathcal{L}_j \subseteq \{v \mid v \in \mathrm{SUP}_i^+\}.$$

is a learning set. Its ordered representation is a vector

$$\mathcal{L}_j = \left(v_1^j, v_2^j, \ldots, v_m^j\right)^T,$$

where $m = |\mathcal{L}_j|$. Furthermore,

$$\forall j : |\mathcal{L}_k| = |\mathcal{L}_j|,$$

and

$$\forall l \forall j \forall k : |v_l^j| = |v_l^k|.$$

□

The parameters need little comment, only that LambdaE is λ^{ext}, LambdaT is λ^{ext}, and that Rho is ρ. The parameters are the same for all networks. This has the theoretical advantage that a system of networks can be regarded as a two layer network model consisting of one input layer, made up of the input networks, and a recurrent layer with long and short range recurrent synapses; that is, a synchronous-network acceptor.

What is the relation between systems of synchronous-network acceptors and little linguistic creatures? Remember that we would call a system a little linguistic creature if it is autonomous and adaptive, and therefore closed. That is, if it evaluates its inputs in the context of its state, that it acts in order to optimize a relation defined on its system parameters, and gets its internal structure from an adaptation process that took place under pressure of the optimization of this relation between the system parameters. Moreover, there have to sensors that assign values to the system parameters.

Systems of synchronous-network acceptors are not themselves little linguistic creatures. But they carry all requirements in them. Each network in a system is adaptive, in an unsupervised way, so the system can get organized in virtue of the peripheral inputs alone. Each network that is not an input network is a nondeterministic finite-state acceptor, hence its interpretation of input depends on the state it is in. Consequently, a system of these networks will do so also. The peripheral networks provide input information that has an effect on all connected networks. Every network in the system can be interpreted as a system parameter, and the connection pattern defines a relation between all networks. However, there is no clear sense in which an arbitrary system of synchronous-network acceptors optimizes the relation defined by the connection pattern. It is also unclear that any system undergoes an adaptation process controlled by this optimization tendency. So, the construction of a system of synchronous-network acceptors as a little linguistic creature requires an additional design effort.

6 A Little Linguistic Creature

The experiment described in the following subsections is a computer simulation of a simple little linguistic creature according to the theoretical position toward linguistics taken in the previous sections. The experiment serves to demonstrate that the theoretical position can be used for effective practical applications in manipulating natural language.

6.1 Goal

Little linguistic creatures exist. To demonstrate this is the major goal of the experiment. The theoretical stance is that the close relation between the artificial stochastic neural architecture and the architecture of the brain is a sufficient environment for developing natural language manipulating creatures. No demands upon complexity are made within the theory so the proposed model is rather simple, but sufficient to demonstrate this stance.

6.2 Set up

An ensemble of networks modeling a little linguistic creature has to be defined according to the specifications described in section 5. First we will define the architecture of the network on an abstract level and give a formal account of verifying the correctness of this structure. The second part will be dedicated to the input data; the relation of these data with the phenomena perceived by human language users is very important for specifying a little linguistic creature. The last part then will define the parameter setting for the model.

Architecture

The figure below is an abstraction of the architecture of this specific model, where just the composing networks and their connections are

shown. Accepting networks are marked with tuples < *type, type* >, respectively stating its behavior during learning and evolution. For a formal account of these types see section 5 Large

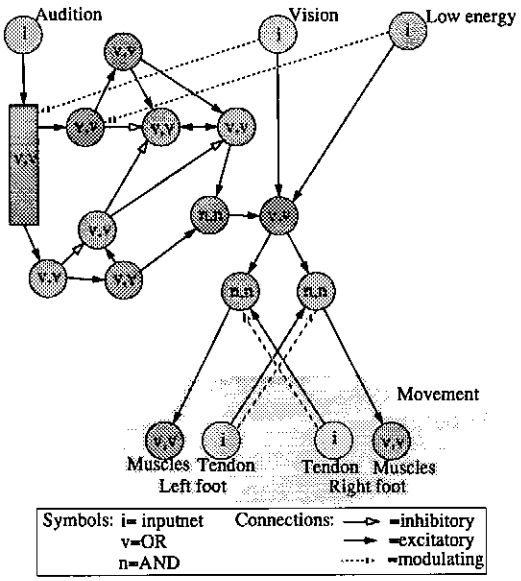


Figure 2: *Architecture of the little linguistic creature.*

rectangles correspond to networks connected in a divergent relation with the projecting network, this relation will be explained below. The relation of the little linguistic creature with the outside world exists by means of the input and output nets. Input nets are defined as such and receive data from the environment, represented in files containing data for the various modes of input, while output nets exist only in a metaphoric sense: any network can be chosen to represent output. A description of the data representing the outside world can be found in the section concerning input below.

Divergence and Convergence

Connections from small networks to larger networks can be achieved by means of convergence or divergence. These larger networks then consist of a set of nets, connected in parallel to the smaller network. The principles of convergence and divergence make it possible to connect nets of arbitrary size, these principles however are purely practical: connections between nets of different size are not accounted for in the theory. Typically a subset of the neurons of the projecting network is connected to a subset of neurons of the set of accepting networks or vice versa. The precise connection matrices are defined by means of the anatomy matrices described in section 5. The relations can be, for both convergence and divergence, many-to-one, one-to-many, many-to-many and one-to-one.

Verification of the Architecture

Abstracting from the specific pattern of activity of a network in the evolution phase, as in the previous subsection, it can be described as a system with two possible states: active or inactive. Complex ensembles of networks can be verified for their distribution of activity on this abstract level for subsequent time steps **t**, when the following requirements are fulfilled:

- A network is defined as *(netName, mode)*, where *netName* is a unique name and *mode* is in {**i**,**n**,**v**}. (*mode* refers to input, AND and OR, as described in 5)

- Connections are defined as *(netFrom, netTo, sign)*, where *netFrom* and *netTo* are net names and *sign* is in {-1,1}.

The activity **a** of net (N_i, T), where T is in {**n**,**v**}, on **t** can now be calculated in the following ways:

$\mathbf{a_t}((\mathbf{N_i}, \mathbf{n})) = 1$ iff
$((\forall \mathbf{N_j}(\mathbf{N_j}, \mathbf{N_i}, 1) \rightarrow \mathbf{a_{t-1}}(\mathbf{N_j}) = 1) \wedge$
$(\forall \mathbf{N_j}(\mathbf{N_j}, \mathbf{N_i}, -1) \rightarrow \mathbf{a_{t-1}}(\mathbf{N_j}) = 0) \wedge$
$(\exists \mathbf{N_j}((\mathbf{N_j}, \mathbf{N_i}, 1) \wedge \mathbf{a_{t-1}}(\mathbf{N_j}) = 1)))$
else $\mathbf{a_t}((\mathbf{N_i}, \mathbf{n})) = 0$.

$\mathbf{a_t}((\mathbf{N_i}, \mathbf{v})) = 1$ iff
$\sum\{\mathbf{V} \mid ((\mathbf{N_j}, \mathbf{N_i}, \mathbf{V}) \wedge (\mathbf{a_{t-1}}(\mathbf{N_j}) = 1))\} \geq 1$
else $\mathbf{a_t}((\mathbf{N_i}, \mathbf{v})) = 0$.

The activity for networks (N_i,**i**) is predetermined for every **t** that is to be calculated. Activity in networks of type **n** or **v** on $\mathbf{t} = 0$ is always 0. The input for these abstracted networks is equivalent to the input sets for the actual model during evolution and thus to perceptual data for a little linguistic creature.
The verification tool has been implemented in Prolog. When the specifications of the ensemble of nets concerning connections and modes in evolution are accounted for in the abstraction, the direct mapping of Prolog to first-order-logic yields a proof for the calculated activity of composing networks.

Input

The relation of a little linguistic creature with the world outside, which is represented by files for a computer simulation, exists by means of the perceptual apparatus of the creature, modeled by input nets. For this little linguistic creature vision, audition, motivation and motor-sensory information have been chosen as perceptual modes. The choice for vision and audition is made on behalf of the perception of language in the form of speech where vision provides the creature with appropriate context for this perception during learning: sounds are associated with specific visual stimuli thus creating an ability to segment the speech flow into meaningful segments. The last two modes of perception may seem less obvious choices. For a being capable of moving through its environment it is necessary to have a constant information flow concerning its position, in terms of muscle activity and position. Not knowing what the position of the body, or parts of the body is, means not being able to perform the planned movement in the correct way. Several experiments with patients unable to perceive this information (due to lesions along the afferent pathway carrying this information) have shown that performance is strongly impaired by this inability [13]. These sources of information, in the model the tendon organs, as well as motivational information, must be input nets because the direct cause of reaction of these systems is not neuron based. Tendon organs measure the rate of tension of a muscle and transfer this information to the brain, while the motivational system of this little linguistic creature consists of information about the energy level of the creature, that is, whether it is hungry or not, which is caused by measurements of specific chemoreceptors.

Hunger is represented by an input net representing chemoreceptors for signaling a shortage in energy supplies. This net can be active or inactive, no specific patterns are needed. The tendon organs are presented two different, opposite patterns, representing the two possible states of each leg, namely front or backward position. The two tendon organs are activated with opposite patterns each update, while one leg always must be front, the other backward. Input is presented in a continuous stream: no input means physical damage to this system.

Vision for this little linguistic creature is very primitive and makes discriminations in a gradual way between light and dark. Somewhere between these extremes lies the threshold for activity. The amount of light is simply represented by the amount of active neurons within the visual input net. A more sophisticated approach is needed for the perception of language in terms of sound.

Speech is perceived by humans as a complex longitudinal waveform. This pattern of fluctuations in air pressure is transformed by the intricate system of the ear into electrical pulses. The part of the ear where this transformation is conducted, the *cochlea* applies a Fourier transformation to the complex wave, so the electrical output, a cross subsection of the auditory nerve on a certain moment, can be seen as a tonotopic map of the different frequencies composing the complex wave on this moment. The amplitudes of this different waves are represented by the firing rate of the neurons representing the specific frequencies.

The human ear is typically capable of perceiving sound in the range of 20 to 20.000 Hz, for speech however the most important and sufficient part for making the correct discriminations between sounds, is in the range of 50 to 3000 Hz. Every sound has its specific frequencies in this range where the amplitude is highest. For speech these so called *formants* are determined by the shape of the vocal tract and are, for vowels, limited to three in the range 50 to 3000 Hz. These formants differ for every person, but an average can be approximated for male and female speakers. For the same speaker these formants are rather constant within the vowels. The formants for consonants are much dependent on the preceding and following phones, however, and can not be determined in the same fashion. (Data from [2]) The data used for the model consists of vectors representing the range important for speech where the bits of the vector can be either 0 or 1. A 0 means that the specific frequency is absent, a 1 that it is present in the period represented by the vector. We chose time slices to be rather crude, while the storage capacity of a net is limited. This means that variations in volume can not be accounted for, since specific firing rates of different neurons can not be represented within this scale, but this in accordance with the behavior of the creature: there is no need for volume perception because there is no behavior associated to differences in volume. We also abstracted from the context dependency of consonants for the same reason. This, however, is not a principal difference either because context dependency means that patterns will be more unique and temporal relationship more easily be stored, a task for which the artificial neural architecture is highly

qualified.

Parameters

The parameters were:

Parameters = {
NumNodes = 30
NetDimensions = 1
LambdaE = 40.0
LambdaT = 25.0
Rho = 0.1
W = 1.0
H = 16.0
Epochs = 10
Noise = 0.0
}.

6.3 Procedure

Learning was applied in one step, which is a divergence of the normal method where first the projections are learned, then the temporal associations. These two forms of learning, described in 5, need not necessarily be applied in distinct steps and considering the relative simplicity of our system we chose not to do so. All data were presented to the system in sequences where speech data was a continuous stream of patterns, while activity was presented to the other input nets on the parts of the speech sequence to be associated with certain behavior.

6.4 Results

A description of the results of the experiment is, in a sense, a metaphorical description of the behavior of the simulated little linguistic creature, we will therefore, where this is possible, refer to this level of description rather than to the level of files and computers. The semantics of perceptual stimuli is determined by the behavior of the creature where this behavior is equivalent with the externally observable changes in its state. For a model this is equivalent with changes in activity of nets chosen to represent output, for *this* model those output nets are the muscles. Perceptual information thus only has meaning for a specific creature when it causes behavior of that creature. For this model this means that some information is meaningful only in the right context, like the way hunger only causes a change in state when the creature is not already frightened by light or sound. This specific little linguistic creature is capable of movement as its sole external observable behavior and it does this when confronted with too much light or with a certain fragment of speech or when it is hungry, that is, when its chemoreceptors signal a lack of resources. Movement is ended when the repetition of the fragment ends, when an error occurs in this fragment, when the light disappears or when the resources have been judged sufficient again. The external world for the little linguistic creature is of course limited in time by the user, but in principle the described input can be repeated endlessly. The behavior of this model, apart from speech recognition, can be described by the term *ortho-kinesis*, used in the behavioral sciences. This means that the behavior of the creature depends solely on the amount of stimulation and is not, for example, goal directed [3]. The success of the flight for the source of the stimulus is only determined by the movement as long as this stimulus is present.

6.5 Discussion

The model presented is small whereas the neural network architecture is designed for large networks. The learning algorithm models processes in living neuronal tissue, where typically thousands or even millions neurons are involved. For very limited systems as our proposed little linguistic creature this means for example, that you can not depend on the stochastic properties of the system and have to be very careful in determining the input data, which is not only a tedious effort but also a deviation from the original idea. However, the model proposed is a real little linguistic creature according to the theoretical stance taken in the previous sections. The system is autonomous in that it has system parameters which are assigned values by sensory information. The parameters are strongly related within the structure of the system and learning means autonomically optimizing this relation according to the unsupervised learning process described in section 4. It may be judged a strange fact that models of motor systems in a system of synchronous-network acceptors have temporal synapses: that is, only input networks, and no output networks can be specified. However, we think that sequences of inputs for the model motor systems are too time-varying to give rise to significant temporal synapses.

7 Conclusions

The synchronous-network acceptor and the unsupervised learning algorithm form a plausible, adaptive model of two-layer modules in the nervous system. Systems of such networks have the necessary requirements to form artificial language-users, or little linguistic creatures, but it requires an additional design effort to construct a system of synchronous-network acceptors that is also a little linguistic creature. A first experiment was promising, and validated the main operational principles. However, the small size of the networks in this experiments gave rise to problems so that we could not design the little linguistic creature on a theoretical basis alone.

References

[1] D.J. Amit. *Modeling Brain Function.* Cambridge University Press, Cambridge, MA, USA, 1989.

[2] P.R. Broecke. *Spraak als betekenisvol geluid in 36 thematische hoofdstukken.* Floris, Dordrecht, The Netherlands, 1988.

[3] R. Brown and R.J. Hernstein. *Psycholoy.* Methuen & Co., London, UK, 1975.

[4] N Chomsky. *Syntactic Structures.* Mouton & Co., The Hague, The Netherlands, 1957.

[5] N Chomsky. *Topics in the Theory of Generative Grammar.* Mouton & Co., The Hague, The Netherlands, 1966.

[6] N Chomsky. *Knowledge of language: its nature, origin, and use.* Praeger, New York, 1986.

[7] Patricia Churchland. *Neurophilosophy: toward a unified science of the mind-brain.* MIT Press, Cambridge, MA, USA, 1986.

[8] M. F. J. Drossaers. Neural-network acceptors. Memoranda Informatica 92-36, University of Twente, Enschede, The Netherlands, 1992.

[9] M. F. J. Drossaers. Noisy network-acceptors. Memoranda Informatica 94-63, University of Twente, Enschede, The Netherlands, 1994.

[10] M. F. J. Drossaers. Unsupervised learning in synchronous-network acceptors. Memoranda Informatica (In preparation), University of Twente, Enschede, The Netherlands, 1994.

[11] K. Jensen, N. Wirth, A. Mickel, and J. Miner. *Pascal User Manual and Report: Revised for the ISO Pascal Standard.* Springer, New York, NY, USA, third edition, 1985.

[12] R.R. Kandel and R.D. Hawkins. The biological basis of learning and individuality. *Scientific American,* 267(3):53–60, September 1992.

[13] R.R. Kandel, J.H. Schwartz, and T.M. Jessell, editors. *Principles of Neural Science.* Elsevier, New York, NY, USA, third edition, 1991.

[14] T. McEnery. *Computational Linguistics: a handbook & toolbox for natural language processing.* Sigma, Winslow, UK, 1992.

[15] T. Winograd and F. Flores. *Understanding computers and cognition: a new foundation for design.* Ablex, Norwood, NJ, USA, 1986.

WORD AGENT BASED NATURAL LANGUAGE PROCESSING

Hermann Helbig and Andreas Mertens
FernUniversität Hagen
Praktische Informatik VII/Artificial Intelligence
D-58084 Hagen
Germany
e-mail:
hermann.helbig@fernuni-hagen.de
andreas.mertens@fernuni-hagen.de

ABSTRACT

Natural language processing (NLP) is often based on declaratively represented grammars with emphasis on the competence of an ideal speaker/hearer. In contrast to these broadly used methods, we present a procedural and performance-oriented approach to the analysis of natural language expressions. The method is based on the idea that each word-class can be connected with a functional construct, the so-called word agent, processing its own part of speech. The syntactic-semantic analysis performed by the word-class agents is surveyed by a *Word-Class Agent Machine* (WCAM) used in the bibliographic information retrieval system LINAS at the University of Hagen.

The language processing is organized into four main levels: on the first and second level, elementary and complex semantic constituents are created which have their correspondence in mental kernels built during human language understanding. On the third level, these kernels are used to construct the semantic representation of the propositional kernel of a sentence. During this process, the subcategorization features of verbs and prepositions play the main part. The task of the last level consists in the syntactic analysis and semantic interpretation of modal operators in the broadest sense and of the coordinating or subordinating conjunctions which connect the propositional kernels.

1 INTRODUCTION

Although word-based natural language analysis is not a widely used method today, it has been of growing interest for a couple of years. Some examples of word-based approaches are the *Word Expert Parsing* [Small 81], the *Word-Class Controlled Functional Analysis* [Helbig 86], the *word-oriented parsing* [Eimermacher 86] and the analysis with *word actors* [Bröker, Hahn, Schacht 93].

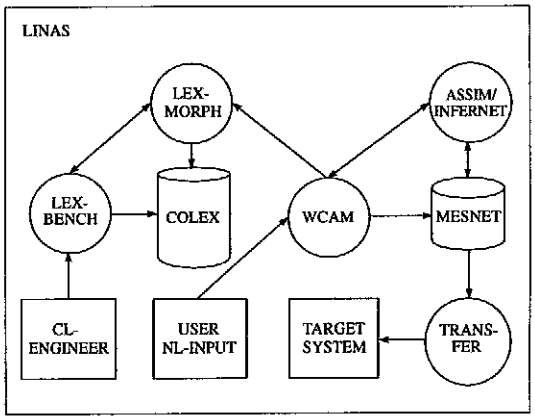


Figure 1: Overview of the NL system LINAS

The theory of word expert parsing (WEP) approaches the understanding of natural language as a distributed process of interacting words. Each word is connected with an expert process which "actively pursues its intended meaning in the context of other word experts and real-world knowledge" [Small 87].

Eimermacher's word-oriented parsing relies upon the word expert paradigm, but distinguishes between word-class experts representing general grammar rules and word experts analyzing the relations between single words [Eimermacher 88].

The *ParseTalk* model is a concurrent, object-oriented parsing system with grammatical knowledge completely integrated into the lexicon. Each word of a sentence the parser is currently working on activates a word actor which

communicates with other initialized word actors and other components of the system [Hahn, Schacht, Bröker 94].

Within the model of word-class controlled functional analysis (WCFA), experts are introduced for each word-class and the grammatical function of a word-class is given by a procedural specification. The characteristic feature of the word-class functions discerning them from other approaches is their partition into two different functional components which are activated at different times of the sentence analysis [Helbig 86]. In this paper, the essential aspects of the word-class agent machine (WCAM), which is a further development of the WCFA, will be presented. For more detailed information of this approach see [Helbig, Mertens 94]. A formal description of the four main levels of the word agent based language processing can be found in [Helbig 94].

2 SYSTEM COMPONENTS

The LINAS project aims at developing a natural language understanding system and, as a practical side effect, at providing a natural language interface to bibliographic databases. An architectural overview of those system components which are relevant to NLP is shown in Figure 1. The following major components are distinguished: an interactive lexicographer's workbench (LEXBENCH), a morphological processor (LEXMORPH), a word-class agent machine (WCAM), a knowledge assimilation and inference component (ASSIM/INFERNET), and an interface module to database systems (TRANSFER). In addition, there are two main knowledge sources: a computer lexicon (COLEX) and a network knowledge base (MESNET).

For each word of a sentence the WCAM is currently working on the morphological analysis (LEXMORPH) is activated. If the current word form has no lexical entry, the corresponding entry (the stem) can be determined by cutting off inflectional syllables. The information included in the basic entry such as stem, grammatical specification and the accompanying word-class agent will be returned to the WCAM.

The aim of the WCAM is to analyse the NL input and to represent it by means of a multilayered extended semantic network (MESNET). The analysis is done by a set of word-class agents (cf. Table 1) and the result produced by the agents is stored in the network knowledge base MESNET. The assimilation of the knowledge is organized by the ASSIM/INFERNET component. In the case of a question to a database management system as target sytem (not to the knowledge base of LINAS itself), the semantic description of the NL expression is translated into the database query language. This is done by the translation and interface module TRANSFER.

word-class agent	characterization
**ART*	articles
**ADJ*	adjectives
**NOM*	nouns
**VB*	verbs
**PREP*	prepositions
**AV*	auxiliary verbs
**IPATTR*	interrogative pronouns (in attributive use)
**IPNOM*	interrogative pronouns (in nominal use)
**ADV*	adverbs
**GRAD*	graduators (adverbs of degree)
**POSSPR*	possessive pronouns
**RELPR*	relative pronouns
**PART*	participles
**COMP*	comparative forms
**NUM*	numerals
**NEG*	negative forms
**CONJ*	conjunctions
**COMMA*	commas
**STOP*	full stop

Table 1: Some examples of word-class agents

The lexical entries of the computer lexicon (COLEX) are organized as feature structures. For typical entries see Figures 3 and 4. LEXBENCH is an interactive workbench supporting the process of creating and checking lexical entries.

3 THE WORD-CLASS AGENT MACHINE (WCAM)

As well as in the theory of WEP, the basic idea of the WCAM is that words play the main part in NLP. In contrast to WEP, which provides one *expert* for each word, we present a word-class ori-

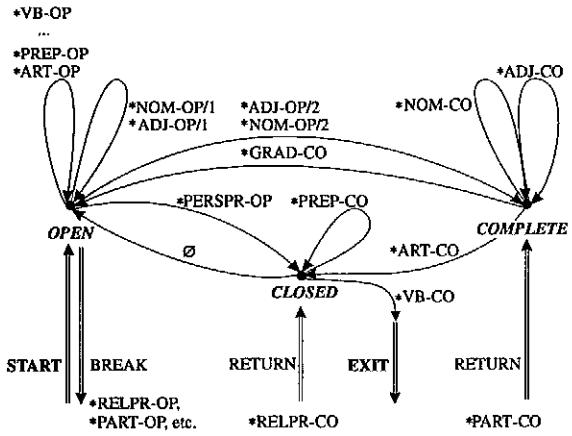


Figure 2: WCAM transition diagram

ented approach. For each word-class its grammatical function is defined by a procedural specification, the so-called word-class agent, and each element of a word-class will be associated with the corresponding procedure. In LINAS more than 35 word-class agents are distinguished. Some of them are listed in Table 1. In addition to agents such as *ART (article), *ADJ (adjective), *NOM (noun) and *VB (verb), which are suggestive of the word-classes in traditional grammars, there are other agents such as *IPATTR and *IPNOM, which represent interrogative pronouns in attributive and nominal use, respectively. Furthermore, there are special agents responsible for the grammatical function of punctuation marks such as *COMMA (comma) and *STOP (full stop).

The word-class agents are divided into an OPEN-act and a COMPLETE-act.[1] This division corresponds to the two major activity modes of a word during NLP: on the one hand, a word opens certain valencies which may be satisfied later on by other words or constituents of the current sentence. On the other hand, the valencies opened in the first activity mode must be satisfied by suitable candidates. These are the words or constituents which have been already included in the analysis and which saturate the opened valencies. In Artificial Intelligence (AI) the specifying of conditions (slots) which must be filled by specific data (filler) satisfying these conditions is known as the *slot-filler-mechanism* of frames.

[1]This subdivision, however, is not made in each case. There are some word-class agents such as *ADV (adverb) and *IPNOM (the interrogative pronoun in nominal use) without a COMPLETE-act. The elements of these word-classes do not open valencies. Their part is to saturate the valencies of other words in the current sentence.

The two activity modes of a word are represented by different procedural specifications: the OPEN-act and the COMPLETE-act of a word-class agent. According to the special tasks of OPEN-act and COMPLETE-act, the two acts are triggered by the WCAM at different moments in the analysis.[2] The division into an OPEN-act and a COMPLETE-act is one of the main differences to the other word-oriented approaches mentioned in Section 1. Another characteristic feature of the WCAM approach is the integration of semantic interpretation processes into the COMPLETE-acts of the word-class agents.

In Figure 2, the WCAM transition diagram is shown. The three states of the WCAM given in this Figure have the following meaning:

- *OPEN* - if a word is analysed by the WCAM, first the morphological component is activated in order to generate the feature structure of the corresponding lexical entry. Then this structure is passed on to the WCAM. If the current word opens valencies, the semantic description of the word gets the marker OP (open) before it is stored in the working memory. Otherwise, if the word does not open valencies (for instance, in the case of adverbs, proper names, etc.), the description of the word is marked by CL (closed).
- *COMPLETE* - in this state the COMPLETE-act of a word-class agent is activated in order to satisfy the valencies being opened by the corresponding OPEN-act. Thus, in this state the decision that a group of words forms a particular constituent is made. For example, coming to the end of a nominal phrase (NP), the *ART-CO-act is triggered in order to saturate the valencies being opened before in the *ART-OP-act. As a result, an NP-constituent will be constructed and its complex syntactic-semantic structure will be stored in the working memory.
- *CLOSED* - this state of the WCAM indicates the fact that the current constituent is completely analysed with all its valencies being saturated. The semantic structure of this constituent is marked by CL, now being permitted to satisfy the valencies of other constituents. At the end of a sentence, indicated by a triggered *VB-CO-act, the valencies of

[2]The OPEN-act and the COMPLETE-act of a word-class agent are marked by extending the name of the agent with one of the two endings -OP or -CO, respectively (cf. the WCAM transition diagramm in Figure 2).

the verb will be satisfied. Otherwise, if there are no COMPLETE-acts being left to perform, the WCAM immediately changes into the OPEN-state (cf. the Ø-transition in Figure 2).

The word-class agents are triggered and inspected by the WCAM. Depending on the word the WCAM is currently analyzing, the corresponding word-class agent is activated. For example, consider the following sentence consisting of the articles *der* and *ein*, the adjective *jung*, the nouns *Mann* and *Buch*, and the verb *schreiben*:

- *Der junge Mann schrieb ein Buch.*
- *The young man wrote a book.*

The initial state of the WCAM is OPEN (cf. Figure 2). When the first word *der* of the given sentence is included in the analysis, the WCAM activates the OPEN-act of the word-class agent *ART, which is called *ART-OP. The system remains in the OPEN-state. Analyzing the next word, i.e. the adjective *jung*, the WCAM triggers the *ADJ-OP-act. If there is no graduator[3] between the article and the adjective, the *ADJ-OP/1-act is activated and the WCAM remains in the OPEN-state because at this moment not any valencies could be satisfied.

Analyzing the noun *Mann*, the *NOM-OP/2-act[4] is activated and the WCAM changes to the COMPLETE-state. Now the valencies opened by the article and the adjective can be satisfied and the COMPLETE-acts of the adjective and the article are activated successively. In the course of these COMPLETE-acts, the agreement with respect to gender, case and number of the analysed word forms is checked. As a result of the *ART-CO-act, the WCAM changes to the state CLOSED. In this state, the NP is completely analysed and a semantic representation of the whole phrase has been created and stored.

If an OPEN-act of a word-class agent is activated, this information is stored in the central working memory (CWM) which is organized as a stack. Thus, as analysis proceeds, the corresponding COMPLETE-acts will be triggered just the other way round.

Because there are no COMPLETE-acts being left to perform (cf. the description of the WCAM-states given above), the WCAM changes into the OPEN-state. Next the verb *schreiben* is read and its OPEN-act will be activated. The following NP is processed by analogy to the first one. The processing of the NP results in the CLOSED-state of the WCAM. At this stage, the *VB-CO-act is triggered and the items of the case frame on the syntactic and the semantic level are allocated to the valencies opened by the verb (cf. Figure 4). Expecting an agent (AGT) and an object (OBJ), among others, the valencies of the verb are satisfied by the complex semantic structures of the first and the second NP, playing the AGT- and OBJ-role, respectively. On the syntactic level, the agreement in respect of person, number and case for the AGT-role and with reference to case for the OBJ-role is checked. On the semantic level, for both roles the agreement in regard of semantic sorts is verified.

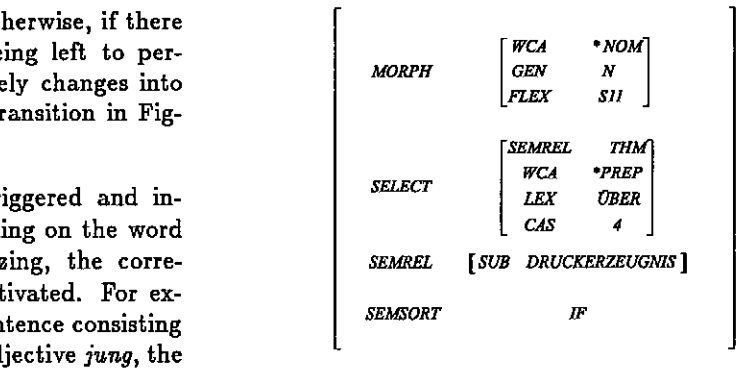


Figure 3: Lexical entry of *Buch* (*book*)

4 GENERAL ASPECTS

4.1 DISAMBIGUATION PRINCIPLES

Although a variety of approaches and a lot of systems have been developed, a general natural language system is still out of sight. This desideratum is generally explained with the enormous complexity of natural language (see, e.g., [Stürmer 93]). One of the main reasons for this complexity lies in the ambiguity of natural language expressions. Therefore, NLP must be based on adequate stragegies which support the disambiguation processes.

In order to explain the human preferences in natural language understanding, researchers in the fields of computational lin-

[3] For instance, if the article is followed by an adverb of degree such as *sehr* (*very*), the *ADJ-OP/2-act is triggered and the WCAM changes to the COMPLETE-state.

[4] *NOM-OP/1 is triggered in the case of apposition (e.g., *Peter der Große*, *Mrs Jane Smith*, *the Oxford English Dictionary*, *Archimedes' Law*).

guistics (CL) and psycholinguistics have suggested several principles such as *Minimal Attachment*, *Right Association*, *Head Attachment*, *Lexical Preferences*, etc. Some of them have been confirmed by psycholinguistic experiments (see, e.g., [Hemforth et al. 92] or [Hemforth, Konieczny, Scheepers 94]).

In the word agent based language processing of the WCAM the following general disambiguation principles are implemented:

- *Completion* - semantic constituents, which have their correspondence in mental kernels built during human language processing, are created and completed as soon as possible.
- *Valency* - in the broadest sense, the valencies of words affect the attachment preferences. The principles of *Compatibility* and *Incompatibility* handle these preferences by making use of the subcategorization feature SELECT and the compatibility feature COMPAT in the lexicon (cf. Figure 4). Compatible feature values of two constituents permit the subordination of the second one and, on the other hand, feature values which are not compatible inhibit the attachment of constituents. Furthermore, the features are processed in an descending order (principle of *Priority*). Obligatory valencies of a word have priority over optional ones and the latter have priority over adverbial qualifications.
- *Right Association* - a new constituent tends to be interpreted as being part of the preceding constituent.
- *Preference of Reading* - this principle states that, in the case of polysemous words, one of the alternative meanings tends to be preferred. The preferred reading must be marked in the lexicon.

A detailed description of the disambiguation principles used in the WCAM and illustrative examples are given in [Helbig, Mertens 94]. The co-operation of the strategies for disambiguation and the lexical information, which plays an important part in resolving structural ambiguities, is presented in [Helbig, Mertens, Schulz 94].

4.2 LEXICAL INFORMATION

A lexical entry in LINAS consists of morphological, syntactic and semantic information. The feature MORPH has as its value a feature structure which itself comprises such features as WCA, GEN and FLEX. WCA denotes the corresponding word-class agent such as *VB (verb), *NOM (noun) and *PREP (preposition). The feature FLEX contains the description of the morphosyntactic properties of a word, including type of declension (for articles, adjectives and nouns) and conjugation (for verbs), respectively. The types of declension for words in the nominal group and the types of conjugation for verbs are devided into several groups. For instance, the group S02 represents the nouns with the ending *-s* in genitive and in all plural forms (e.g., *Auto - car*). In the case of words of the nominal group, the feature GEN designates the gender. An example of a noun entry is shown in Figure 3.

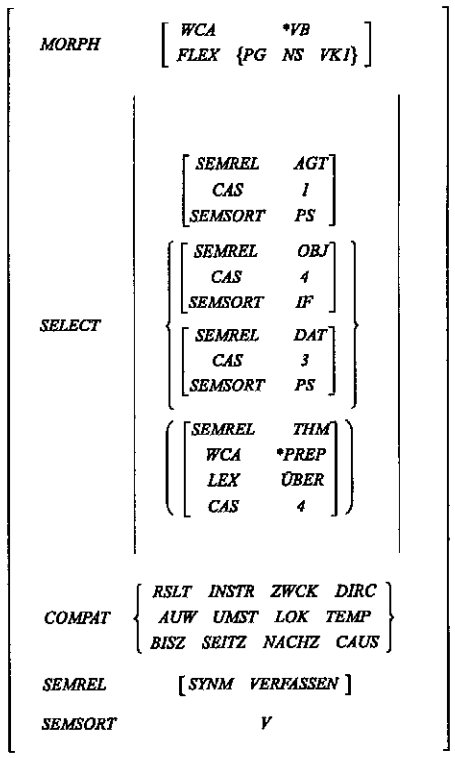


Figure 4: Lexical entry of *schreiben (to write)*

The values of the feature SELECT are obligatory and optional valencies of words. For verbs it represents the case frame on the syntactic and the semantic level. For instance, the German verb *schreiben (to write)* requires an agent (AGT) and at least one of the two roles object (OBJ) or a dative role (DAT). Besides, it allows for an optional thematic role.[5] The shortened lexical entry of this verb is given in Figure 4. In addition, the

[5] Another possible role - BENF (beneficiary) - is omitted here for the sake of brevity.

feature structure COMPAT expresses the compatibility between verbs and adverbial qualifications.

The feature structure SEMREL of a lexical entry contains the semantic relations to other words in the lexicon. For instance, if two words have the same meaning, they are in the relation of synonymy (e.g., *schreiben - to write* and *verfassen - to pen*). Some other examples of semantic relations included in the lexicon are SUB (hyponymy), PARS (part-whole-relation), ASSOC (association), etc. The information about semantic relations plays a crucial part in lexical disambiguation.

relation	characterization
AGT	agent
ASSOC	association
CAUS	causality
DAT	dative role
DIRC	direction
EQU	equality
INSTR	instrument
LOC	location
OBJ	object involved
ORIGM	original material
PARS	part-whole-relation
POSS	possessive relation
PROP	property, attribute
RSLT	result
SUB	subordination, hyponymy
SUBA	subordination of events
TEMP	temporal relation
THM	thematic role

Table 2: Some examples of semantic relations

The feature SEMSORT has as its value the semantic sort of the lexical entry such as *V* - Vorgang (event), *PS* - Person (person) or *L* - Lokation (location). In LINAS the classification of entities is divided into the following three levels:

- *epistemic-ontological level* - entities are divided into sorts which are philosophically motivated and which are used in defining the algebraic structure of the knowledge representation apparatus.
- *socio-cultural level* - entities are specified by a given set of features allowing for a cross classification and being motivated by the importance of these categories for the description of human activities.
- *linguistic subtlety level* - entities are characterized by connecting them to parts of a semantic network. This information is used for linguistic fine tuning of selection restrictions.

A detailed discussion of the underlying three-level classification and illustrative examples can be found in [Helbig, Herold, Schulz 94].

4.3 KNOWLEDGE REPRESENTATION

The syntactic and semantic analysis aims at representing NL input by means of a multi-layered extended semantic network (MESNET) and at integrating the results of the interpretation process in a network knowledge base.

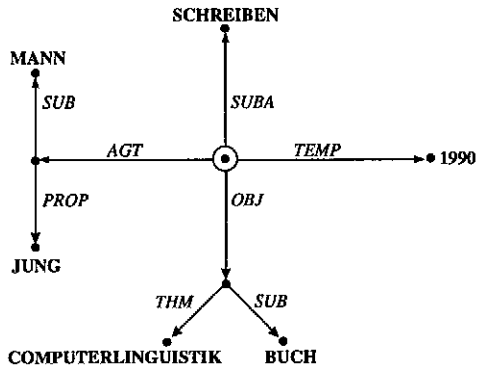


Figure 5: Semantic representation of a sentence

In LINAS the language interpretation process involves selecting the appropriate semantic relations which encode the connections between the constituents of a given NL sentence, and using them to construct a semantic network that represents the complete information of the sentence. Some examples of semantic relations and their characterizations are listed in Table 2. The semantic representation of the following example is given in Figure 5.

- *1990 schrieb der junge Mann ein Buch über Computerlinguistik.*
- *In 1990 the young man wrote a book about computational linguistics.*

The knowledge representation method used in

LINAS extends the basic approach of semantic networks (SN) by accounting for the information such as the partitioning of the SN into shells and layers, the distinction between different levels and dimensions of the underlying classification, etc. The main characteristics of this new approach are the following:

- *stratification* - the SN is organized in different layers, the most prominent of them being the intensional and the preextensional layers.
- *semantic sorts* - the nodes of the SN are characterized by semantic sorts which are used in defining the network's algebraic structure. In addition, the nodes can be annotated with the semantic features and the selectional categories of the underlying three-level classification (cf. Section 4.2).
- *semantic relations and functions* - the nodes of the SN are linked to other nodes of the net by means of semantic relations and functions, which represent the semantic connections between the constituents of the sentence.
- *semantic shells* - in order to represent a complex semantic structure, a set of nodes and semantic relations of a SN can be combined in a semantic shell.
- *semantic dimensions* - the semantic dimensions are formed by a bundle of contrastive pairs: e.g., determinate vs. indeterminate reference of concepts, virtual vs. real concepts, individual vs. generic concepts, and collective vs. non-collective concepts. Furthermore, the nodes of the SN can be specified by intensional and preextensional indications of quantities.

5 THE FOUR MAIN LEVELS OF PROCESSING

The language processing of the WCAM is organized into four main levels (cf. Figure 6):

- on the *first level*, elementary semantic constituents are created which have their correspondence in mental kernels built during human language understanding;
- on the *second level*, complex semantic constituents are constructed by connecting the elementary constituents;
- on the *third level*, the constituents created in the first and second level are used to construct elementary propositions of a given sentence;
- in order to give a semantic representation of the whole sentence, on the *fourth level* a complex proposition is built up by connecting the elementary propositions.

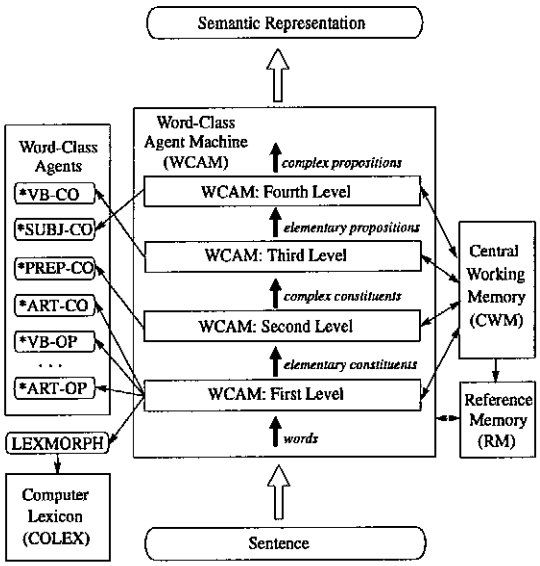


Figure 6: Overview of the WCAM levels

The levels of the WCAM will be illustrated by analyzing the following variation of the example which has been introduced in Section 4.3:

- *Der junge Mann schrieb ein Buch über Computerlinguistik.*
- *The young man wrote a book about computational linguistics.*

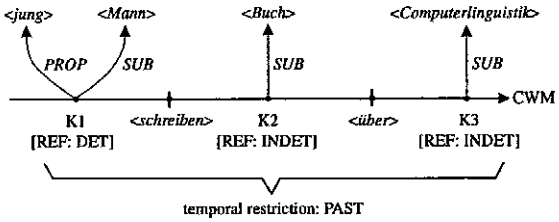


Figure 7: The CWM structure on the first level

On the first level, for each word of a sentence the morphological analysis is activated in order to read the feature values of the corresponding lexical entries. If the whole sentence has been analysed on the first level (the activation of the word-class agents and the changing of the WCAM states are described in Section 3), three elementary semantic constituents have been created which have their correspondence in *mental*

kernels built during human language understanding. In Figure 7, which shows the structure of the CWM at the end of the first level, these kernels are labelled K1 to K3.[6]

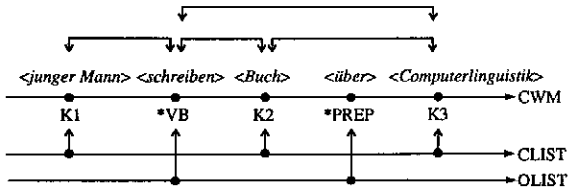


Figure 8: OList and CList on the second level

The task of the WCAM on the second level consists in constructing complex constituents by connecting the elemenary constituents determined on the first level. A difficult question concerns whether the elementary semantic constituents should be subordinated to a preceding elementary constituent or to the verb.

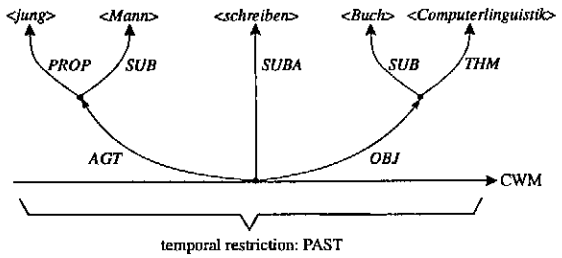


Figure 9: The structure of the CWM after creating an elementary proposition on the third level

The elementary constituents K1 to K3 are marked by CL (closed) because they are permitted to satisfy the valencies of other constituents of the sentence. The verb *schreiben* and the preposition *über* are marked by OP (open) because these words open valencies (cf. Section 3). The constituents marked by CL are stored in the CList and the words marked by OP in the OList. Given that the elements of the CList and the OList are arranged in the order of the current sentence, a useful technique that takes advantage of the structure the two lists is to analyse the sentence in reverse order. In our example, the kernel K3 may be subordinated to the kernel K2 or to the verb (see Figure 8). The lexical entries of the noun *Buch* and the verb *schreiben* shown in Figures 3 and 4 allow both subordinations. Following the disambiguation principle of *Right Association*

[6]Determinate vs. indeterminate reference of concepts is marked by DET and INDET, respectively. This contrastive pair forms one of the semantic dimensions mentioned in Section 4.3.

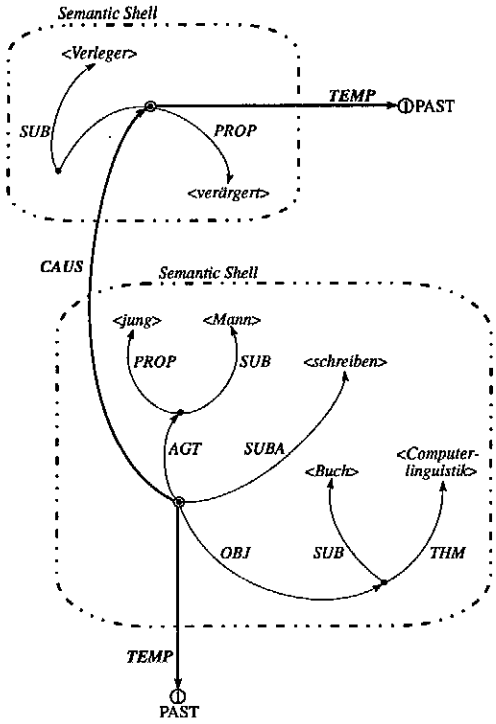


Figure 10: Constructing a complex proposition

(see Section 4.1), K3 will be attached to the preceding kernel K2. Thus, a complex kernel is built up by connecting the kernels K2 and K3.

On the third level, the kernels determined on the levels before are subordinated to the verb in order to saturate its open valencies. The valencies are processed in a descending order. First the obligatory valencies of the verb and then the optional valencies are satisfied. Finally the adverbial qualifications are subordinated to the verb. The subordination of the kernels is done by making use of the subcategorization feature SELECT for the obligatory and optional valencies and the compatibility feature COMPAT for the adverbial qualifications (cf. Figure 4). The case frame of the verb *schreiben* requires an AGT and at least one of the two roles OBJ or DAT. In our example, the AGT-role and the OBJ-role are expressed by the elementary kernel <*junger Mann*> and the complex kernel <*Buch über Computerlinguistik*>, respectively. At this state, the semantic representation of the given sentence is stored in the CWM (cf. Figure 9).

In addition, the case frame of *schreiben* allows for an optional thematic role (THM) which may be satisfied by the kernel K3. This saturation is blocked, however, for the reasons mentioned above.

On the fourth level, the elementary proposition which has been determined in the preceding level may be connected to another elementary proposition. This process of the WCAM will be illustrated by analyzing another variation of the example introduced before:

- *Der Verleger war verärgert, weil der junge Mann ein Buch über Computerlinguistik geschrieben hatte.*
- *The publisher was angry because the young man had written a book about computational linguistics.*

Now a clause of reason being added to the example, two elementary propositions can be determined by the WCAM on the third level and, in each case, the nodes and relations of the propositions are combined in separate semantic shells (cf. Figure 10).

On the fourth level, first the temporal information will be added to the SN and then the main task of the WCAM consists in analysing the subordinating conjunction *weil*. In order to guarantee the interpretation of the conjunction, the COMPLETE-act of the word-class agent *CONJ is activated and, as a result of this process, the two propositions are combined by the corresponding semantic relation *CAUS*.

6 CONCLUSIONS

In this paper we summarized the fundamentals of our approach to word agent based NLP. We further discussed the syntactic-semantic analysis of the WCAM and the different levels of language processing. In addition, we introduced the system components which are directly connected to the WCAM.

The word based approach presented in this paper can be characterized briefly by the following essential properties:

- the word and its combinatory potential plays the central part in natural language understanding;
- each word-class is connected to a corresponding word-class agent representing its grammatical function;
- integration of syntactic and semantic interpretation;
- the word-class agents are divided into two different acts corresponding to the opening and satisfying of valencies;
- the word-class agents are triggered and inspected by the word-class agent machine;
- the language processing is organized in four main levels: elementary and complex constituents, elementary and complex propositions;
- the disambiguation processes are supported by a group of strategies and, in addition, by special information in the lexicon;
- the result of the analysis is represented by means of a multilayered extended semantic network.

Our approach is practically applied in a natural language understanding system and the components described in the previous sections are implemented and successfully used in the bibliographic information retrieval system LINAS at the University of Hagen.

REFERENCES

[Bröker, Hahn, Schacht 93] Bröker, N., Hahn, U., Schacht, S.: Ein Plädoyer für Performanzgrammatiken. In: Deutsche Gesellschaft für Sprachwissenschaft, Sektion Computerlinguistik: Deklarative und prozedurale Aspekte der Sprachverarbeitung, Tagungsband der 4. Fachtagung. Universität Hamburg 1993, pages 6-11

[Eimermacher 86] Eimermacher, M.: Wortorientiertes Parsing mit erweiterter Chart-Repräsentation. In: Rollinger, C.-R., Horn, W. (eds.): GWAI-86 und 2. Österreichische Artificial-Intelligence-Tagung, Ottenstein/Niederösterreich, September 22-26, 1986. Berlin etc.: Springer 1986, pages 131-142

[Eimermacher 88] Eimermacher, M.: Wortorientiertes Parsen. Dissertation. Technische Universität Berlin, Fachbereich Informatik 1988

[Görz 92] Görz, G. (ed.): KONVENS 92 - 1. Konferenz *Verarbeitung natürlicher Sprache*, Nürnberg, 7.-9. Oktober 1992. Berlin etc.: Springer-Verlag 1992

[Hahn, Schacht, Bröker 94] Hahn, U., Schacht, S., Bröker, N.: Concurrent, Object-Oriented Natural Language Parsing: The *ParseTalk* Model. CLIF-Report 9/94, Universität Freiburg 1994

[Helbig 86] Helbig, H.: Syntactic-semantic analysis of natural language by a new word-class controlled functional analysis (WCFA). Bratislava: Computers and AI 5 (1986) 1, pages 53-59

[Helbig, Mertens 94] Helbig, H., Mertens, A.: Der Einsatz von Wortklassenagenten für die automatische Sprachverarbeitung. Teil I - Überblick über das Gesamtsystem. Informatik Berichte 158. FernUniversität Hagen 1994

[Helbig 94] Helbig, H.: Der Einsatz von Wortklassenagenten für die automatische Sprachverarbeitung. Teil II - Die vier Verarbeitungsstufen. Informatik Berichte 159. FernUniversität Hagen 1994

[Helbig, Mertens, Schulz 94] Helbig, H., Mertens, A., Schulz, M.: Die Rolle des Lexikons bei der Disambiguierung. In: [Trost 94], pages 151-160

[Helbig, Herold, Schulz 94] Helbig, H., Herold, C., Schulz, M.: A Three Level Classification of Entities for Knowledge Representation Systems. Informatik Berichte 162. FernUniversität Hagen 1994

[Hemforth et al. 92] Hemforth, B., Konieczny L., Scheepers, C., Strube, G.: SOUL-Processing: Semantik-orientierte Prinzipien menschlicher Sprachverarbeitung. In: [Görz 92], pages 198-208

[Hemforth, Konieczny, Scheepers 94] Hemforth, B., Konieczny L., Scheepers, C.: Principle-based or Probabalistic Approaches to Human Parsing: How Universal is the Human Language Processor? In: [Trost 94], pages 161-170

[Small 81] Small, S.: Viewing word expert parsing as linguistic theory. Proc. IJCAI-81, Vancouver: 1981, pages 70-76

[Small 87] Small, S. L.: A Distributed Word-Based Approach to Parsing. In: Bolc, L. (ed.): Natural Language Parsing Systems. Berlin etc.: Springer-Verlag 1987, pages 161-201

[Stürmer 93] Stürmer, T.: Semantic-Oriented Chart Parsing with Defaults. In: Sikkel, K., Nijholt, A. (eds.): Natural Language Parsing - Methods and Formalisms, Proceedings of the sixth Twente Workshop on Language Technology. Enschede: Universiteit Twente, Faculteit Informatica 1993, pages 99-106

[Trost 94] Trost, H. (ed.): KONVENS '94 - Verarbeitung natürlicher Sprache, Wien, 28.-30. September 1994. Berlin: Springer-Verlag 1994

PHONEME-LEVEL SPEECH AND NATURAL LANGUAGE INTEGRATION FOR AGGLUTINATIVE LANGUAGES *

Geunbae Lee
Jong-Hyeok Lee
Kyunghee Kim
Department of Computer Science & Engineering
and Postech Information Research Laboratory
Pohang University of Science & Technology
San 31, Hoja-Dong, Pohang, 790-784, Korea
gblee@vision.postech.ac.kr

ABSTRACT

A new tightly coupled speech and natural language integration model is presented for a TDNN-based large vocabulary continuous speech recognition system. Unlike the popular n-best techniques developed for integrating mainly HMM-based speech and natural language systems in word level, which is obviously inadequate for the morphologically complex agglutinative languages, our model constructs a spoken language system based on the phoneme-level integration. The TDNN-CYK spoken language architecture is designed and implemented using the TDNN-based diphone recognition module integrated with the table-driven phonological/morphological co-analysis. Our integration model provides a seamless integration of speech and natural language for connectionist speech recognition systems especially for morphologically complex languages such as Korean. Our experiment results show that the speaker-dependent continuous Eojeol (word) recognition can be integrated with the morphological analysis with over 80% morphological analysis success rate directly from the speech input for the middle-level vocabularies.

*This research was supported in part by a grant from KOSEF (Korean Science and Engineering Foundation). We also thank to Wonll Lee for coding the lexicon and the morphological parser and to professor Hong Jeong for his valuable suggestions for the earlier draft of this paper. An extended version of this paper was submitted to the journal of natural language engineering for a review.

1 INTRODUCTION

A spoken natural language system requires many different levels of knowledge sources including acoustic-phonetic, phonological, morphological, syntactic, and semantic levels. The knowledge sources are grouped and processed in either speech processing models or statistical/symbolic natural language processing models. Since the speech and the natural language communities have conducted almost independent researches, these models were not completely integrated and often biased by neglecting either the acoustic-phonetic or the high-level linguistic information. The spoken language system requires seamless integration of speech signals into the high level language processing components. Recent advances in large vocabulary continuous speech recognition makes an integrated speech and natural language system possible and feasible. In a spoken language architecture, we must consider all the acoustic-

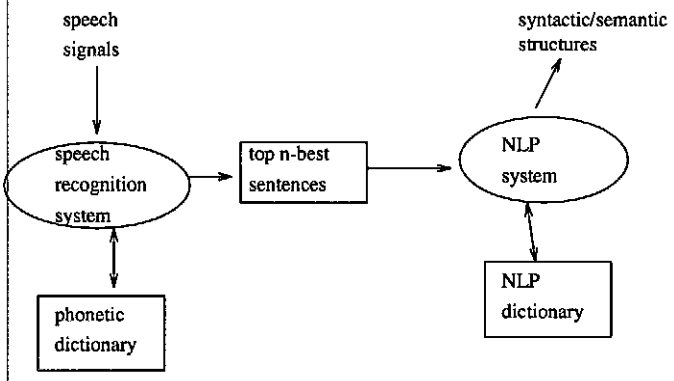


Figure 1: N-best search: current speech and natural language integration method

phonetic and linguistic information equally and choose the most feasible candidates at each acoustic and language processing step.

Current speech and natural language integration mainly relies on the word-level n-best search techniques [1] which are obviously inefficient for morphologically complex agglutinative languages such as Korean. Figure 1 shows the current n-best integration method.

For HMM-based speech recognition systems, the n-best search techniques [1, 2] have been successfully applied to the integration of speech recognition systems into the natural language systems. However, current implementations of the n-best techniques only support integration at the word level (using word sequences or lattice), and mainly used for the integration of existing speech and natural language systems [3, 4]. Also the n-best search is viable only for short sentences since the n grows exponentially with the sentence length (number of words in the sentence). Because the n-best search integrates at the word level, the natural language systems usually support word-level dictionary which seems to be a reasonable assumption in morphologically simple languages such as English. However, most natural language systems which deal with the morphologically complex languages currently use the morpheme-level dictionary for the linguistic generality. For these languages, the dictionary size for large vocabulary continuous spoken language system will grow very fast if we adhere to the full word-level phonetic dictionary because new words can be almost freely generated by concatenating the constituent morphemes (e.g. noun + postposition or verb + verb-endings in Korean). To incorporate the general morpheme-level dictionary into the spoken language system, we must develop a sub-word level integration technique between speech and natural language. The technique is more important in the languages which have very complex morphological structures caused by complex postpositions and verb-endings, such as Korean.

In this paper, we present a new integration architecture of speech and natural language based on the table-driven phonological/morphological co-analysis using the well-known dynamic programming technique [5] and the connectionist diphone spotting technique. Our model integrates a phonological/morphological parsing into a speech recognition, not at a word-level, but at a phoneme-level for a more tightly coupled integrated system. We present a new integration architecture, not for the popular HMM-based systems, but for recently developed connectionist speech recognition systems. Connectionist speech recognition [6] has several advantages compared with the classical symbolic and stochastic modeling. Especially, the time-delay neural network (TDNN) model [7] has been widely used to model the time shift invariance of speech signals. However, the integrated speech and natural language processing models using the TDNN have not been much researched before[1]. In this regard, we present a phoneme-level integration method for large vocabulary connectionist speech recognition model using the TDNN, especially for the morphologically complex agglutinative languages.

[1] One notable exception is the researches by Sawai [8, 9].

2 FEATURES OF SPOKEN KOREAN

Korean, which can be classified into a morphologically agglutinative and syntactically SOV languages, has several unique linguistic features. The followings are morphological and phonological features of spoken Korean for the understanding of our integration method. For the syntactic level features, [10] explains some Korean syntax modeling. In this paper, the Yale romanization is used for representing the Korean phonemes.

1) A Korean word, called Eojeol, consists of more than one morphemes with clear-cut boundaries in between. For example, an Eojeol *pha-il+tul+ul (files [obj])* consists of 3 morphemes:

pha-il (file) + tul (plural suffix) + ul (object case-marker)

2) Korean is a postpositional language with noun-endings, verb-endings, and prefinal verb-endings. These functional morphemes determine the noun's case roles, verb's tenses, modals, and modification relations between Eojeols. For example, in *swu-ceng-ha+yess+ten pha-il (the file that was edited)*, the verb *swu-ceng-ha (edit)* is of past tense and modifies *pha-il (file)* according to the given verb-endings:

swu-ceng-ha (edit) + yess (past tense prefinal verb-ending) + ten (adnominal verb-ending)

3) The unit of pause in a spoken Korean (called Eonjeol) may be different from that in a written Korean (called Eojeol). For example, in speaking *nay-ka e-cey swu-ceng-ha-yess-ten pha-il-tul-ul /tmp lo pok-sa-ha-ye-la* (spaces delimit Eojeols, meaning that "copy the files that I edited yesterday to /tmp"), a person may pause after *nay-ka* and after *e-cey swu-ceng-ha-yess-ten pha-il-tul-ul*, and after */tmp lo pok-sa-ha-ye-la.*

4) Phonological changes occur in a morpheme, between morphemes in an Eojeol, and between Eojeols in an Eonjeol. These changes include assimilation, dissimilation, contraction, and insertion. For example, a morpheme *pok-sa* is pronounced as *pok-ssa* (dissimilation, meaning "copy"), and *kwuk-min* is pronounced as *kwung-min* (assimilation, meaning "nationality"). An Eojeol *su-ceng-ha-yess-ten* is pronounced as *su-ceng-ha-yet-tten.*

3 SYSTEM ARCHITECTURE FOR SPEECH AND NATURAL LANGUAGE INTEGRATION

Our integration technique employs a phoneme lattice and a morpheme-level phonetic dictionary. This can be more microscopic integration compared with the classical approaches of using the word lattice and the word-level dictionary, such as the n-best integration technique which is mainly used for English. The phoneme lattice makes the phonological rule modeling possible in an early stage of spoken language processing. The phonological/morphological analysis can be performed together using the morpheme-level phonetic dictionary, and the dictionary size becomes stable regardless of the vocabulary size because new vocabularies can be generated by combining existing morphemes in the dictionary. Unlike the conventional integration method which uses the separate dictionaries for the speech recognition and the natural language processing, our integration model uses a unified morpheme-level phonetic dictionary together with the declarative morphotactic and phonotactic information. In our spoken language architecture, we employ a hierarchy of diphone spotting TDNNs for the acoustic-level processing, and develop a phonological/morphological co-analysis technique for the seamless integration. The output of the integrated architecture can be directly fed to the conventional natural language syntax/semantics analysis systems. Figure 2 shows the integrated spoken language pro-

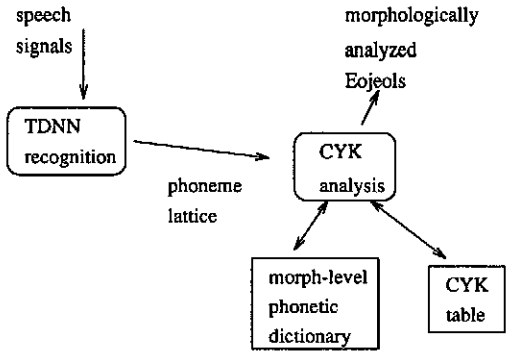


Figure 2: TDNN-CYK integration architecture

cessing architecture, a tightly coupled integration model of speech and natural language. The speech signal is analyzed using the TDNN diphone recognizer. The diphone recognizer also rearranges the diphone strings to produce the phoneme lattice. From the phoneme lattice, the morphological analyzer produces the morphologically analyzed Eojeols by handling the morphological segmentation, morphotactics verification, and the irregular conjugation. The phonological processing is integrated into the morphological parsing through the declarative phonological rule modeling. In the next section, we will explain the speech recognition and the morphological/phonological processing in detail.

4 Diphone-based speech recognition

For large-vocabulary continuos speech recognition, the sub-word level recognition must be supported. We selected a group of diphones for the sub-word unit because direct phoneme recognition in Korean is very difficult. The 46 Korean phonemes are very similar each other especially in the following cases: 1) the Korean diphthongs are hard to distinguish from the mono-vowels, and 2) the syllable-final consonants are hard to differentiate from the syllable-first consonants. The selected diphone groups (figure 3) have more information than the phonemes and are much fewer in numbers than the popular triphones

diphone groups	diphone numbers
V	21
C1V	378
VC2	147
C2C1	126

Figure 3: Korean diphone groups (V: vowel, C1: syllable-first consonant, C2: syllable-final consonant)

[11].

Figure 4 shows the diphone-based TDNN speech recognition system. The system consists of total 19 different TDNN networks for recognition of the Korean diphone groups.

The speech recognition is performed through the following steps (for more details, see [12]):

1) Pre-processing: The digitized speech signal is segmented into 200 msec size, 512 order FFTed and 16 step mel-scaled to obtain the filter-bank coefficients. For the end-point detection, the short-time energy and the zero-crossing rate are used. Each frame size is 10 msec and the 20 frames of 16 value normalized filter-bank coefficients are fed to the vowel group recognition TDNN.

2) Vowel group recognition: The input is 20 frame vectors (20*16 = 320 units) and the output is 18 units for the 18 vowel groups (the 17 groups according to the contained vowels and one CC group with no vowel). For each vowel group, separate diphone recognition TDNN is invoked, and the system has a hierarchical TDNN architecture. Each TDNN has the standard architecture which is well described in [7].

3) Diphone recognition: According to the

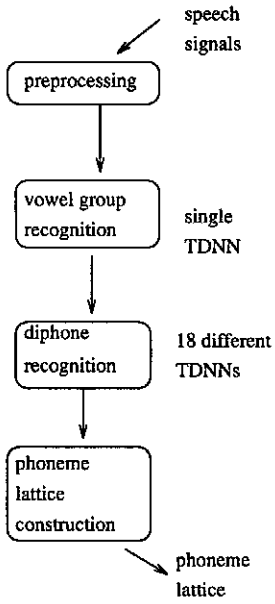


Figure 4: Diphone-based TDNN speech recognition system

recognized vowel group, each pertinent diphone recognition TDNN is activated. For each TDNN, the input is the same 20 frame vectors, and the output is the classified diphones for each vowel group. For example, for the /ya/ vowel group, there are total 15 output units: 9 for C1V type diphones (/kya/, /nya/, /tya/, /lya/, /mya/, /pya/, /sya/, /kkya/, /hya/), 5 for VC2 type diphones (/yak/, /yal/, /yan/, /yam/, /yang/) and one for V type diphone (/ya/). Each of the 18 TDNNs has the different number of output units according to the number of diphones in each vowel group.

4) Diphone2phoneme decoding: From the resulting diphone sequences, this step obtains the phoneme lattice which contains the candidate phoneme sequences. We use a simple deterministic decoding heuristics without any probabilistic calculations, and try to maintain all the possible diphone spotting results in the phoneme lattice since the later phonological/morphological processing can safely prune the incorrect recognitions. The decoding begins by grouping the diphones into the same types (C1V, V, VC2, C2C1 types). The frequency count for each diphone, that is, the number of specific diphones per 10 msec frame shift, is utilized to fix the insertion errors by deleting the lower frequency count diphones, and finally the diphones are split into the constituent phonemes by merging the same phonemes in the neighboring diphones. This simple non-probablistic decoding scheme surprisingly works well for our domain, and the resulting phoneme lattice reliably provides all the possible output phonemes in the speech recognition.

5 Morphological analysis from the phoneme lattice

The morphological analysis transforms the phoneme lattice into the sequences of morphologically analyzed Eojeols (which is a unit of spacing in Korean orthography and usually consists of single noun or verb-stem plus several functional morphemes). Our morphological analysis takes a phoneme lattice rather than a phoneme string as an input since we want to have a chance to exploit all the speech recognition results during the morphological analysis. The phoneme lattice provides alternative phonetic transcriptions of speech sounds which must be transformed to produce the orthographic morpheme strings. Unlike the conventional morphological analysis from the written text input, the morphological analysis of the *phoneme lattice* must solve the following subproblems: 1) The phonetic transcriptions must be segmented and mapped into the orthographic morphemes which are basic units of written language processing. 2) The phonological changes that can be captured by the Korean phonological rules must be modeled and processed during the morphological analysis. 3) An efficient dictionary search is required because the phoneme lattice results in exponential number of phoneme chains.

The Korean morphological analyzer [13] was implemented based on the well-known CYK parsing technique [5] and augmented in order to handle the Korean phonological changes and phoneme lattice input. Figure 5

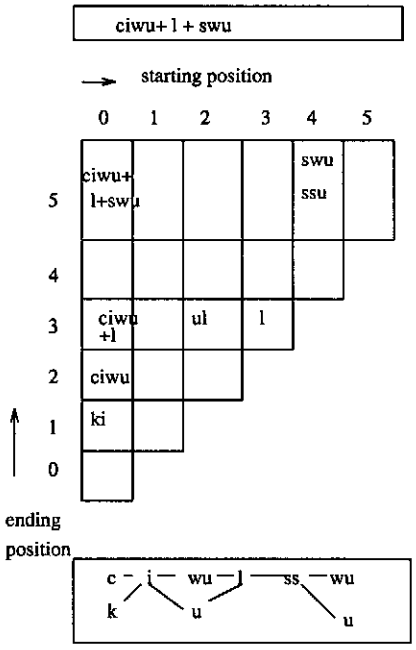


Figure 5: Morphological parsing of the phoneme lattice (from top: morphologically analyzed output Eojeol, CYK triangular table, input phoneme lattice). The example phoneme lattice was obtained from the input speech *ci-wul-ssu* (deletable) using the diphone-based TDNN speech recognition system, and the morphological analysis produces *ci-wu+l+su*, where "+" is the morpheme boundary, and "-" is the syllable boundary. The CYK triangular table was filled in with all the possible morphemes which are obtainable from the dictionary look-up, and also with all the possible morpheme combinations.

shows our morphological analysis scheme for the phoneme lattice.

The basic process of the Korean morphological analysis consists of the morpheme segmentation, checking the possible morpheme connectivity (handling of the morphotactics), and the reconstruction of the original morphemes from the irregular conjugations (handling of the orthographic rules).

The morpheme segmentation is performed using the morpheme entry in the dictionary. During the left to right scanning of the input text, when the morpheme is found in the dictionary, it is enrolled in the CYK table in the proper position. For example, in figure 5, the 3 different morphemes, that is, verb *ci-wu*, adnominalizing verb-ending *l*, and the bound noun *swu* are enrolled in the position (0,2), (3,3), and (4,5) respectively. The position (i,j) designates the start and end position of the input characters, and the verb *ciwu* starts in the position 0 (first position) and ends in the position 2, hence consists of 3 characters. We enroll all the matched morphemes on the input string in the CYK table (see figure 5 for other possible morphemes). During the segmentation, the possible morpheme connectivity must be checked for the selection of the correct morpheme boundaries for the input string. The morpheme connectivity can be verified from the Korean morphotactic information. The morphotactic information is included in the dictionary using the specialized Korean part-of-speech symbols (called connectivity information) [13]. We divided the major 13 Korean part-of-speech symbols into about 200 different refined symbols (tags) for the efficient verification of the connectivity of each morpheme, and constructed the morpheme connectivity matrix which designates the possible relative placement of 200 refined part-of-speech tags in the string. For example, in figure 5, the morpheme *ci-wu* (verb stem, meaning "delete") can be in the left side of the morpheme *l* (adnominalizing verb-ending) because the morpheme connectivity matrix verifies that the connection of *verb stem* to the *adnominalizing verb-ending* is legal. The CYK table provides the possible positions of the connectivity checking. For example, in figure 5, the connectivity information of *ci-wu* and *l* is worth checking because the position (0,2) and (3,3) can be concatenated to produce the position (0,3) so the result *ci-wu+l* is put in the position (0,3). The irregular conjugations are handled in declarative way by putting the inflected forms as well as the original forms of the morphemes in the dictionary.

The above-mentioned basic morphological analysis scheme was augmented to solve the

phonetic transcription header	original morpheme	left morphological connectivity	right morphological connectivity	left phonemic connectivity	right phonemic connectivity
ci-wu	ci-wu	regular verb	regular verb	'c' sound no-change	'wu' sound no-change
l	l	adnominalizing verb-ending	adnominalizing verb-ending	'l' sound no-change	'l' sound no-change
sswu	swu	bound-noun	bound-noun	's' sound change to 'ss'	'wu' sound no-change

Figure 6: The morpheme-level phonetic dictionary. The figure shows three different morpheme entries *ci-wu, l, swu* with their phonetic transcription headers, orginial morphemes, left and right morphological connectivity, and left and right phonemic connectivity information. In the actual dictionary implementation, the morphological and phonemic connectivity information is encoded using the specialized symbols. The left and right distinction is for the morphemes that have the different connectivity information to the left concatenated and right concatenated morphemes.

three sub-problems in handling the phoneme lattice input and the phonological changes during the morphological analysis.

1) For phonetic transcription into the orthographic morpheme mapping, we indexed each morpheme in the dictionary by the corresponding phonetic transcription header, and constructed so called morpheme-level phonetic dictionary. The single phonetic transcription can be associated with many different morpheme entries for the homophone style morphemes. In this way, the accessing of phonetic headers can lead to all the corresponding morphemes in the orthographic forms. Figure 6 shows the morpheme-level phonetic dictionary.

2) In Korean, the phonological changes can occur within the morpheme or across the morpheme boundary. For the former case, the phonetic transcription headers in the dictionary already reflect the phonological changes since the dictionary entry is the whole morpheme. However, for the latter case, we have to model the Korean phonological rules to handle the between-morpheme-phonological changes. We declaratively modeled the major Korean phonological rules including the 2nd consonant standardization, consonant assimilation, palatalization, glottalization (consonant dissimilation), and insertion according to the Korean Ministry of Education Standard, and processed the Korean phonology during the morphotactic verification. The declarative phonological rule is encoded in the left and right phonemic connectivity information in the dictionary. For example, the bound noun *swu* has phonetic realization *sswu* after the *l* sound. Figure 6 designates the phenomenon in the left phonemic connectivity information in the *swu* entry. The separate phoneme connectivity matrix records all the possible relative phoneme placement much like the morpheme connectivity matrix. When the morphotactics is checked, the phoneme connectivity matrix is also checked to verify the possible phonological changes between the morphemes.

3) To handle the phoneme lattice search, we use the TRIE indexing for the fast dictionary access [14]. The breadth-first search on the TRIE structures for the phonetic transcription header can prune the unnecessary paths efficiently, and hence deal with the complexity of the phoneme lattice search.

6 Implementation and the Experiment Results

The TDNN-CYK spoken language system was implemented using C and standard X-window user interface under the UNIX/Sun Sparc platforms. The system's inputs are carefully articulated Korean speech in the normal laboratory environment, and the outputs are morphologically analyzed Eojeol sequences that can be directly fed to the conventional natural language syntax analysis system [10]. We constructed a 1000 entry morpheme-level phonetic dictionary in the UNIX operating system domain, and about over 100 entries of morpheme connectivity and phoneme connectivity matrix for the phonological/morphological analysis.

The dictionary is indexed using the phoneme-based TRIE to handle the phoneme lattice search. Since we don't have any standard segmented Korean speech database yet, we constructed our own by recording and manually segmenting 73 most frequent Korean diphones. The 73 diphones are acquired from the 300 Korean Eojeols (each Eojeol is pronounced 15 times by a female speaker) in the 100 Korean sentences, which can appear in the natural language commanding to the UNIX operating system[15].

Several experiments were performed to verify the system's performance of time-shift invariance, diphone recognition, and final Eojeol recognition including the morphological analysis. Belows are the brief results of each performance test. In each experiment, the input speech patterns are prepared as follows: Eojeols were recorded in a normal laboratory environment with an average S/N ratio of 12 dB. Speech data were sampled at 16kHz, and hamming windowed. From this windowed data, 512-point DTFTs were computed at 5 msec intervals. The DTFTs were used to generate 16 Mel-scale filter-bank coefficients at 10 msec intervals [7]. These spectra were normalized to produce suitable input levels for the four-layer structured TDNNs. We used hyperbolic arc tangent error function in the weight updating [16] in the back propagation training. We updated the weights after a small number of iterations [17].

6.1 Time-shift invariance of Korean diphones

We generated 2400 diphone samples for the typical 12 Korean diphones. The input patterns for the two tests are set the same in order to compare the *no shift* and *shift* cases. Figure 7 shows that the Korean diphone recognition has the time-invariance property of TDNN and suggests the optimal time interval near 200 - 250 msec for the diphones. These results imply that the context-independent diphone-based TDNN recognition is possible.

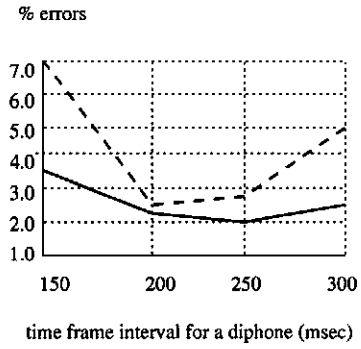


Figure 7: Average error rate of the segmented time frame (solid lines) versus the same time frame with maximum 40 msec left or right temporal shift (grey lines)

unit of recognition	number of targets	number of samples	recognition rate
phoneme	9	1080	94.06%
	17	2040	89.80%
diphone	9	1080	95.42%
	17	2040	95.27%

Figure 8: Diphone recognition versus phoneme recognition test

6.2 Comparison of diphone recognition vs. phoneme recognition

This experiment is to show that the diphone can improve the recognition rate of the Korean vowels regardless of many rising diphthongs, compared with the phoneme recognition. In the test, we set 150 msec time range for the phoneme and 200 msec for the diphone segmentation. Compared with the phoneme recognition case, figure 8 shows that the recognition rate of diphones doesn't decrease much when the number of targets with similar features doubly increases. Moreover, the unit with more than one feature can be efficiently recognized at the high rate in the diphone recognition.

a. continous diphones

	total	correct	delete	insert
pattern size (rec. rate)	7772	7259 (93.4%)	513 (6.6%)	3000 (38.6%)

b. segmented diphones

	vowel group	sub-TDNNs average	total average
rec. rate	94.8%	98.2%	93.8%

Figure 9: Continuous diphone spotting versus segmented diphone spotting

6.3 Performance of continuous diphone recognition

In this experiment, we extracted most typical 72 diphones in Korean from 66 Eojeols, each of which is pronounced 15 times to generate about 5500 diphone patterns. The 5500 training samples are used to train the vowel group TDNN and 10 different sub-TDNNs for each diphone group. During the recognition, the new 262 Eojeols are selected to generate the test patterns of 2432 Eojeols, and shifted 30 msec during the application to obtain the TDNNs diphone spotting performances in a continuous speech. Figure 9-a shows the continuous diphone spotting performance. We have total 7772 target diphones from the 2432 test Eojeol patterns. The *correct* designates that the correct target diphones were spotted in the testing position, and the *delete* designates the other case. The *insert* designates that the non-target diphones were spotted in the testing position. To compare the ability of handling the continuous speech, we also tested the diphone spotting using the hand segmented test patterns with the same 7772 target diphones. Figure 9-b shows the segmented diphone spotting performance. Since the test data are already hand-segmented before input, there are no insertion and deletion errors in this case. The fact that the segmented speech performance is not much better than the continuous one (93.8% vs. 93.4%) demonstrates the diphone's suitability to handling the continuous speech.

6.4 Performance of continuous Eojeol recognition

In order to test the ability of the full Eojeol recognition including the phoneme decoding and morphological analysis performance, a middle-vocabulary experiment was carried out. The task is a speaker-dependent and continuous Eojeol recognition which produces the morphologically analyzed Eojeol sequences. In the process, the speech recognizer produces the phoneme lattice that includes the correct phoneme sequence in the input Eojeol, and then the morphological analyzer produces the analyzed Eojeol from the phoneme lattice. So, in this task, all the intermediate steps, that is, diphone spotting, phoneme lattice decoding and morphological analysis from the phoneme lattice, are combined to produce the final recognition performance. The same 262 Eojeols in section 6.3 are fed to the total integrated system that has the pre-trained TDNN networks. Figure 10 shows the final performance of the continuous Eojeols. We have total 9605 target morphemes from the same 2432 test Eojeol patterns used in section 6.3. In the figure, the *correct* designates that the correct morpheme sequences can be analyzed from the speech input, and the *delete* means that the correct morpheme sequences cannot be generated. The *insert* designates the percentage of the spurious morphemes that are generated from the insertion errors. The performance is above 80% in the final morphological analysis success rate, which is promising but still relatively low compared with the continuous diphone recognition. The relatively low performance is due to the large insertion errors during the long range of continuous speech which cannot be handled properly in the phoneme lattice decoding. However, the morphological analyzer performed perfectly when the phoneme lattice contains the correct phoneme sequences.

	total	correct	delete	insert
pattern size (rec. rate)	9605	7696 (80.1%)	1909 (19.8%)	7182 (74.76%)

Figure 10: Continuous Eojeol recognition including morphological analysis

7 Comparison with the related researches

Recently, the idea of sending only the best n speech recognition results to the natural language system has been implemented using the time-synchronous Viterbi-style beam search algorithm [1]. The algorithm was also improved by the word-dependent search [2] and by adding the A* backward tree search [18]. The n-best integration is mainly utilized for the HMM-based continuous speech recognition systems, and many existing speech systems and natural language systems were successfully integrated using the n-best word search techniques [3, 4]. However, until now, the n-best search techniques are only implemented to directly produce the n-best sentences using the word sequences or word lattice, and this word-level integration was successful for the morphologically simple languages such as English. On the contrary, our integration is at the phoneme-level using the phoneme lattice because we need more sophisticated phonological/morphological handling in the integration process. The word-level n-best integration also assumes the word-level dictionary which is an unreasonable assumption for the morphologically complex languages.

The HMM-LR integration [19, 20] was implemented using the HMM's phoneme spotting ability integrated with the generalized LR parsing techniques [21]. Unlike the n-best integration, the HMM-LR integration was more tight and implemented at the phoneme-level by extending the LR parser's terminal symbols to cover the phonetic transcriptions. In this scheme, the LR parsing selects the most probable parsing results by obtaining the probability of the end-point candidate phonemes from the HMM's forward probability calculation. So the total integrated system is working by the LR parser's prediction of the next phoneme candidates which are then verified by the HMM's phoneme spotting abilities. The idea of extending the LR grammar to the phonetic transcriptions seems to be working for the phoneme-level integration. However, the scheme doesn't have any separate language-level dictionary, which results in the degenerated phonological/morphological processing, and also has the difficulty in the necessary scale-ups. On the contrary, our TDNN-CYK integration focuses on the general phonological/morphological handling which is essential for the agglutinative languages.

The idea of extending LR grammar to the phonetic transcriptions was also applied to the TDNN-LR integration method [9, 8] which was similarly implemented by replacing HMM's phoneme spotting by the TDNN's phoneme spotting. The integration was implemented by dynamic time warping (DTW) level-building search [22] between TDNN's phoneme sequences and LR grammar's phoneme sequences. However, the performance was relatively poor compared with the HMM-LR integration method [9]. There are basically two reasons for the poor TDNN-LR performances compared with the HMM-LR integration: 1) the TDNN model has rarely been applied to the practical large vocabulary systems yet, therefore it lacks in the fine tuning compared with the popular HMM models, and 2) the TDNN model has yet to find a right way to be effectively integrated into the natural language processing model. The HMM model supports a natural integration into the general chart-based parsing models such as generalized LR parsing because there are well-defined probablistic search techniques to be integrated. However, output activations of the multiple TDNNs are difficult to normalize and therefore difficult to naturally integrate into the popular probabilistic search schemes such as Viterbi

search. Our TDNN-CYK method doesn't employ any probabilistic search in its integration, but send the entire phoneme lattice to the morphological analyzer. In this way, we can exploit all the TDNN's outputs in the language processing level which is somewhat inefficient but safe for the current scheme.

8 Conclusion

This paper presents a phoneme level integration of speech and natural language in a connectionist speech recognition model for agglutinative languages such as Korean. Our model's main contribution is to define the phoneme level integration that can support sophisticated phonological/morphological processing in the integration of speech and language, which is essential for the morphologically complex agglutinative languages. Also, the TDNN-CYK integration is a first attempt to develop a morphologically general integration model using the connectionist speech recognition paradigm.

Our TDNN-CYK spoken language architecture has many novel features for speech and natural language processing. First, the diphone-based TDNN proposes a nice subword unit of recognition, well reflecting the Korean phonetic characteristics. Secondly, the morphological analysis combined with the declarative phonological rule modeling is well suited to the phonetic transcription into the orthographic morpheme mapping, which is an essential task for every spoken language processing model. Finally, the TRIE structured phonetic transcription indexing can serve to reduce the phoneme access complexity in the direct morphological analysis from the phoneme lattice.

The experiments show that the final Eojeol recognition is over 80% in the middle-vocabulary speaker-dependent continuous Eojeol recognition, which is very promising in considering the continuous speech and the combination of several steps of performances such as diphone spotting, phoneme lattice decoding and morphological analysis. However the performance is relatively low compared with the continuous diphone recognition (which is over 93% in the same condition) because of the enormous insertion errors for long duration speech (Eonjeol or phrase). To recover from the insertion errors, we plan to incorporate an error correcting scheme into our phoneme decoding process that will result in the error-free phoneme lattice from which the morphological analyzer can produce the perfect analysis results.

References

[1] Y. L. Chow and R. Schwartz, "The n-best algorithm: An efficient procedures for finding top N sentence hypothesis," in *Proceedings of the second DARPA workshop on speech and natural language*, Los Altos, CA, 1989, Morgan Kaufmann Publishers, Inc.

[2] R. Schwartz and S. Austin, "Efficient, high-performance algorithms for n-best search," in *Proceedings of the third DARPA workshop on speech and natural language*, Los Altos, CA, 1990, Morgan Kaufmann Publishers, Inc.

[3] M. Agnas, H. Alshawi, I. Bretan, D. Carter, K. Ceder, M. Collins, R. Crouch, V. Digalakis, B. Ekholm, B. Bamback, J. Kaja, J. Karlgren, B. Lyberg, P. Price, S. Pulman, M. Rayner, C. Samuelsson, and T. Svensson, "Spoken language trnaslator: first year report," Technical Report ISRN SICS-R-94/03-SE, Swedish Institute of Computer Science and SRI International, 1994.

[4] M. Bates, R. Bobrow, P. Fung, R. Ingria, F. Kubala, J. Makhoul, L. Nguyen, R. Schwartz, and D. Stallard, "The BBN/HARC spoken language understanding system," in *Proceedings of the ICASSP-93*, 1993.

[5] A. Aho and J. D. Ullman, *The theory of parsing, translation, and compiling, Vol 1: parsing*, Prentice-Hall, Englewood Cliffs, NJ, 1972.

[6] D. Morgan and C. L. Scofield, *Neural networks and speech processing*, Kluwer Academic Publishers, Inc., 1991.

[7] A. Waibel, T. Hanaazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 3, pp. 328 – 339, 1989.

[8] H. Sawai, "The TDNN-LR large-vocabulary and continuous speech recognition system," in *Proceedings of the international conference on spoken language processing (ICSLP)*, 1990.

[9] H. Sawai, "TDNN-LR continuous speech recognition system using adaptive incremental TDNN training," in *Proceedings of the ICASSP-91*, 1991.

[10] W. Lee, G. Lee, and J. Lee, "Table-driven neural syntactic analysis of spoken Korean," in *Proceedings of COLING-94*, 1994.

[11] K. F. Lee, *Automatic speech recognition*, Kluwer Academic Publishers, Inc., 1989.

[12] K. Kim, G. Lee, and J. Lee, "Integrating TDNN-based diphone recognition with table-driven morphology parsing for understanding of spoken Korean," in *Proceedings of the international conference on spoken language processing (ICSLP)*, 1994.

[13] E. C. Lee and J. H. Lee, "The implementation of Korean morphological analyzer using hierarchical symbolic connectivity information," in *Proceedigns of the 4th conference on Korean and Korean information processing*, 1992 (in Korean).

[14] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data structures and algorithms*, Addison-Wesley Publishing Company, 1983.

[15] W. Lee and G. Lee, "From natural language to shell-script: A case-based reasoning system for automatic unix programming," *Expert systems with applications: An international journal*, vol. 9, no. 2, , 1995 (in press).

[16] S. E. Fahlman, "Faster-learning variations on back-propagation: An empirical study," in *Proceedings of the 1988 connectionist models summer school*, 1988.

[17] P. Haffner, A. Waibel, H. Sawai, and K. Shikano, "Fast back-propagation learning methods for large phonemic neural networks," in *Proc. of the Eurospeech-89*, 1989.

[18] F. K. Soong and E. Huang, "A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition," in *Proceedings of the third DARPA workshop on speech and natural language*, Los Altos, CA, 1990, Morgan Kaufmann Publishers, Inc.

[19] K. Kita, T. Kawabata, and H. Saito, "HMM continuous speech recognition using predictive LR parsing," in *Proceedings of the ICASSP-89*, 1989.

[20] T. Hanazawa, K. Kita, S. Nakamura, T. Kawabata, and K. Shikano, "ATR HMM-LR continuous speech recognition system," in *Proceedings of the ICASSP-90*, 1990.

[21] M. Tomita, *Efficient parsing for natural language - A fast algorithm for practical systems*, Kluwer Academic Publishers, 1986.

[22] C. Myers and L. Rabiner, "A level building dynamic time warping algorithm for connected word recognition," *IEEE Trans. on ASSP*, vol. 29, no. 2, pp. 284–279, 1981.

Generation of spoken monologues by means of templates

Kees van Deemter Jan Landsbergen René Leermakers Jan Odijk

Institute for Perception Research (IPO)
P.O. Box 513
5600 MB Eindhoven, The Netherlands
email: {deemter,landsbrn,leermake,odijkje}@prl.philips.nl

ABSTRACT

The paper describes a system that produces spoken monologues derived from information in a database. The sentences of these monologues are generated from templates of syntactic structures, which may contain open slots in which other elements, usually NPs, can be inserted. It is shown how these sentences string together to form a coherent message. This message has to be pronounced correctly, which means, among other things, that it has to be prosodically acceptable. It is indicated how linguistic information is used to arrive at an acceptable prosodic structure, which, in turn, feeds into a module which takes care of the phonetic realization of the monologue.

1 INTRODUCTION

Recently, a number of telephone services have entered the market in which spoken language is generated by using pre-recorded speech in which appropriate words can be inserted at specific positions. There is a large discrepancy between these systems and the systems aimed at by most research on text generation (see e.g. [Dale et al 1992]), which are supposed to contain a comprehensive grammar that is controlled by a component with abstract reasoning capabilities. We will propose a system architecture that is less ambitious in the latter respects, but that focuses on spoken language and tries to meet the demands of a new class of applications.

As a research vehicle we are currently building a system that produces spoken monologues derived from information stored in a database about compact discs with compositions written by Mozart. The database contains information about the compositions, the performers and the recordings. The purpose of the monologue generator described here is to generate from these data a large variety of coherent texts, including all information required for a correct pronunciation. The monologues produced are interleaved with musical fragments.

A generator like this could be part of an electronic shopping system, where users can express their interest in a certain area without being very specific, and where the system provides information and 'sales talk'. There is a clear distinction with question-answering systems, which are supposed to give precise answers to precise questions. The way in which users can indicate their areas of interest will not be discussed in this paper.

A database representation of a recording could be:

KV 309
DATE 10/1777 - 11/1777
SORT piano sonata
NUMBER 7
PERFORMER Mitsuko Uchida
PLACE London
VOLUME 17
CD 2
TRACK 4

Our current system could, after a client has shown interest in the composition described by the above database object, come up with the following:

> You are now going to listen to a fragment of piano sonata number seven, K. 309, played by Mitsuko Uchida. It was composed for Rosine, the daughter of the court musician and composer Christian Cannabich in Mannheim. Influences of the Mannheimian orchestral techniques are discernible. The composition was written from October 1777 to November 1777. The recording of K. 309 was made in London. It starts at track four of the second CD of volume 17.

2 SYSTEM ARCHITECTURE

An important system requirement is that a large variety of texts can be produced from the same database structures. Presentations are generated on the basis of database information by making use of *templates*: structured sentences with variables, i.e. open slots for which expressions can be substituted. These templates indicate how the information provided by a (part of a) database object can be expressed in natural language. The required variety is achieved by having many different templates for the same information and by having a flexible mechanism for combining the generated sentences into texts. In addition, information is available that does not fit in the uniform database format. This is called *gossip*, and is represented by object-specific templates expressing this information. In the example text in section 1, the mentions of Rosine and the Manheimian techniques are 'gossip'. The remainder of the presentation has been generated by general templates. A template can be used, in principle, if there is enough information in the database to fill its slots. However, there are extra conditions to guard the well-formedness and effectiveness of presentations. These will be discussed in section 3. A simple example of a condition is that the same information should not be expressed twice in the same presentation. If a piece of information has been expressed at an earlier time, say during a previous presentation, this may also inhibit the use of certain templates expressing it. The appropriateness of a sentence at a particular time also depends on the topic being discussed, i.e. on the information expressed by foregoing sentences. All this means that it is important for the system to maintain a *knowledge state*, a record showing which information has been expressed, and when it has been expressed.

Many variations of the above presentation are possible. One could, for instance, start with mentioning the date of composition, or information could be added that contrasts this composition with a previous one. Note that even a small modification, like the insertion of an extra sentence, or the permutation of two adjacent sentences, has consequences within sentences. For instance, there are various ways to refer to the composition being discussed, by name (*K. 309*), with a definite noun phrase (*the composition*), or with a pronoun (*it*); the appropriateness of a referring expression depends, among others, on the existence and kinds of references to the referred object in previous sentences. For instance, if the second sentence above would be the opening sentence, it could not refer to the composition by 'it'. This means that it is important to maintain a *context state*, a record of which objects have been introduced in the text, and how and when they have been referred to.

As was mentioned above, templates in our system are structured sentences with slots. A simplified example is given in figure 1. The slots

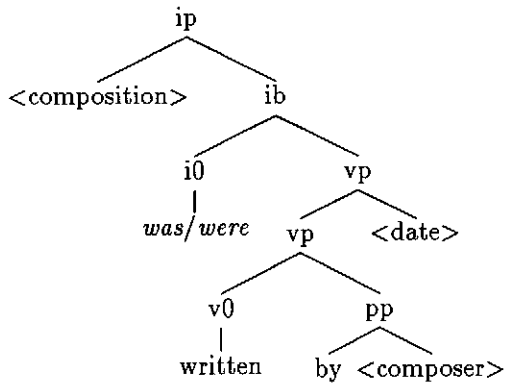


Figure 1: Template for a sentence like "K. 32 was written by Mozart in March 1772". *Composition, composer, date* are the variables of the template. *Was/were* indicates that a choice must be made, depending on the subject.

composition, composer, date are to be filled with structured expressions that contain database information. This is done with other, smaller, templates. As was mentioned above, the way a slot can be filled depends on circumstances. For more details, see section 3. One reason for having templates of syntax trees is that the prosodic module of our system, in which accents and pauses are determined, needs the syntactic structure of sentences. Another is that this structure enables one to write rules to check the syntactic correctness of the sentences after slot-filling and to adjust them if necessary.

The architecture of our system is shown in figure 2. It displays three modules: *Generation*, *Prosody* and *Speech*. Module *Generation* generates syntax trees on the basis of the database, a collection of templates, and it maintains the knowledge and context states. It is described in more detail in section 3. Module *Prosody* transforms a syntax tree into a sequence of annotated words, the annotations specifying accents and pauses. The internals of the module *Prosody* will be described in section 4. Module *Speech* transforms a sequence of annotated words into a speech signal. In this paper, it is not discussed in detail.

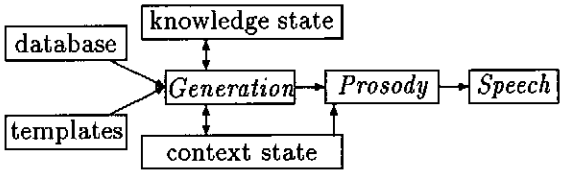


Figure 2: System Architecture: *Generation* generates syntax trees on the basis of the database and a collection of templates, and it maintains the knowledge and context states. *Prosody* operates on syntax trees and the context state to produce a sequence of annotated words.

3 Text Generation

The purpose is to generate a coherent, varied, entertaining and correctly pronounced monologue which conveys the information from the database, e.g. the information about a particular composition by Mozart. It will be discussed here how we can generate a text which conveys this information and which satisfies these requirements. We will concentrate on three aspects: (1) the generation of a text; (2) how to achieve its coherency, and (3) how to achieve the required variation. The correct pronunciation of the texts is dealt with in section 4.

As a basis for this discussion we will assume that the database contains information about Mozart's life, e.g. that he lived from 1756 to 1791, and that information about the compositions is stored in the following format:

KV 32
TITLE Galimathias Musicum
DATE 3/1766
SORT quodlibet

3.1 Sentences generated from templates

A text is a sequence of sentences. So, first of all, sentences must be created. As explained in the previous section, they are generated by means of *templates*. A template indicates how the meaning of a database record (or a part of a record, or a combination of records) can be verbalized. Since there are various ways to verbalize the content of a record, and many ways to group information from different records into one verbalization, this will lead to a large number of possible sentences for conveying the database information.

In the examples below we will use the template introduced in figure 1 (repeated in (1a)) and a new one (1b). For expository convenience, we will represent only the terminals of templates. Variable parts (e.g. <composition>) will be represented between angled brackets.

(1) a <composition> was/were written by <composer> <date>
b We will now present information about <composition>.

A sentence can be constructed from a template by filling the variable parts. Example sentences derived from the two templates could be:

(2) a We will now present information about 'Galimathias Musicum'.
b K. 32 was written by Mozart in March 1766.

Of course, we might have made different choices. Some other sentences which can be generated from these two templates are:

(3) a We will now present information about a quodlibet.
b We will now present information about K. 32.
c This quodlibet was written by the composer in March 1766.
d 'Galimathias Musicum' was written by a composer in the same year.

Having several different options to choose from has two useful functions. First of all, in combination with the availability of a large number of templates it will supply the system with the possibility to vary the texts generated: a different text will result each time information about a composition is generated. A second function is that it supplies the system with sufficient options to generate a text at all: not every option chosen can be used in each situation, as we will see below. When a selected option is rejected, the availability of other options makes it possible to attempt a different way of expressing the same information.

One might ask where the expressions that substitute for variables come from. In general, these expressions are generated themselves from templates again. In some cases, these templates will be fixed expressions. Thus, most variables can be replaced by appropriate pronominal expressions (e.g. *it, she, his*). In certain cases such expressions can be constructed directly on the basis of information in the database by very general rules.

E.g., if a composition has a title, then an expression consisting of this title can be substituted for variables for this composition.

But the substituted expressions can also be the result of substitutions in other templates. The database supplies a value for the SORT attribute, and a general rule states that the composition can be expressed by a structure of the form *the <sortexpression>* or *a <sortexpression>*, where <sortexpression> is a structure which expresses the value of the sort attribute in English. A more complex example is the template *when <composer> was only <age> years old*, where <composer> is a variable for an expression for the composer of the relevant work, and <age> is a variable for the age of the composer at the moment he wrote the work. The latter information is not directly available in the database and must be computed. The template of figure 1 might thus also lead to a sentence such as *The quodlibet was written by Mozart when he was only ten years old.*

The fact that templates are structured objects makes it possible to formulate various conditions on the form of variable parts. In this way, it is possible to avoid the generation of incorrect sentences such as:

(4) a *We will now present information about he.
b *It were written by him when Mozart was only ten years old.

In the first example the pronoun *he* is selected to express the composition, but that is wrong in two respects. First, *he* is not an appropriate pronoun for compositions (but only for persons), and second, in the sentence given its form should be *him*, not *he*. In the second sentence, the choice for the finite verb *were* is incompatible with the singular subject *it*, and the co-occurrence of *him* and *Mozart* suggests that these expressions refer to two different persons, though they actually refer to one and the same person.

Since templates are structured objects, conditions guaranteeing the appropriate choice of pronouns can refer to information contained in these structures (e.g. that *he* refers to persons and that *he* is governed by the preposition *about*).

Similarly, it can be read off the syntactic structure that the pronoun *it* is the singular subject of the second sentence and that therefore the finite verb should be *was*. The infelicitous choice of *him* and *Mozart* is prevented by a more complex condition on the proper sentence-internal distribution of pronouns, proper names and other expressions. This condition is a (currently somewhat simplified) version of the so-called *Binding Theory* (see [Chomsky 1981], [Chomsky 1986]), which is crucially formulated in terms of configurations in syntactic structure.

It is also possible to formulate a condition on the categorial status of certain expressions. Thus, the date at which Mozart completed this composition must be expressed by means of a prepositional phrase (PP: *in March 1766*) or an adverbial subordinate clause (introduced by *when*), but not by a noun phrase (NP: *March 1763*), although in other contexts this may be the appropriate form (e.g. if it is used to express a period: *from <date1> to <date2>*).

At the moment we use so-called surface syntactic structures as the syntactic structures for templates, but this choice was made mainly for pragmatic reasons. Other options which might be considered include 'deeper' syntactic levels of representations (e.g. *syntactic derivation trees* or even *semantic derivation trees*, cf. [Rosetta 1994]). Each of these representations has its own advantages and disadvantages, which we will not discuss here, but all are compatible with the template approach as sketched here.

3.2 Which templates are to be used?

Now we are able to use a large variety of sentences to convey the relevant information, but it is as yet unclear which sentences should be used in a given situation. This problem is solved in two steps..

First of all, it has to be determined what is going to be said. This is represented in the form of a set of attributes and relations from the database. It is assumed that some other mechanism makes a selection from the attributes and relations from the database. This mechanism must take into account which composition is to be described and which compositions have been described earlier, but also wishes of the user to determine the length and degree of detail of the text to be generated, and the perspective from which to present the relevant information (e.g. if the user has special interests). This mechanism is a second source of variation of the texts generated.

Secondly, a selection has to be made from all templates in such a way that the text generated conveys all and only the required information. A minimal requirement is that those templates are selected which are able to convey the relevant in-

formation. An additional requirement is that, under normal circumstances, the same information is presented not more than once. Furthermore, the form in which this information is presented should vary to avoid stylistic infelicities. These requirements have been incorporated in the text generator, which also has as its task to present the sentences in such a way that the text shows a certain coherence. Information should be grouped into convenient clusters and presented in a natural order.

We will not deal here with the exact mechanisms of the text generator. Many approaches are possible here. One method would be to write an explicit text grammar which defines the possible order of sentences. In the current system we have adopted a different approach. In this approach each template 'attempts', so to speak, to get a sentence generated from it into the text. Whether this succeeds depends on the information conveyed by the sentence, which information has been conveyed earlier, and whether the sentence can find a place in a natural grouping of sentences in paragraphs. The method is characterized by the fact that only local conditions on the knowledge state and the properties of the current template (in particular: which information does it convey) determine whether a sentence is appropriate at a certain point in the text. As a consequence, no global properties of the text are considered and no explicit planning is involved. For a more detailed description of this approach we refer to [Odijk 1994].

We can, however, illustrate the text generator with the two example templates given above. A reasonable approach would be to require that a text first introduces an entity before it describes its additional properties. Given this assumption, a text which opens with a sentence generated from template (1b) is more natural than a text which opens with a sentence generated from template (1a):

(5) a We will now present information about K. 32. This quodlibet was written by Mozart when he was only ten years old.
b K. 32 was written by Mozart when he was only ten years old. We will now present information about this quodlibet.

3.3 The use of proper names

A second aspect of text coherence is the proper use of linguistic means within sentences. The sentence-internal order of constituents is one example. Other examples relate to the correct use of proper names and anaphoric devices. We will briefly discuss the latter two.

We have seen that sentences of texts can be generated with the given templates by substituting expressions for the variable parts. One type of expression which can fill such slots is a proper name. For the composition which we currently discuss there are two expressions which can count as such, viz. the title of the composition (*'Galimathias Musicum'*) and the expression for its KV number (*K. 32*). However, we cannot use these proper names just anywhere. Consider the following text:

(6) We will now present information about K. 32. 'Galimathias Musicum' was written by Mozart when he was only ten years old.

In this text we have substituted the two proper names *'Galimathias Musicum'* and *K. 32* for the two occurrences of the variable <composition>. But the resulting text is only felicitous if it can be taken for granted that 'Galimathias Musicum' and 'K. 32' are the same composition. If (6) is the beginning of the monologue, this does not seem plausible. Apparently, rules must be formulated which determine when different proper names which refer to the same object can be used. There are various devices to establish this relation. One way is to use circumscriptions such as *K. 32, also known as 'Galimathias Musicum'* or *'Galimathias Musicum', with KV number K. 32.* Another way is to use one proper name as an apposition to the other proper name (e.g. *We will now present information about K. 32, 'Galimathias Musicum'*).

Similarly , there must be rules to avoid excessive use of the same proper name for one object. The texts in (7), in which the same proper name is used twice are less felicitous than e.g. (5a), and though these two-sentence texts are still relatively acceptable, the continuous use of the same proper name in a longer text will lead to results which are increasingly worse.

(7) a We will now present information about K. 32. K. 32 was written by Mozart in March 1766.
b We will now present information about 'Galimathias Musicum'. 'Galimathias Musicum' was written by Mozart in March 1766.

3.4 ANAPHORIC DEVICES

A too extensive use of the same proper name in a longer text can be avoided by using other anaphoric devices, i.e. expressions which refer to other entities in the context. Natural language provides one with several different anaphoric devices. Well-known examples are various pronouns, such as *he, it, his, they, himself, etc.*. Other examples are definite descriptions, i.e. noun phrases introduced by the definite article *the* (e.g. *the composition, the quodlibet*), 'demonstrative descriptions', i.e. noun phrases introduced by the determiners *this, that, etc.* (e.g. *this quodlibet*), various 'relational descriptions' (e.g. *his sister*).

Here again, there are strict rules which determine when the use of such a device is appropriate. If no such rules are incorporated in the text generator, it is possible to generate deviant texts such as:

(8) a We will now present information about it. K. 32 was written by him when the composer was only ten years old.
b We will now present information about the quodlibet. 'Galimathias Musicum' was written by Mozart when he was only ten years old.

In the first text, the pronouns *it* and *him* are used without a proper antecedent, and in both sentences the definite descriptions *the composer* and *the quodlibet* are also used incorrectly. Thus, it is necessary to formulate rules which guarantee the proper usage of such anaphoric devices.

For each type of expression conditions must be formulated which determine their proper use in a text. Apart from pronouns and definite descriptions as discussed above, indefinite expressions and various quantified expressions must be dealt with as well. Plural noun phrases and negation introduce yet other complexities which must be dealt with adequately.

In addition, there are various conditions on the 'distance' between an antecedent and an anaphoric device. The determination of the exact formulation of 'distance' is a complex issue (a definition in terms of the number of preceding sentences is in general too simplistic), but such conditions must be incorporated to achieve the appropriate coherence in a text (see [Grosz et al. 1986], [Dorrepaal 1990]).

At the moment we use a DRT-inspired (see e.g., [Kamp and Reyle 1994]) mechanism to deal with such conditions on the relation between antecedents and anaphora. An important part of the context state (see figure 1) is a *discourse model*. Starting with an empty discourse model, each candidate sentence adds discourse referents and relevant associated information (gender, number, occurrence position, etc.) to this model. Rules for anaphora establish the antecedents for anaphora, and afterwards it is checked whether the resulting discourse model is well-formed (e.g. by checking whether each pronoun has an antecedent, whether definite descriptions have been used appropriately, etc.). If the discourse model is found to be well-formed, the candidate sentence can be used as an actual sentence. If not, a different candidate sentence is subjected to examination, etc.

3.5 VARIATION

We have now discussed how texts are generated, and how their coherency is guaranteed. In passing, we have pointed out how the required variation of texts is achieved, namely by: (1) having a large number of different templates available, in particular several different templates to express the same information, (2) having templates differ in the manner in which they group different pieces of information, (3) supplying a wide variety of expressions to substitute for variable parts, (4) ensuring that the text generator can generate different kinds of texts, varying in detail, length and perspective, and ensuring that the text can be adapted depending on the history.

Another mechanism to make the presentations more interesting relates to content rather than form. Many facts about a composition or about Mozart are known, but cannot be incorporated in a uniform database. In order to be able to convey such facts anyway, we represent them by means of specific templates (the above-mentioned 'gossips').

4 PROSODY AND SPEECH

Text-to-speech conversion is still a notoriously difficult problem, since generating acceptable speech requires syntactic and semantic information that is hard to extract from unannotated text. In the present setting, however, where speech is synthesized on the basis of generated discourse, the task of speech generation is made easier by the availability of syntactic and semantic information. When the generation module outputs a sentence, the generated structure contains all the *syntactic* information that was present in the template from which it results. Moreover, the discourse model (see subsection 3.4) contains *semantic* information about the sentence. More specifically, it says, for each constituent that acts as a filler of a template, what information it is that the constituent refers to. For example, the expression *the composition* may be used to refer to any one of Mozart's compositions. Both kinds of information, syntactic and semantic, are exploited by the next module of the system and the way in which this is done will be stressed in what follows.

The study and manipulation of speech sound are customarily divided into segmental and suprasegmental aspects. We will here focus on the latter, which deals with relations (in temporal structure, or in pitch height, for instance) between speech segments. As with segmental sound, one may distinguish between a phonological and a physical aspect. The first of the two, which is discussed in section 4.2, concerns linguistically relevant questions such as, for example, whether a given word must occur accented or not. The second, which is discussed in 4.3, concerns questions of physical (i.e., phonetic) realization, such as the question of how a given accent must sound: as a Rise or as a Fall in pitch, and at what exact pitch level. Both sections will focus on that part of the problem area where the interplay between syntax and semantics is most clearly illustrated, namely accenting.

4.1 PROSODY

What words in a text are to be accented? At least since Halliday's [Halliday 1967], students of Germanic languages have known that one factor that must be taken into account to answer this question is *informational status*: in these languages, 'new' information is more often stressed than 'given' information. These ideas were later corroborated by experimental research in which it was found that new information must be stressed, while given information may or may not be stressed (see e.g. [Terken and Nooteboom 1987]). Existing speech synthesis systems (e.g. Bell Labs' Newspeak program) have capitalized on this insight, and have managed to improve upon predecessors which stressed all content words, by destressing all words that had occurred in the recent past. Yet, these systems are generally judged as still stressing too many words [Hirschberg 1990]. Consequently, there is still considerable opportunity for improvement.

An attempt was made to improve upon current approaches to accent placement by combining syntactic and semantic information in the following way. As far as semantic information is concerned, we have aimed at a more adequate definition of the notions of given and new information than the ones that were used in other approaches. This definition makes givenness and newness properties not of individual words, but of entire phrases. This definition is then combined with a version of Focus-Accent theory to determine the exact word at which the accent must land. We will first say something about the purely semantic part of the problem, which may be rephrased as the question of which major phrases (more specifically, which slot fillers) are 'in focus', and then the syntactic part of the problem will be discussed, which deals with the question of what word in a phrase that is in focus must be accented. The latter problem area is therefore aptly called that of Focus-Accent theory.

Givenness and newness redefined. Givenness and newness are customarily defined in ways that do not do full justice to the semantic nature of these distinctions: in most approaches, a word is considered given if it is either identical to a word that has already occurred, or a slight morphological variation of such a word. However, inspection of the relevant facts suggests strongly that words of very different forms may cause a word to have 'given' status. For example, the word *wrote* can not only become 'given' due to an occurrence of *wrote*, *write*, etc., but also due to an occurrence of the word *compose*. In addition, givenness is not restricted to individual words. For example, an occurrence of *K.32* or of *this composition* may become 'given', and hence de-stressed (de-accented) due to an earlier reference to K. 32, as when 9a is followed by 9b.

(9) a You have selected K.32.
b You will now hear K.32\this composition.

Thus, semantic theories are natural suppliers of proper definitions of newness and givenness. Moreover, de-stressing and pronominalization occur in roughly the same environments, namely those in which an expression contains 'given' information. This suggests that both may be viewed as reduction phenomena that are caused by semantic redundancy. For these and similar reasons it has recently been proposed that theories of anaphora should be used in accent prediction algorithms. More specifically, [van Deemter 1994] has proposed that a variant of Discourse Representation Theory be used, because these may be viewed as a direct incorporation of the 'givenness' perspective on anaphora.

In the setting of the current system, a part of the context state (figure 2), namely the discourse model, presents itself as a natural candidate to implement this idea, since it contains all the relevant information. In particular, it says, for each referentially used Noun Phrase, whether and where in the discourse the object that it refers to was described earlier. If such an 'antecedent' for an expression is found earlier in the same paragraph, the expression is considered 'given' information (i.e., it is not 'in focus'). If not, it is considered 'new' (i.e., it is 'in focus'). As we will see presently, this is a prominent example of how semantic information forms the input to the module of the system that incorporates Focus-Accent theory.

A version of Focus-Accent theory Focus-Accent theory was first conceived by Ladd and others [Ladd 1980], and later refined by various authors. Our own implementation of Focus-Accent theory, which is sketched below, extends an implementation by Dirksen [Dirksen 1992] (which, in turn, is based on theoretical work by Baart [Baart 1987]), by adding semantic considerations to Dirksen's syntactic account. Interested readers are referred to [Dirksen 1992] and [van Deemter 1994] for specifics.

The basic insight of Focus-Accent is the idea that the syntactic structure of a sentence co-determines its 'metrical' structure. Metrical structure is most conveniently represented by binary trees, in which one daughter of each node is marked as *strong* and the other as *weak*. Metrical structure determines which leaves of the tree are most suitable to carry an accent on syntactic grounds. Roughly, these are the leaves that can be reached through a path that starts from an expression that is 'in focus', and that does not contain *weak* nodes. More exactly, this is the normal procedure: *if* a given major phrase is 'in focus', it is also marked as *accented*, and so is each strong node that is the daughter of a node that is marked as *accented*. Accent is realized at those leaves that are marked as *accented*. However, there may be several obstacles that prevent this from happening. Leaves may end up unaccented in several circumstances, e.g. :

(a) A major phrase is marked $-A$ if it is not in focus.
(b) A leaf x is marked $-A$ if there is a recent occurrence of an expression y which is semantically subsumed by x.
(c) A leaf is marked $-A$ if it is lexically marked as unfit to carry an accent that is due to informational status. (Examples: *the*, *a*, some prepositions.)

Only the first of these cases will be discussed below. The result of an $-A$ marking is that the so-called[1] Default Accent rule is triggered, which transforms one metrical tree into another:

Default Accent rule: If a *strong* node n_1 is marked $-A$, while its *weak* sister n_2 is not, then the *strong/weak* labeling of the sisters is reversed: n_1 is now marked *weak*, and n_2 is marked *strong*.

The Default Accent Rule may cause accent to move to a word where syntactic factors alone would never place it. Consider the little piece of discourse in example (9). In English, it is usually, but not always, the right daughter of a mother node that is *strong*. Thus, the metrical tree looks as in figure 3. Assume that the Verb Phrase is 'in focus' and therefore labeled as *accented*. If semantic factors would not intervene, *K. 32* would carry an accent. But since *K. 32* is also referred to in the previous sentence of the discourse, *K. 32* represents 'given' information (i.e., it is not in focus), and is marked $-A$. As a result, the Default Accent rule swaps the *strong/weak* (S/W) labeling between *hear* and *K. 32* before the *accented* labels are assigned. Consequently, the sentence accent trickles down along a path of *strong* nodes and ends up on *hear*.

This is one example of how the Default Accent

[1] The Default Accent rule was named by Ladd at a time when 'default' could still mean 'alternative'.

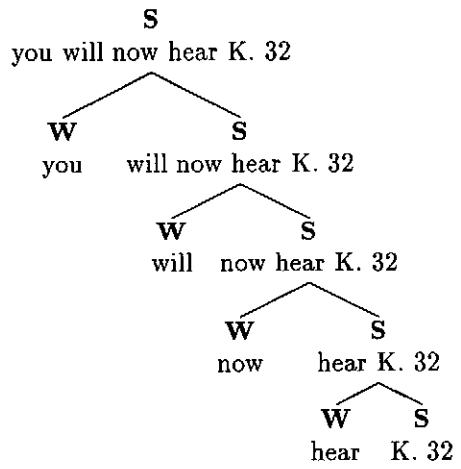


Figure 3: Example of a metrical tree

rule operates, and it illustrates how informational status (in this case: the 'given' status of *K. 32*) cooperates with purely syntactic factors to predict the place of the main accents of a sentence. Before we move on to a brief explanation of the speech module, some shortcomings of the present version of the prosodic module must be mentioned. One obvious limitation is that only informational status has been taken into account as a factor that triggers accent. For example, consider the following discourse:

(10) a You are now going to listen to a string trio and a string quartet.
b The quartet is well-known, but the trio is not.

Here *trio*, in (b), constitutes 'given' information, and yet the contrast with *quartet* makes an accent obligatory. Contrast is one of the factors influencing accent that the current system does not deal with, but that are anticipated for future versions. (Cf. [van Deemter 1994b] for a preliminary study.) Another shortcoming is the limitation to only two (*yes/no*) levels of accenting, whereas experimental results suggest that intermediate values do exist. One area in which these intermediate values might be used is to distinguish between information that is completely new and information that is new yet somehow related to given information, as in so-called bridging phenomena. These limitations notwithstanding, we believe that the current proposal constitutes a modest improvement over earlier accent placement algorithms.

4.2 Intonation

As was explained earlier on, the *Prosody* module of the system adds information about accentuation and phrasing to the output of the language generation module. More precisely, it determines for every word in the monologue whether or not it occurs accented (*notation:* '+' or no marking), and it determines for every word boundary whether it is accompanied by a major prosodic boundary, a minor prosodic boundary, or by no boundary at all (*notation:* '*p2*' or '*p1*' or no marking). For example, the prosodic module may output an enriched sentence such as the following:

> *This sonata for +violin and +piano p1 was written in +Salzburg p2 in +1735.*

This prosodically enriched sentence is then passed on to the speech module, whose job it is to 'realize' this abstract structure in sound. The speech module is not the main topic of this paper, and will be sketched in mere outline.

Ideally, the speech module consists of two independent parts, one of which takes care of segmental information and the other realizes accenting and phrasing. For present purposes, let us assume that all the segmental information is in place[2] and focus on the suprasegmental information, and especially on accenting. Accents are realized in accordance with the IPO model of intonation ['t Hart et al. 1990]. This model has been applied to English [Willems et al. 1988], and this work was used to extract rules for synthesis, which are applied in roughly the following way.

First, the prosodically enriched sentence is transformed into a structure in which the abstract information concerning accenting is 'interpreted' in terms of Rises and Falls. For instance, a sequence of accents is typically represented by a series of so-called 'pointed hats', followed by one 'flat hat', where the 'flat hat' represents the last two accents in the sequence. A 'pointed hat' designates an abrupt Rise in (F_0) pitch, immediately followed by an abrupt Fall, whereas a 'flat hat' has an intermediate phase in which pitch remains equal.

Note that the crucial terms in this representation ('pointed hat', 'flat hat') are intermediate

[2] Ultimately, we envisage a segmental module that makes use of diphones, i.e., recorded transitions between phonemes. The current system, however, makes use of the commercially available DECTALK system, which is based on synthesized formants.

between the linguistic level of prosody on the one hand and the physical level of actual sound on the other, because they abstract away from such physical parameters as absolute pitch height, or speech tempo. Later modules transform this abstract representation into one that is even closer to the physical level, by taking into account that accents are superimposed on a background of pitch declination. For example, a pitch accent that occurs at the beginning of a sentence typically has a higher 'peak' than one that occurs close to the end of the same sentence. Finally, a particular 'speaker' has to be chosen, with its own characteristic pitch range, timbre, etc. It is only at this stage that all the properties of the speech sound have been determined and that the generation of the monologue is finished.

5 Conclusion

In this paper we have outlined a method for generating a large variety of spoken monologues on the basis of information stored in a database. An important aspect of this method is the use of templates, syntactic structures of sentences with variables for which expressions can be substituted. By using templates we avoid some of the complexities that are met by full-fledged natural language systems, while maintaining the flexibility needed for the class of applications we have in mind. The method supports incremental improvement, which can be achieved by increasing the number of variables and the syntactic power of the substitution rules.

We have shown that the use of templates, in combination with maintaining a knowledge state and a context state, makes it possible (1) to generate correct sentences, (2) to combine these sentences into coherent texts and (3) assign an acceptable prosody to the generated sentences.

Although the developed techniques have been applied to the generation of monologues it may be obvious that they can be fruitfully applied in dialogue systems as well.

Acknowledgement: Thanks are due to our colleagues in the DYD project, without whose efforts the techniques described in this paper would not have added up to a working system. Special thanks are due to Steffen Pauws for helpful comments on section 4.2.

References

[Baart 1987] J.L.G. Baart, Focus, Syntax and Accent Placement. Ph.D.thesis Leyden Univ.

[Chomsky 1981] N. Chomsky, Lectures on Government and Binding. Foris, Dordrecht.

[Chomsky 1986] N. Chomsky, Knowledge of Language: Its Nature, Origin and Use. Praeger, New York.

[Dale et al 1992] R. Dale, E. Hovy, D. Rösner and O. Stock, Aspects of Automated Natural Language Generation, Springer, Berlin.

[van Deemter 1994] K. van Deemter, What's new? A semantic perspective on sentence accent. Journal of Semantics 11, pp.1-31.

[van Deemter 1994b] K. van Deemter, Contrast, contrariety and focus. To appear in Proceedings of the Journal of Semantics Conference on Focus.

[Dirksen 1992] A. Dirksen, Accenting and deaccenting: a declarative approach. Proc. of Coling Conference 1992, Nantes, France.

[Dorrepaal 1990] J. Dorrepaal, Discourse anaphora, Proceedings of COLING 90, vol II, p. 95-99.

[Grosz et al. 1986] B. Grosz, K.S. Jones and B.L. Webber, 'Readings in Natural Language Processing', Morgan Kaufman Publishers, Los Altos, Cal.

[Halliday 1967] M.A.K. Halliday, Notes on transitivity and theme in English, in Journal of Linguistics 3: 199-244.

['t Hart et al. 1990] J. 't Hart, R. Collier and A. Cohen, A Perceptual Study of Intonation. Cambridge University Press, Cambridge.

[Hirschberg 1990] J. Hirschberg, Accent and discourse context: assigning pitch accent in synthetic speech. Proc. of AAAI 1990, p.953.

[Kamp and Reyle 1994] H. Kamp and U. Reyle, From Discourse to Logic. Kluwer, Dordrecht.

[Ladd 1980] D.R. Ladd, The structure of intonational meaning: evidence from English. Indiana Univ. Press, Bloomington, In.

[Odijk 1994] J. Odijk, Text generation without planning. Paper presented at CLIN V, University of Twente.

[Rosetta 1994] M.T. Rosetta, Compositional Translation, Kluwer Academic Publishers, Dordrecht.

[Terken and Nooteboom 1987] J. Terken and S. Nooteboom, Opposite effects of accentuation and deaccentuation on verification latencies for 'given' and 'new' information. Language and Cognitive Processes 2, 3/4, pp.145-63.

[Willems et al. 1988] N. Willems, R. Collier and J. 't Hart, A synthesis scheme for British English intonation. J.Acoust. Soc. Am. 84 (4).

THE SPEECH-LANGUAGE INTERFACE IN THE SPOKEN LANGUAGE TRANSLATOR

David Carter and Manny Rayner
SRI International
Cambridge Computer Science Research Centre
23 Millers Yard
Cambridge CB2 1RQ, U.K.

dmc@cam.sri.com, manny@cam.sri.com

ABSTRACT

The Spoken Language Translator (SLT) is a prototype for practically useful systems capable of translating continuous spoken language within restricted domains. The prototype system translates air travel (ATIS) queries from spoken English to spoken Swedish and to French. It is constructed, with as few modifications as possible, from existing pieces of speech and language processing software.

The speech recognizer and language understander are connected by a fairly conventional pipelined N-best interface. This paper focuses on the ways in which the language processor makes intelligent use of the sentence hypotheses delivered by the recognizer. These ways include (1) producing modified hypotheses to reflect the possible presence of repairs in the uttered word sequence; (2) fast parsing with a version of the grammar automatically specialized to the more frequent constructions in the training corpus; and (3) allowing syntactic and semantic factors to interact with acoustic ones in the choice of a meaning structure for translation, so that the acoustically preferred hypothesis is not always selected even if it is within linguistic coverage.

1. OVERVIEW OF THE SLT SYSTEM

The Spoken Language Translator (SLT) is a prototype system that translates air travel (ATIS) queries from spoken English to spoken Swedish and to French. It is constructed, with as few modifications as possible, from existing pieces of speech and language processing software. This section gives a brief overview of the speech recognition and language analysis parts of the SLT system and the philosophy underlying them; for a longer treatment, including details of transfer, generation, and speech synthesis, the reader is referred to Agnäs *et al*, 1994. After the overview, we describe three ways in which the language analyser makes intelligent use of the N-best list of sentence hypotheses it receives from the recognizer.

At the highest level of generality, the design of SLT has two guiding themes. The first is that of *intelligent reuse of standard components*: most of the system is constructed from previously existing pieces of software, which have been adapted for use in the speech translation task with as few changes as possible. The second theme is that of *robust interfacing*. In this paper, we focus on an important means by which robustness is achieved: the delaying of decisions about words, utterances and utterance meanings until sufficient information is available to make those decisions reliably.

The speech recognizer used is a fast version of SRI's DECIPHER [TM] speaker-independent continuous speech recognition system (Murveit *et al*, 1991). It uses context-dependent phonetic-based hidden Markov models (HMMs) with discrete observation distributions for four features: cepstrum, delta-cepstrum, energy and delta-energy. The models are gender-independent and the system is trained on 19,000 sentences and has a 1381-word vocabulary. A bigram language model is used. The output is an N-best hypothesis list, produced using a *progressive recognition search* (Murveit *et al*, 1993) in which the space of possible utterances is pruned by successively

more powerful but more costly techniques. The motivation for this kind of search is to avoid making hard decisions without sufficient evidence, while at the same time maintaining reasonable efficiency.

Fully-fledged linguistic analysis can be viewed from the perspective of the speech recognition task as the final stage of progressive search: the most powerful, most costly techniques used in the system, exploiting complex syntactic and semantic knowledge, are applied, reducing an already fairly limited set of possible utterances to a single choice. Another, equally valid, perspective on language analysis is from the standpoint of *utterance understanding*: the purpose of source language processing in SLT is to map from the acoustic signal to a representation of the utterance meaning, and identifying the correct word sequence is a by-product of this process rather than being a goal in its own right.

Language analysis in SLT is performed by the SRI Core Language Engine (CLE), a general natural-language processing system developed at SRI Cambridge (Alshawi, 1992). The English grammar used for this is a large general-purpose feature grammar, which has been augmented with a small number of domain-specific rules. It associates surface strings with meaning representations in Quasi Logical Form (QLF; Alshawi and Crouch, 1992). Transfer and generation are performed by a second copy of the CLE loaded with a French or Swedish grammar and transfer rules for the appropriate language pair.

The system components are connected together in a pipelined sequence as follows. First, DECIPHER processes the input signal and constructs a list of N-best hypotheses, each tagged with an associated acoustic score; N=5 gives a good tradeoff between speed and accuracy. The construction of this list using the progressive search technique constitutes a thorough pruning of the original search space of all possible word sequences.

The hypothesis list is passed to the English-language version of the CLE, which implements the final phase of progressive search by applying the three processing stages outlined below and described more fully in the remainder of this paper. The CLE achieves robustness in the speech-language interface by postponing the selection of a correct utterance (and utterance meaning) until all available knowledge has been applied. This strategy is made acceptably efficient by the use of a specialized fast parsing technique. The processing stages are these:

- As described in Section 2 below, the CLE examines the hypotheses for evidence of speech repairs, and if it finds any, it adds possible corrected versions to the list without removing the originals, thus postponing a decision about whether the correction is valid or not.

- It then uses the grammar, specialized and compiled for both speed and accuracy as described in Section 3, to analyze each speech hypothesis (original and repaired) and extract a set of possible QLF representations. This typically results in a set of between 5 and 50 QLFs per hypothesis.

- The CLE's *preference component* is then used to give each QLF a numerical score reflecting its *a priori* linguistic (acoustic, syntactic, semantic and, within limits, pragmatic) plausibility. The final score for a QLF is calculated as a weighted sum of the scores assigned to it by a range of preference functions, and the highest-scoring QLF is passed on for transfer and target language generation. We describe the functioning of this component in Section 4 below.

We now move on to examining these stages in more detail, starting with the repair mechanism.

2. DETECTION AND CORRECTION OF REPAIRS

One important way in which spoken language differs from its written counterpart is in the prevalence of self-repairs to speaker errors. Examples such as the following occur in the ATIS domain:

1. `list` *LIST* `FLIGHTS BETWEEN OAKLAND AND DENVER.`

2. `does this` *DOES THIS* `FLIGHT SERVE BREAKFAST.`

3. `COULD I HAVE MORE DETAILS ON FLIGHT d l sixteen` *D L SEVEN* `TWO SIX.`

4. `SHOW ME ROUND TRIP FARES FOR flight two` SORRY *FLIGHT FOUR* `FOUR OH ZERO.`

5. `I WANT A FLIGHT from boston` *FROM DENVER TO BOSTON.*

6. `OK WHAT TYPES OF AIRCRAFT do` *DOES* `DELTA FLY.`

In each case, the reparandum (material to be repaired) is shown in lower case and the repair itself in *ITALICS*, with any explicit repair marker, such as "sorry", shown in **BOLD**. Note that, once the reparandum and any repair marker have been identified, the location of the right hand end of the repair does not affect the interpretation of the sentence (e.g. the repair in (3) could be viewed as "D L seven two six").

In (1) and (2) the reparandum and repair are identical. In (3)-(6) they differ. (3) shows the substitution of a word after the repeated material, (4) shows the use of an explicit repair marker, (5) is an example of the additional material in the repair being inserted, rather than appended, and (6) shows a correction of a suffix, with no strictly identical words occurring.

However, not all repeated word sequences and (possible) explicit repair markers indicate repairs; items (1') to (4') below are non-repairs superficially similar to (1) to (4) above, with (5') providing additional evidence that not all repetitions are repairs. The typographic conventions show how the word sequences might be misconstrued as repairs.

1'. SHOW ME ROUND TRIP FARES FOR U S FLIGHT four *FOUR* oh *oh*.

2'. IS u s *U S* AIR.

3'. ARE ANY OF THE flights *NONSTOP FLIGHTS*.

4'. I WANT a flight with NO *STOPS*.

5'. FROM PHILADELPHIA FROM DENVER AND FROM PITTSBURGH.

It is known that repairs are often indicated acoustically (Bear *et al*, 1992; Nakatani and Hirschberg, 1993) and DECIPHER could be modified to detect possible repair indicators and pass the information on to the CLE. However, this raises some difficult issues of identification, representation and transportability, and it is worth investigating how effectively repairs can be dealt with on the basis of word strings alone.

In line with the philosophy behind progressive search, that of postponing decisions until sufficient information is available, the CLE's repair mechanism has the following novel feature: when a possible repair is located, no immediate decision is made on whether it is genuine. The (putatively) corrected word sequence is added to the N-best list, and given a reduced acoustic score, without the original hypothesis being removed. Thus QLFs can be built from either sequence, and the final choice of a word sequence is a by-product of the choice of a QLF, which, just as for choices between original hypotheses, takes advantage of full linguistic processing of all parts of the sentence.

This methodology allows a range of repairs to be hypothesized by a fairly straightforward algorithm while minimizing the number of false positives found. Given the word sequence actually uttered, it is in general possible to determine the reality of a repair on the basis of (a) specific, fairly superficial knowledge of what kinds of word sequence tend (in the ATIS domain) to indicate repairs, (b) general and ATIS-specific syntactic and semantic considerations, and (c) knowledge of the discourse and reasoning about the domain. In the translation task, a false positive — "correcting" a non-existent repair — is a more serious error than failing to deal with a repair that has occurred, because the former kind of error is likely to confuse the user and to be viewed as much less acceptable. The repair detection algorithm therefore attempts to hypothesize just those possible corrections that seem plausible on the basis of type (a) knowledge and that, if they are false, are likely to be detectable using type (b) knowledge, i.e. by syntactic, semantic and preference processing. Type (c) knowledge is not available within the SLT system.

The detection mechanism identifies possible repairs by first searching for repeated roots in the sentence, i.e. pairs of words (other than numbers, which are often repeated intentionally) that can be analysed morphologically by the CLE as having the same root. Examples are "...flight...flight...", "...do...does..." and "...is...are...". It combines these pairs to identify sequences that begin and end with the same roots, e.g.

I WANT TO GO FROM BOSTON NO
FROM DENVER TO BOSTON ON TUESDAY.

Sequences that have intervening material and consist only of one of a set of very common words ("a", "and", "from", "in", "of", "or" and "to") are discarded at this point, as inspection of the data suggests they are likely to lead to false positives. In all other cases, however, the two sequences (underlined above) are first matched from left to right. Two points are awarded for a shared root, and one is deducted when a word is skipped in either sequence. The match proceeds

(by dynamic programming) so as to maximize the score. In the example, two points are awarded for the matches on each of "from" and "Boston" and one is deducted for skipping each of "Denver" and "to".

If there is no intervening material, the match is now complete, and the hypothesized repair is returned. If there is intervening material (as with "no" above) it may form part of either the repair or the reparandum. Similar, but more general, matches are therefore carried out in both the forward and the backward directions.

The forward match begins at the start of the intervening material and just after the end of the second repetition sequence, i.e., at "no" and "on" in the example, and continues forwards until all the intervening material is consumed. The backward match starts at the end of the intervening material and just before the start of the first sequence, i.e. at "no" and "go", and words backwards. One point is deducted for skipping a word in either sequence, unless the match is forward and the word is known to be an explicit repair indicator, in which case a point is awarded. (Explicit repairs are counted only in the forward match to ensure they are identified as material to be deleted). Two words match each other, with no adjustment in the score, only if they share a major category.

Once all matches have been completed for all possible pairs of root sequences, the best one(s) are selected. Higher-scoring matches are preferred, with those involving the deletion of fewer words being favoured when scores are equal. If there are non-overlapping repairs (e.g. "I want to go from from Boston to San San Francisco") then the best options for both are accepted.

In the example above, the best path is for the forward match. It consists simply of recognizing "no" as a repair indicator and not progressing the second pointer at all. This gives a reparandum of "from Boston no" and a repair of "from Denver to Boston" with a total score of three.

On the main training corpus of 4615 reference sentences used during the project, the repair mechanism suggested corrections for 135 sentences. As far as could be determined by inspection of the word string alone, 89 of these actually were repairs and 41 were not, with the status of five being impossible to determine without reference to prosody. The subsequent behaviour of the system for the 130 sentences whose status was clear was as shown in Table 1. Correct decisions are shown in bold type.

	Actual repairs	False alarms
No QLF found	10	4
Right repair chosen	**77**	-
Wrong repair chosen	1	2
Non-repair chosen	1	**35**
Total	89	41

Figure 1: Decisions on possible repairs

Restricting attention to sentences for which some QLFs were found, of the 79 sentences involving repairs for which a QLF for a repaired version was chosen, the repaired string was correct, or as plausible as any other choice, in 77 cases. In the other two cases, a wrong repair, and no repair, were selected respectively. When no repair was actually present, the preferred QLF was for the unrepaired version in all but 2 of 37 cases. Thus the repair mechanism caused 77 sentences to receive an analysis for the correct string where this would not otherwise have happened, and caused 2 sentences to receive a bogus interpretation when they would not otherwise have received one. In other words, it increased coverage on the training set by (77-2)/4615, or 1.6%.

Of course, performance on reference versions (corresponding to perfect speech recognition) of training sentences is likely not to be a good indicator of performance on errorful recognizer outputs for unseen sentences; and in fact, applying the current repair mechanism to such outputs does tend to result in the acceptance of noticeably more bogus repairs, nearly all arising from incorrect sentence hypotheses. As already indicated, this is quite undesirable.

However, many if not most errors of this type are due to the fact that the repair mechanism is being applied to a qualitatively different kind of data from that used to guide its design. We are encouraged by the fact that, for the reference sentences, a relatively simple repair *suggestion* algorithm can lead to such accurate decisions on the validity of the repair by the much more sophisticated subsequent language processing (only 4 wrong choices of string out of 116 cases where a choice was made). Further work will involve redesigning the algorithm, and probably training it automatically, to handle the kinds of output characteristic of the recognizer. As Section 4 below will argue more fully, training language processing decisions on typical recognizer behaviours

rather than only on reference sentences can enhance decision-making considerably.

3. GRAMMAR SPECIALIZATION FOR FAST PARSING

Language models used in the context of speech recognition are normally some variety of finite-state grammar. Bigram grammars are probably still the most popular choice, and one is used by the version of DECIPHER incorporated in SLT. Trigram or higher N-gram models and stochastic context-free grammars are also fairly common. The advantages of finite-state models are well-known: they are fast, robust, and easy to train. The disadvantages are also clear: viewed as grammatical formalisms, they are insufficiently expressive to capture many important types of linguistic regularities, and so although they are useful in the non-final stages of the progressive search task, they are not adequate for the final stage, nor indeed for constructing a sufficiently rich semantic representation to support translation.

However, use of more powerful and expressive grammar formalisms tends to be impractical due to the excessively slow processing times associated with most known parsing algorithms. This would be especially problematical in the SLT system when the language analysis carried out by the CLE counts as a single, final stage of progressive search, so that many possibilities are considered before any are ruled out.

In the language analysis part of the SLT system, we have therefore implemented what we think is an interesting compromise between the opposing positions of fast finite-state language models and general linguistically-motivated grammars. The bulk of this work (most of which has carried out in collaboration with Christer Samuelsson of SICS, Stockholm) has been reported elsewhere (Rayner, 1988; Rayner and Samuelsson, 1990; Samuelsson and Rayner, 1991; Samuelsson, 1994). We content ourselves here with a brief summary relating it to the themes of the present paper.

We start with a general, linguistically motivated grammar, which has been given enough specialized vocabulary to have good domain coverage. In the SLT project, we used the CLE grammar for English (Alshawi, 1992, chapters 4 and 5; Agnäs *et al*, 1994, chapter 7), but the techniques do not make any special use of its peculiarities, and would be applicable to any general unification-based phrase-structure grammar. The key point is that the general grammar is unsuitable for the language modelling task because it is *over*-general; in particular, there is no need in the context of a normal spoken language domain to have a fully recursive grammar.

We *specialize* the grammar to the domain by first using it to parse a substantial corpus of examples (in the concrete experiments carried out, we used a set of about 5000 ATIS sentences). We then extract a much simpler grammar from the original one by cutting up the analysis trees from the parsed corpus into sub-trees, where each sub-tree corresponds to a linguistic "chunk" or unit; we used only four chunk types (utterance, noun phrase, non-recursive noun-phrase and preposition phrase), compared to about twenty-five different phrase types in the original grammar. The rules contained in each sub-tree are then "collapsed" into a single rule for the appropriate chunk-type, using the so-called Explanation-Based Learning algorithm (van Harmelen and Bundy, 1988; Hirsh, 1987). With a suitable choice of chunk-types, we can produce a specialized grammar whose rules correspond to chunk patterns occurring in the training corpus.

By construction, the specialized grammar has strictly less coverage on the domain than the original one. Our experiments suggest, however, that given a substantial training corpus the loss of coverage is on the order of a few percent at most. This loss of coverage is more than counterbalanced by the greatly simplified structure of the specialized grammar, which can be parsed nearly two orders of magnitude more quickly than the general one, using an LR parsing algorithm (Samuelsson, 1994). The gain in speed is due to the fact that the grammar, after specialization, is nearly finite-state; we have in effect automatically squeezed a general grammar into a finite-state format, after cutting off the few pieces that refuse to fit.

Apart from the enormous gain in speed, it is also worth noting that the specialized grammar is less ambiguous than the general one; for a given sentence, it normally produces substantially fewer different analyses. This implies that the task of identifying a correct analysis becomes correspondingly simpler. The "preference component" described in the next section has less work to do, and makes incorrect choices less often. In practice, we have discovered that this extra accuracy more or less cancels out the loss of grammatical coverage; the few sentences outside specialized

grammar coverage tend to be so complex and ambiguous that there is a high chance of an incorrect analysis being preferred.

4. DISAMBIGUATION

Once zero or more QLFs have been produced for each of the original and repaired sentence hypotheses in the N-best list, the preference component of the CLE has the task of selecting the most appropriate one for translation. It does this by assigning a score to each QLF and selecting the highest-scoring one, as we will now describe. A full account is given in Alshawi and Carter (1994).

4.1 Preference Functions

The score assigned to a QLF is a scaled linear sum of the scores returned by a set of about twenty individual *preference functions*. Preference functions are of three types.

- Firstly, there is a *speech* function which simply returns the acoustic score for the sentence hypothesis that gave rise to the QLF (or a default low score if the hypothesis was suggested by the repair algorithm).
- Secondly, *structural* functions examine some aspect of the overall shape of the QLF. Typically, the number of occurrences of some relatively unlikely type of grammatical construction is counted, so that readings which contain instances of it can be penalized relative to those that do not.
- Thirdly, *combining* functions collect instances of linguistic objects such as: N-grams in the underlying word string; the syntax rules used to create the QLF; and triples of the form (H_1, R, H_2), where H_1 and H_2 are the head predicates of QLF substructures representing words or phrases in the sentence and R indicates the relationship (e.g. a preposition or an argument position) between them. Semantic classes are used to group place names, numbers and other sets with similar distributions. For example, the set of triples for the correct analysis of "Show me the flights to Boston" includes these:

```
(show_CauseToSee,3,flight)
(flight,to,*place)
```

the second of which indicates the attachment of "to Boston" to "flights" rather than to "show". The combining function calculates, by addition or averaging, a score for the QLF based on the scores for the individual objects. The objects in turn take their scores from the pattern of their occurrence in correct and incorrect QLFs observed in training on recognizer outputs on a corpus for the domain in question. Roughly, an object score is intended to be an estimate of the log probability that a QLF from which the object arises is the correct one.

4.2 Scaling Factors

The scaling factors used to derive a single summed score for a QLF from the scores returned for that QLF by the various preference functions are also trained automatically in order to maximize of the chances of the highest-scoring QLF being correct. Scaling factor training has two phases.

The first phase makes use of a measure of the similarity between each QLF for a sentence and the correct QLF (selected in advance by interaction with a developer) for that sentence. This measure is sensitive to differences both in the underlying word sequences and in the groupings of the words into phrases by the QLFs. Linear (least squares) optimization is carried out to find the scaling factor values that make the preference scores for QLFs resemble the similarity measures as closely as possible. This is an analytic process that can be carried out fairly quickly. However, its objective function, that of modelling similarity to the correct QLF, is only approximately related to the behaviour we want, that of ensuring that the correct QLF is placed first in the preference ordering, regardless of the scores of incorrect QLFs relative to each other.

In the second phase, therefore, scaling factors are adjusted iteratively to increase the number of training sentences for which the correct QLF gains the highest score; that is, attention is focused on selecting correctly among the few most plausible QLFs, and not on predicting the scores of clearly implausible ones, whose relative merits are unimportant. Since this task is non-linear, it is fairly computationally intensive, and may only find a local optimum, so that the first, linear phase is essential to find a good starting point for it.

After scaling in this way, the preference functions are able to select the correct QLF (as judged by an expert) in 90 to 95% of cases when

trained on four fifths of a corpus of the reference versions of 4092 within-domain, within-coverage ATIS sentences of up to 15 words in length and tested on the other one fifth, with each one fifth being held out in turn for testing. This result is for the QLFs produced with a version of the grammar that had not undergone the specialization process described in Section 3 above. The figure would be still higher if only the smaller number of QLFs arising from the specialized grammar were compared. Thus, as remarked earlier, the tendency of grammar specialization to reduce coverage slightly is largely offset by the fact that, for sentences that are still in coverage, fewer erroneous QLFs are produced which may be preferred over the correct one.

4.3 A Comparison

To appreciate the importance of some of the points in the above description, it is instructive to compare the process described above with the somewhat simpler training procedure used in an earlier version of the system. For clarity, we will call the earlier version SLT-0, and current version, implementing the above procedure, SLT-1. SLT-0 lost some accuracy because in it, the various scores and scaling factors were optimized for tasks related to, but not identical to, that encountered at run-time.

The first difference is that in SLT-0, the linguistic objects used by some of the combining metrics were scored not by comparing good and bad QLFs but on the basis only of their frequency of occurrence in good QLFs. As we will see in the next section, this is suboptimal, because an object is not a good predictor of correctness simply because it occurs frequently in good QLFs; it may occur just as often in bad ones.

SLT-0's second drawback was that training with respect to the corpus was decoupled from training with respect to the speech recognizer. That is, object scores and all the non-speech scaling factors were calculated by looking only at QLFs for the reference versions of corpus sentences, and not at recognizer outputs. The scaling factor for the speech function was then found by trial and error on a separate training corpus. Thus SLT-0 had no opportunity to adapt to and compensate for typical recognizer errors.

QLF selection accuracy turned out in fact to be relatively insensitive to the value of the acoustic factor, which can be doubled or halved without noticeable effect. However, the lack of training on incorrect sentence hypotheses was a more serious drawback. There are syntactic and semantic patterns which seldom occur in analyses of correct sentence hypotheses and therefore were not assigned very large scores, but which often crop up as a consequence of certain kinds of recognizer error. A known example of this behaviour is number disagreement between subject and predicate in a sentence hypothesis with main verb "be", for example "What is the first flights to Boston?". This is grammatically possible but most unlikely to be correct, and usually indicates that the head noun of the predicate phrase has been recognized with the wrong number: in the example, the word spoken would actually have been "flight". There are also examples of semantic triples, and perhaps also syntax rules, which likewise tend to characterize analyses of wrong hypotheses but which, for that very reason, are not observed when training only on correct word strings. It is not sufficient to finesse this problem by penalizing objects only observed infrequently in training on reference sentences, because there is no *a priori* way of knowing whether such an object, when encountered at run time, indicates a recognizer error or just an unusual, but genuine, form of words. In the next section, we focus in more detail on this problem and how it is is overcome.

4.4 Tuning to the N-Best Task

The deficiencies just described for SLT-0 had the effect that selecting a sentence hypothesis using the trained combination of speech, structural and combining preference functions only yielded a 2% increase in sentence accuracy (as measured on a 1000-sentence unseen test set) over using the speech score alone. This figure is in a sense misleadingly pessimistic, since we are interested in translation rather than recognition *per se*, and the combined functions always select a hypothesis for which a QLF, and therefore potentially a translation, is found, whereas the recognizer alone sometimes prefers an unanalysable string, which even if correct will not be translated. Nevertheless, it seemed likely that introducing linguistic factors, if done optimally, should improve sentence accuracy by more than a couple of per cent.

We therefore carried out some experiments (reported in full in Rayner *et al*, 1994) in which several preference functions were trained on N-best data as in SLT-1, but with sentence hypothesis selection, rather than QLF selection, as the objective. The value of N was chosen to be 10, rather

than 5 as in the run-time system. The preference functions used were:

- The speech function, returning the recognizer score.
- Two structural functions: one which returned 1 if any QLFs were found for the sentence using the specialized grammar, and otherwise 0; and one which returned 1 if the best QLF for the string (as judged by the existing preference module) contained a subject-predicate number mismatch, and otherwise 0.
- Two combining functions: one for grammar rules used in the best QLF for the string, and one for the semantic triples for that QLF.

We found that over the 1000-sentence test set, the optimized combination of functions selected the correct hypothesis 70.5% of the time, compared to a maximum possible 84.2% where the correct hypothesis occurred at all in the 10-best list, a score of 66.3% for the speech function alone, and a score of 67.8% for the more traditional approach of selecting the first hypothesis in the recognizer list that received a parse. Thus the optimized combination gave an absolute improvement over the speech function alone of 4.2%, double the corresponding figure for SLT-0. For sentences of 12 words and under, the improvement was 5.6%. These sentences showed a larger improvement because they were more likely to be analysable by the CLE; if no QLFs are produced for any hypothesis, the linguistic functions have no contribution to make. Because of this drawback, it turned out that N-gram combining functions for N=1 to 4, which can be applied even when no QLFs are produced, were slightly more powerful in combination with the speech function than the CLE-based functions were, although for 12-word sentences and the acceptable variant criterion, no difference was apparent. Not surprisingly, when N-gram knowledge sources were added, a still better result, 73.7%, was obtained.

We concluded from these results that it is well worth training linguistic functions in this way. One further possible improvement is that for sentence recognition (although probably not for translation, because of the risk of errors), it would also be desirable to derive QLF analyses of parts of a sentence when no full analysis could be found; this would allow linguistic functions always to make some contribution, even if only an imperfect one, and would improve accuracy on utterances for which no hypothesis was perfectly correct and those which included constructions outside the coverage of the grammar.

5. SUMMARY AND CONCLUSIONS

We have described the ways in which language analysis in SLT makes intelligent use of the N-best hypothesis list delivered by the speech recognizer, implementing the final stage of progressive search by avoiding nearly all hard decisions about word identities or sentence meanings until all available linguistic knowledge has been applied. That is, the CLE creates its whole search space before pruning away any of it. Thus alternative QLF analyses for the same recognizer hypothesis, for different recognizer hypotheses, and for repaired as well as unrepaired versions of hypotheses are all constructed and compared in a uniform way. The use of an automatically tuned grammar and associated fast parser makes this generate-and-test process acceptably fast (typically a few seconds per speech hypothesis on a SPARCstation 10) by eliminating many impossible search paths and some possible but unlikely ones.

It would be possible to speed up the system further by parallelizing it. Each recognizer hypothesis could be analysed separately, and the highest-scoring QLF (if any) resulting from it returned for a final choice to be made.

The unattainable ideal in any search problem is for the search space constructed to consist only of the correct solution, or of solutions that are equally likely to be correct. We approximate this ideal in the speech understanding task by training and selecting grammar rules (the objects that generate possible solutions) on human-transcribed reference material, so that, as far as possible, correct solutions will fall within the search space and incorrect ones will fall outside it. In practice, of course, by no means all incorrect solutions will be excluded in this way; so we train preference functions on recognizer and language analysis output, to maximize our chances of distinguishing correct from incorrect solutions, whatever stage of processing they arise from.

In Section 4.4 above we gave performance details for speech and language analysis. Sentence recognition accuracy using optimized speech (DECIPHER) and language (CLE and N-gram) information on unseen ATIS data is 73.7%. Full details of the performance of an earlier version

of the full system (roughly SLT-0) are given by Rayner *et al*, 1993. Briefly, however, for sentences within the ATIS domain and up to twelve words in length, if a correct speech hypothesis is selected then a Swedish translation is produced on about three occasions in four, and 90% of those translations are acceptable. The remaining 10% can nearly all be clearly identified by the hearer as errors because they are ungrammatical or unnatural; divergences in meaning, which might lead to more serious forms of dialogue failure, are extremely rare.

ACKNOWLEDGEMENTS

The bulk of the work described here was done on a project funded by Telia Research AB. The partners were SRI International, Menlo Park (speech recognition); SRI International, Cambridge (language processing software and English grammar); the Swedish Institute for Computer Science (English-Swedish transfer rules, Swedish grammar and fast parser for specialized grammar), and Telia Research (speech synthesis). Adaptation to French was carried out by ISSCO, Geneva, in collaboration with SRI Cambridge.

REFERENCES

Agnäs, M-S., and 17 others (1994). *Spoken Language Translator: First Year Report.* Joint report by SRI International (Cambridge) and SICS. Order from `preben@sics.se`.

Alshawi, Hiyan, editor (1992). *The Core Language Engine.* The MIT Press, Cambridge, Massachusetts.

Alshawi, Hiyan, and David Carter (1994). "Training and Scaling Preference Functions for Disambiguation". *Computational Linguistics*, 20:4.

Alshawi, Hiyan, and Richard Crouch (1992). "Monotonic Semantic Interpretation". In *Proceedings of 30th Annual Meeting of the Association for Computational Linguistics*, pp. 32–39, Newark, Delaware.

Bear, J., J. Dowding, and E. Shriberg (1992). "Integrating multiple knowledge sources for the detection and correction of repairs in human-computer dialog". In *Proceedings of 30th Annual Meeting of the Association for Computational Linguistics*, pp. 56–63, Newark, Delaware.

van Harmelen, Frank, and Alan Bundy (1988). "Explanation-Based Generalization = Partial Evaluation" (Research Note), *Artificial Intelligence 36*, pp. 401–412.

Hirsh, Haym (1987). "Explanation-Based Generalization in a Logic-Programming Environment", *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, pp. 221–227.

Murveit, Hy, John Butzberger, Vassilios Digalakis, and Mitch Weintraub (1993). "Large Vocabulary Dictation using SRI's DECIPHER(TM) Speech Recognition System: Progressive Search Techniques". In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pp. 319–322, Minneapolis, Minnesota.

Murveit, Hy, John Butzberger, and Mitch Weintraub (1991). "Speech Recognition in SRI's Resource Management and ATIS Systems". In *Proceedings of the 4th Speech and Natural Language Workshop.* DARPA, Morgan Kaufmann.

Nakatani, C., and J. Hirschberg (1993). "A speech-first model of repair detection and correction". In *Proceedings of 31th Annual Meeting of the Association for Computational Linguistics*, pp. 46–53, Columbus, Ohio.

Rayner, M. (1988). "Applying Explanation-Based Generalization to Natural-Language Processing". *Proceedings of the Conference on Fifth Generation Computer Systems*, Tokyo.

Rayner, M., and C. Samuelsson (1990). "Using Explanation-Based Learning to Increase Performance in a Large-Scale NL Query Interface". *Proceedings of the 3rd DARPA Speech and Natural Language Workshop*, Hidden Valley.

Rayner, M., and 11 others (1993). "Spoken Language Translation with Mid-90's Technology: a Case Study". *Proceedings of Eurospeech-93*, Berlin.

Rayner, M., D. Carter, V. Digalakis and P. Price (1994). "Combining Knowledge Sources to Reorder N-Best Speech Hypothesis Lists". *Proceedings of the 1994 ARPA Workshop on Human Language Technology*, Princeton.

Samuelsson, C., and M. Rayner (1991). "Quantitative Evaluation of Explanation-Based Learning as a Tuning Tool for a Large-Scale Natural Language System". *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, Sydney.

Samuelsson, C. (1994). *Fast Natural Language Parsing Using Explanation-Based Learning*, PhD thesis, Royal Institute of Technology, Stockholm.

Time Synchronous Chart Parsing of Speech Integrating Unification Grammars with Statistics*

Hans Weber
IMMD 8
University Erlangen-Nurnberg
Germany
email: weber@faui80.informatik.uni-erlangen.de

Abstract

We present an active chart parser which parses left connected wordgraphs in a strictly time synchronous way. The parser performs a beam search on the possible paths through the word graph and on the possible derivations of the unification grammar simultaneously. A metric is given to assign scores to edges, taking into account the whole left context thereby combining acoustic probabilities, n-gram probabilities and unification grammar probabilities. A specialized model for the derivation of typed unification grammars is introduced. Different ways of coupling the parser with an LR beam decoder in an online time synchronous fashion are defined and several experimental results are presented. Two top down and one bottom up method are investigated. In bottom up mode, the decoder sends word hypotheses as they are found from left to right, while the parser keeps step. In verify mode, the decoder is always a frame ahead, while the parser verifies received hypotheses, providing language information to the decoder. In predict mode, the parser is a frame ahead, sending possible successor information to the decoder. The latter uses this information to restrict its search space. Finally a method to maximally reduce multiple paths in a left connected word graph produced by a beam decoder ist presented, which can be used in all of the three strategies.

*This work was funded by the German Federal Ministry for Research and Technology (BMFT) in the framework of the Verbmobil Project under Grant BMFT 01 IV 101 H / 9. The responsibility for the contents of this study lies with the author.

1 Introduction

This article gives an overview of ongoing research in the area of time synchronous processing in speech understanding. Here we are mainly concerned with the design of a chart parser for that issue. The parser which we call a *Left-Right Incremental Active Chart Parser* (LR-ACP) has the following features specific to the task.

- All word hypotheses are processed simultaneously in one chart from left to right.
- The parser uses a typed unification grammar the derivations of which are subject to a probabilistic model.
- Agenda items and edges are supplied with a combined normalized score of acoustic, bigram and grammar model probabilities.
- The parser performs a beam search implemented on the agenda, thereby constraining forward and backward search.
- Extensions of the algorithm for time synchronous interaction with a one pass beam decoder are implemented.
- Receiving the decoder's word ending hypotheses frame by frame, a mapping of hypotheses belonging to the same word copy is performed during parsing.

The ideas which led to this work are based, besides others, on works like Shieber 85 [19], Görz 88 [9], Ney 93 [16], Päseler 88 [17], Fujisaki et al. 91 [8]. For a more detailled description see Weber 94 [20]. Some of the work concerning the

coupling with the beam decoder is a joint work with Andreas Hauenstein[1] and has already been published in Hauenstein & Weber 94 [10, 11].

2 LR-ACP with Beam-search on the Agenda

The use of active chart parsers and extensions of these in parsing speech has tradition. Our LR-ACP can be seen as a consequent next step to the work of Päseler 88 [17].

2.1 Basic Operations

The basic data structures of an active chart parser can be given as follows:

Definition 2.1 *Basic data structures of an ACP*

Vertices: *A chart is a directed acyclic graph, whose vertices are totally ordered. Vertices correspond to word boundary hypotheses.*

Edges: *An edge consists of a pair of vertices, a dotted rule, a (record of) score(s) and a couple of book-keeping information as there a are the covered string of word hypotheses, pointers to daughter edges and also pointers to the left and right hand sides of rule instances, aso.*

Agendas: *For every vertex i there exists an $agenda_i$, keeping triples of actives, inactives and scores, such that the end vertex of the inactive of a triple is vertex i. Agendas are accessed best first.*

We use the standard active chart parser operations, adapted to unification grammars to drive the LR-ACP. For an edge A, we write selectors for the components of A, using A.from and A.to to refer to the begin and end vertices of A, A.mother to the left hand side of A's rule, A.next to the category to the immediate right of the dot, aso.[2] We write $\tilde{\cup}$ for the relation of being unifiable, and $\cup$ for the unification operation.

Definition 2.2 *Basic operations of an ACP*

SEEK-UP: *If an inactive A is inserted, for every rule $\alpha \rightarrow \beta$, such that A.mother $\tilde{\cup}$ first(β), insert B, with B.rule := $\alpha \rightarrow .\beta$, B.from := A.from, B.to := A.from, if B does not already exist.*

SEEK-DOWN: *If an active A is inserted, for every rule $\alpha \rightarrow \beta$, such that A.next $\tilde{\cup}$ α, insert B, with B.rule := $\alpha \rightarrow .\beta$, B.from := A.from, B.to := A.from, if B does not already exist.*

FUNDAMENTAL RULE: *For a pair of active A and inactive B, if A.next $\tilde{\cup}$ B.mother, insert C with C.from := A.from, C.to := B.to, C.rule := copy(A.rule), C.next := A.next. Perform the following actions: $\cup$ copy(B.mother),C.next. Shift C.next one position.*

PUSH-EDGES: *When edge A is inserted into the chart, then if A is active, push all pairs (A,B) to agenda, such that B is inactive and A.to = B.from. Else, push all pairs (B,A) to the agenda, such that B is active and B.to = A.from. Insertion is binary according to the score of (A,B) and (B,A).*[3]

The LR-ACP can be driven top down using the SEEK DOWN operation as well as bottom up with the SEEK UP operation, as usual active chart parsers can.

2.2 Efficiency Matters

Efficiency of the basic operations is really a challenge, since the search space given by a couple of thousand word hypotheses is huge and in no way comparable to the easy task of parsing deterministic input.

In the specification above, the SEEK DOWN (and SEEK UP) operations cost a lot of unifications, although no top down restriction is propagated during the rule insertion in order to keep the rule instances finite. In order to achieve efficiency with the unification grammar, we preprocessed and hashed all SEEK-DOWN operations besides the redundancy check.

By testing for unifiability only, but inserting the original rule objects, we can keep the instances of empty active edges finite.

In the preprocessing step, we take every daughter of every rule and compute the series

[1] University Hamburg, Natural Language Dept.

[2] We hope to meet the intuitions of the reader, since we do not explain all notational details.

[3] So the agendas are always kept sorted.

of successor rules possibly being inserted by a SEEK-DOWN operation. In this preprocessing, we use only the type skeleton of the grammar and (nondestructive) type unification in a first pass. A local reduncancy check ends up recursion.

The series produced is filtered afterwards, using all of the features and propagating them through the series.

All rules and daughters have integer codes assigned, so the table lookup as well as the redundancy check in the chart become cheap.

Our treatment of rules can be seen as a variant of Shieber 85's *Restriction* mechanism [19], based on types.

As a result of this technique, there are no more unifications involved in predictor or scanner operations in Earley's terms. Only during the completer step unifications are actually carried out.

The same technique cannot be used when parsing bottom up. The trigger for a SEEK UP operation are inactive edges, which may have infinitely many extensions of feature structures. Nevertheless, the number of types the feature structure (FS) can have is finite. So we again use the type skeleton of the grammar to do a preprocessing. Anyway, we leave out the filtering step in the preprocessing of the SEEK UP operation, accepting a slight overgeneration of rules to be inserted. These additional edges will fail in the completer operation, consuming a part of those full unifications which would have to be done without a preprocessing anyway.[4]

2.3 Initial Local Agendas and Beams

To achieve an incremental time synchronous behaviour we propose the following control loop, where BEAMWIDTH must be given in advance.

1. Create Vertex V_0. Insert active start edge. Set T := 1.
2. Create V_T with agenda$_T$. Read all word hypotheses W which end at T, and insert edges for W into the chart.
3. Set BEAMVALUE := score(top(agenda$_T$)) - BEAMWIDTH
4. Apply fundamental rule to pop(agenda$_T$), until score(top(agenda$_T$)) $\leq$ BEAMVALUE.
5. Save agenda$_T$.
6. Increment T. Goto 2.

This pseudo code implements a beam search directly on the agenda. There are a couple of reasons for this strategy.

First of all, an agenda$_T$ in a cycle T, right after the insertion of edges due to the reading of new hypotheses consists only of entries involving pairs of actives and word hypotheses. We call the agendas in this state of processing *Initial Agendas*. An *Initial Agenda* encodes the possible future completer operations in form of their initial pair of active edge and word hypothesis.

Since the completer step is the expensive operation in Earley's Algorithm anyway and especially in our handling of unification operations, we prune on agenda items.

Further on, since we use acoustic, n-gram and grammar scores in combination, we need a representation where global scores can be compared or at least, where good estimations of global scores of paths are admissible to apply pruning.

By virtue of the INSIDE and OUTSIDE scoring of edges described in section 3 below, we can assign an optimistic estimate of a path's global score to an active edge not yet covering that path. As a consequence, the maximum score of the initial agenda is the maximum of all possible scores arising from successive completer actions.

In step 5 of the algorithm above, we save those entries of a local agenda which fell below the beam. Since we use a combination of symbolic and probabilistic restriction, we can never guarantee that one of the paths inside the beam will result in a valid analysis of some spanning sequence of word hypotheses. In order to make the system robust, a second search phase can be initiated, parsing the global agenda of pruned entries when the time synchronous search failed. When there is a spanning analysis, it will be discovered then.

3 Metrics

This section is concerned with the combination of probabilities coming from different models and the partition of combined scores into an INSIDE score and an OUTSIDE score.[5]

[4] The preprocessing seems to favorize a top down parsing schema.

[5] Although these two scores correspond more or less to Päseler's IS and AS ([17]), we take the names from

3.1 Combining Scores

Different models used in one system tend to have different numbers of parameters and to be trained on different data. So, taken seriously, they cannot be combined.

In practice we overcome this problem by adjusting the magnitude of the scores involved. This is done by introducing weights relating of the scores coming from different models. We use log probabilities, since the overall search is Viterbi like, maximizing rather than summing up.

$$\text{CS(W,D)} = AS(W) + \gamma NGS(W) + \delta GS(D) \quad (1)$$

Acoustic score (AS), bi-gram score (NGS) and grammar score (GS) are just added, γ and δ regulating the relative weights. W stands for an utterance and D for a possibly partial grammatical derivation of it.

The single scores themselves are normalized by the number of operations in order to be able to compare word hypotheses of different length, utterances made up of a different number of words and trees made up of different amounts of grammatical operations.[6]

$$AS(W) = \frac{logP(W|s_{i,j}, \text{hmm})}{j - i} \quad (2)$$

$$NGS(W) = \frac{logP(W|\text{bi-gram})}{\text{words}(W) - 1} \quad (3)$$

$$GS(D) = \frac{logP(D|\text{gm})}{\text{rules}(D) + \text{typeshifts}(D)} \quad (4)$$

The acoustic score for an utterance W is given by the probability assigned by the acoustic model to the signal from frame i to frame j, the portion spanned by W, divided by the number of frames spanned. Analogously we define the bi-gram score and the grammar score, the latter being normalized by the number of rule applications and type shifts, the items of observation of the grammar model *gm* (which will become more transparent in section 4).

This is not the only possible way to combine scores. Another method, first summing weighted probabilities, and normalizing them afterwards by the sum of all operations did not show improvements. The behavior of the system is more sensitive to the choice of values for the weights. It turned out that it was really difficult to find good settings by hand. While it was comparably easy to tune only one weight[7], using only acoustic and n-gram scores in the system, things get more complex when additional models are involved. Although we did not yet test it, we feel that the weights should be subject to an optimizing hill climbing procedure.

For the counters for operations and the absolute log probabilities we write AO, NGO, GO, AP, NGP and GP respectively. Edges in fact carry a record with fields for these single values, since the log probabilities and operations are kept seperate for edges. This makes it easier to compute new values for new edges. Combined scores are only computed for comparisons which happen to be done only on agenda items.

3.2 Inside and Outside Scores

Similar to Päseler 88 [17], we give two different scores to edges. The first one is the INSIDE score of an edge — the score coming from the portion spanned by that edge. Roughly speaking the OUTSIDE score is the cumulated score of those edges in a left context which were leading to the introduction of a certain edge.

In top down mode the OUTSIDE score of an edge i is the best score of some edges j to m, where j to m lead to the introduction of edge i plus the INSIDE score of i.[8] Generally the edges j to m leading to an edge i are spanning the portion from frame 0 to the beginning of i.

In bottom up mode, where empty actives are introduced on the basis of inactive edges with the SEEK-UP operation, INSIDE and OUTSIDE score of an edge fall together. There is no left context for an edge to be determined from which we could say it led to the introduction of that edge.[9] So the following is on the building of the OUTSIDE score in the case of top down parsing.

We define the INSIDE score as well as the OUTSIDE score for the start edge directly below, without referring to the single components due to space limitations.

Definition 3.1 *INSIDE score for an edge E,*

Baker's [2] Inside-Outside-Algorithm, which we think was the first idea in that direction.

[6] We write X(Y) for the number of operations of type X used to produce Y.

[7] The experiments in Hauenstein & Weber 1994 were done only with acoustic and n-gram scores.

[8] ..supplied with the transition penalties between m and j.

[9] This is not really true, since we could use top down filtering (as described by Wirén [21]) during bottom up parsing.

from i to j, spanning string W with analysis D_W

$$\mathrm{IS}(E_{i,j,W,D_W}) = \begin{cases} CS(W, D_W) & i < j \\ \delta GS(E.rule) & i = j \end{cases} \quad (5)$$

Describing the OUTSIDE scores for the initial start edge is easy. The start edge is initialized with a zero value for all parts of the OUTSIDE score. New empty actives inherit the acoustic and n-gram scores from the introducing active. Since we introduce a new grammar rule in an empty active edge, the grammar score has to be updated accordingly. This mechanism ensures that an outside score is always maximal: Since we process the agenda above the beam in a best first fashion, it is guaranteed that the first SEEK DOWN operation leading to a certain empty active is the one with the best score. OS refers to the combined score, OS.X to the single component X of that score accordingly.[10]

Definition 3.2 *OUTSIDE score for the start edge*

$$\mathrm{OS}(E_{0,0,\emptyset,\mathrm{Startgraph}}) = 0 \quad (6)$$

Definition 3.3 *OUTSIDE score for empty actives resulting from a SEEK DOWN operation.*

$$\begin{aligned} &\mathrm{OS.X}(E_{j,j}) = \mathrm{OS.X}(E_{i,j}) \\ &\mathrm{OS.GO}(E_{j,j}) = \mathrm{OS.GO}(E_{i,j}) + 2 \\ &\mathrm{OS.GP}(E_{j,j}) = \mathrm{OS.GP}(E_{i,j}) + \\ &logP(type(E_{j,j}.\mathrm{mother}) | type(E_{i,j}.\mathrm{next}), \mathrm{gm}) + \\ &logP(E_{j,j}.\mathrm{rule} | type(E_{j,j}.\mathrm{next}), \mathrm{gm}) \\ &\\ &E_{j,j} \text{ is introduced by } E_{i,j}, \\ &\text{and X being AO, AP, NGO, NGP} \end{aligned} \quad (7)$$

Do define the OUTSIDE score which is given to a resulting edge by an application of the fundamental rule to its daughter active and inactive, we must refer to AO, NGO, GO, AP, NGP GP, and the strings covered by edges. In record access[11] notation: A.intro ist the string of word hypotheses from vertex 0 through the lattice, on which an OUTSIDE score of A ist based on. A.string ist the string of word hypotheses actually covered by A.

Definition 3.4 *OUTSIDE score of an edge C, resulting from an active A and an inactive B by an application of the fundamental rule.*

$$\begin{aligned} &\mathrm{OS.AO}(C) = \mathrm{OS.AO}(A) + \mathrm{IS.AO}(B) \\ &\mathrm{OS.AP}(C) = \mathrm{OS.AP}(A) + \mathrm{IS.AP}(B) \\ &\\ &\mathrm{OS.NGO}(C) = \mathrm{OS.NGO}(A) + \mathrm{IS.NGO}(B) + 1 \\ &\mathrm{OS.NGP}(C) = \mathrm{OS.NGP}(A) + \mathrm{IS.NGP}(B) + \\ &\mathrm{logP}(\mathrm{last(A.intro)} | \mathrm{first(B.string)}, \mathrm{n\text{-}gram}) \\ &\\ &\mathrm{OS.GO}(C) = \mathrm{OS.GO}(A) + \mathrm{IS.GO}(B) + 1 \\ &\mathrm{OS.GP}(C) = \mathrm{OS.GP}(A) + \mathrm{IS.GP}(B) + \\ &\mathrm{logP}(\mathrm{type(B.mother)} | \mathrm{type(A.next)}, \mathrm{gm}) \end{aligned} \quad (8)$$

$$\mathrm{OS(C)} = \frac{\mathrm{OS.AP(C)}}{\mathrm{OS.AO(C)}} + \gamma \frac{\mathrm{OS.NGP(C)}}{\mathrm{OS.NGO(C)}} + \delta \frac{\mathrm{OS.GP(C)}}{\mathrm{OS.GO(C)}} \quad (9)$$

When we combine two edges we compute new values for all the single components of the INSIDE and OUTSIDE score.

4 Probabilistic Typed Unification Grammars

We wanted our parser not only to help the decoder to find the best word sequence. What was inteded originally was a mapping from a signal to a formal representation of a meaning. So, since the typical grammar produces a lot of derivations for a given string, we had to add some disambiguation device in order to choose one of those multiple analyses. Secondly, we did not intend to compute all of the derivations but rather prune those which were not intended early.

The straight way was to have a probabilistic model of the derivations of our unification grammar. The scores can be combined with the other scores and the general beam search will have effect on the completer operations coming from different analyses of the same word string.

Besides the work on PCFGs by Baker [2], Jelinek [13], Fujisaki [8], and others there is a handfull of publications on probabilistic versions of unification grammars, like Hemphill & Picone 89[12] [12], Briscoe & Carroll 93[13] [3] or Magerman & Marcus 91 [14][14].

All this work is on unification grammars that do not use typed feature structures. So the main thing is to identify finitely many classes of feature structures in order to apply PCFG methods

[10] Again in record notation.

[11] which is really the way we implemented it.

[12] Don't read it.

[13] Read it.

[14] The work on Pearl is rather on replacing a unification grammar, but nevertheless on this topic.

to the observation of derivations of the unification grammar. Mapping of infinitely many FS to a set of classes is usually done by creating a set of *restrictions* in the sense of Shieber 85 [19], which do not unify which each other. Observing the classes in a derivation instead of the original FS leads to the possibility to decide to which class a left hand side of a rule belongs. So we can train some n-gram models on right and left hand sides of rules.

In a typed unification grammar like ours, we use a type system with *appropriatness* as defined in Carpenter 91 [4]. In such a system, we have a finite set of type names associated with well formedness conditions on FS. Using the types of FS as class names, we do not face the same problem as we do using unification grammars without types.

On the other hand, a type is not a fixed label as eg in CFG. Simple types are usually defined in an IS-A hierarchy which determines the unifiability and subsumption relations of types. Complex types arise from unification of types where two types unified have several common subtypes.

Assume the unification of two typed FS as in (10):

$$A[f1 : v1] \cup B[f2 : v2] = C[f1 : v1, f2 : v2] \tag{10}$$

The type C can be equal to A or B or be a common subtype of both.

In our grammar rules global types of FS encode much linguistic information and a lot of linguistic relations are encoded in the type hierarchy instead of grammar rules. An example is the instantiation of the so called *Vorfeld* in german sentences. The following simplified example shows the method.

$$\text{S2\_intrans[..]} \rightarrow \text{Vorfeld[..] V\_intrans[..]} \tag{11}$$

$$\text{Vorfeld} > \text{NP PN Perspron S\_adv...} \tag{12}$$

Having a number of rules like the one in (11) in our typed unification grammar, the type declaration in (12) specifies, FS of which type can instanciate with FS of type *Vorfeld*. The former are just stated to be subtypes of the latter.

Classifying FS by type names and using them directly for a PCFG style probabilistic model leads to trouble. We cannot guarantee to always have $\sum_\beta P(\beta|\alpha) = 1$ for rule schemata $\alpha \rightarrow \beta$. since we do not Know what α's actual instances will be. Compiling out the types and multiplying the rules accordingly would be possible, since in our system we only have a finite number of types – the power set of all simple types. One the one hand this leads to a huge set of rule instances, on the other hand there are systems like the one of Emele & Zajac 90 [7], where infinitely many types may occur. So the method is not general.[15]

So what we do in order to get a proper statistics on derivations on the typed FS rules is to decompose a rule application and a unification of two typed FS into two parts: The shift of the types beeing unified and the application of the rule itself. When for instance, in a sentential form of a left derivation a FS of type *Vorfeld* is unified with a FS of type *NP*, we have two observations, the shift of types *(NP, Vorfeld)* and the application of a rule with original lhs type *NP*. In fact we do not care about the result type of the type unification which might be a complex type.[16]

Definition 4.1 *Given a left derivation in a typed unification grammar of the form:*

$$S[..] \stackrel{*}{\Rightarrow} x\alpha' y \Rightarrow x\beta' y \stackrel{*}{\Rightarrow} w \tag{13}$$

where α' has been generated as an instance of α'' a member of the right hand side of some rule, and β' has been generated by application of a rule $\alpha \rightarrow \beta$ by unification of α and α', we call the pair type(α''),type(α) an **observed type shift**.

In our model *gm* for the derivation of the typed unification grammar we describe each rule application – in other words: step in a left derivation – as a pair of a type shift and original rule of the grammar.

Implicitly this is adding some unary rules for all type shifts and guaranteeing that in every derivation type shifts and rules are applied alternatingly.

Definition 4.2 *A probabilistic typed unification grammar is:*

[15] It is ugly, which is the main argument.

[16] We could have done this, but the amount of paramters would have increased by $O(2^n)$, where n is the number of simple types.

Type hierarchy: *A lattice of types with a top and bottom element which defines the subsumption relations of types.*

Lexicon: *We assume the lexicon to be a set of unary productions, where the right hand side of the production is a word string.*

N-ary grammar rules: *The grammar rules consisting of one typed feature structure as a left hand side and a sequence of typed feature structures as a right hand side.*

Model gm: *Assigning a probability to all pairs of two types and to all pairs of type and rule.*

The model *gm* is organized as two bi-grams, one for the type shifts and one for the rules. The type shifts and rules are thought to be independent. So the following relation holds:

$$\sum_B \sum_\beta (A \overset{\text{tsh}}{\rightarrow} B, B[..] \overset{\text{rule}}{\rightarrow} \beta | gm) = 1 \quad (14)$$

During training, we treated the type shifts as if they were unary productions. We parsed a corpus with the original typed unification grammar. Parses were represented as lists of the type shifts and numbers of rules that were involved. We used a variant of the unsupervised training method of Fujisaki et al. 91 [8] to estimate the models.[17]

During parsing we can distinguish three types of type shifts.

- Those pairs of types which are not unifiable are type shifts of probability 0. Thus, we can prune an impossible analysis on grounds of *gm* instead of performing a unification that will fail anyway.
- Possible but not observed type shifts should receive a smoothed value as usual in standard bi-gram techniques.
- Observed type shifts receive their trained probabilities.

By our method of having an additional bigram of type shifts, we achieve a couple of pleasing properties. The first one is, that we can have a correct probabilistic model of a unification grammar which relates typed feature structures with each other. Furthermore, by using the types as model classes, we can capture a lot of information of the unification grammar in our model, since types are tied to the feature structures by *appropriatness* checking as described in Carpenter 91 [4] and the subsumption relations of types in the hierarchy are captured as well. The method generalizes to the powerful type systems in the style of Emele & Zajac 90 [7] since only pairs of simple types[18] are used in the model. Since in type unification the resulting type is determined by these two types we do not lose any information.

Finally, if for a given grammar the hierarchy of types is flat, the model works like a real PCFG backbone for a unification grammar, as it is used for example by the TINA parser [18]. In that case, all types are unifiable only with themselves and with the top element of the type lattice. When no rules of the grammar use top as a global type of a FS, we will arrive at a collection of observed type shifts all of which have the form (X,X) with probability 1.

5 Tightly Coupling the Parser with a Beam Decoder

Some extensions of the LR-ACP allow for different time synchronous couplings with a beam decoder for word hypotheses. So far we investigated three modes, namely time synchronous parallel bottom up (*BUI*), time synchronous bottom up with top down verification (*BUITDV*) and time synchronous top down predicting mode (*TDPI*). Some results of these couplings driving the parser with an n-gram only have already been presented in Hauenstein & Weber 94 [10, 11]. The decoder is a viterbi one pass beam decoder[19] extended for the different protocols as explained below.

[17] We do not object to supervised learning. We just did not have a tree bank.

[18] In fact complex types may also be used. The bound comes from the type names occuring in the original grammar rules before any grammatical operation took place.

[19] .. developed in the VERBMOBIL TP 15 project, by Andreas Hauenstein, University of Hamburg.

5.1 BUI

In BUI we run the decoder and the parser concurrently. In every frame processed by the decoder, word ending hypotheses above the decoders beam are immediately sent to the parser. The decoder does not use language model[20]. The hypotheses are quadruples *(from,to,key,score)*, where *key* is the name of the wordform and *score* is the acoustic score assigned to the frames *from* to *to* by a word copy of the model for *key*.

The LR-ACP works exactly as described in section 2.3.

The BUI coupling uses no feedback messages from the parser to the decoder, so real parallelism is possible.

5.2 BUITDV

In BUITDV mode the decoder uses top down verifications of the parser as a language model. Since for one word copy a language penalty has to be added only once, a fifth field is added to the bottom up word hypotheses as a flag signalling to the parser whether a hypothesis is new[21] or has already been verified. The whole procedure works as follows.

Definition 5.1 *An example cycle of BUITDV for frame I:*

1. *The decoder finds m+n new word ending hypotheses at frame I. n had already been found at I-1, m are new. All of them are sent to the parser in cycle I with the flags set accordingly.*
2. *The parser takes all m+n word hypotheses and performs PUSH-EDGES resulting in an initial $agenda_I$.*
3. *The parser processes $agenda_I$. The first successful application of the fundamental rule to a pair of edges involving a new word hypothesis leads to a verification of that hypothesis.*
4. *When the beam of $agenda_I$ is reached, the rest of $agenda_I$ is searched for word hypotheses not yet verified. The first one found leads to a verification of that hypothesis.*
5. *The decoder receives verification messages. All those new hypotheses in I, which were not verified are set to a zero probability. All other new hypotheses have the verified language penalty added.*

Verification messages are built by the parser using agenda items. They are quintuples *(from, to, key, score, flag)* as the associated bottom up hypotheses are. *from*, *to* and *key* identify the decoder's active word copy which was to be verified. The *flag* field is supplied with the predecessor word hypothesis' string, which led to the best score inside the parser's search. For *score* a couple of options are given.

Definition 5.2 *Transition score with bi-gram and gm supplied by a verification message for a pair (A, W). of edges on the agenda.*

$$\begin{aligned} &score = \\ &\gamma * logP(first(W.string)|last(A.intro), n\text{-}gram) + \\ &\delta * logP(type(W.mother)|type(A.next), gm) + \\ &\delta * logP(W.rule|type(W.mother), gm) \end{aligned} \tag{15}$$

Definition 5.3 *Transition score with bi-gram supplied by a verification message for a pair (A, W). of edges on the agenda.*

$$\begin{aligned} &score = \\ &\gamma * logP(first(W.string)|last(A.intro), n\text{-}gram) \end{aligned} \tag{16}$$

The score which is used as a transition penalty between words by the decoder can consist of a bi-gram score and a score given by the grammar model for those grammar operations, which are used to incorporate the word hypothesis into an existing partial analysis. The latter consist always of the score for the lexical access[22] and the type shift which takes place when the lexical entry is combined with some rule actually processed by some active edge. This is the part of the grammar operations which can directly be related to a local transition between words. Again both scores are weighted, hence not normalized. Alternatively, only the bi-gram score can be used as in common decoding.

Maximization is not local here. Since predecessor word and analysis are selected on the base of agenda items, the criterion for the maximization on predecessors is global.

[20] But a fixed transition penalty in order to prevent an inflation of short words.

[21] Then this word copy's final state has been above the decoder's beam for the first time

[22] treated as an unary rule, namely W.rule in def. 5.2.

One difficulty we have to face is that the decoder will have access to transition penalties when a word model runs into a final state and not at the beginning of a word. We can overcome this problem with a method inspired by Aubert's et al. 1994 [1] handling of tree lexica in one pass decoding. For both the n-gram transition and the part of the grammar score used above there exists a stable maximum for a word with respect to all possible immediate predecessor words.

So starting a word copy in the decoder we can use this maximum as a transition penalty adding the difference with respect to the maximum when the copy has been verified.

5.3 TDPI

TDPI is the opposite method of coupling to BUITDV in terms of contol, since the parser predicts possible successor word hypotheses starting in a given frame while the decoder selects among those.

The basic procedure in TDPI goes as follows:

Definition 5.4 *An example cycle of TDPI for frame I:*

1. *The decoder has sent all word ending hypotheses at frame I.*
2. *Having parsed all word hypotheses ending in vertes I, the parser takes all the active edges in vertex I and calculates a set of predicted predecessor words. These are sent to the decoder supplied with transition scores as a* **prediction** *for I.*
3. *The decoder in frame I starts only those new word copies which occur inside the prediction. Newly started word copies are initialized with the supplied transition penalty.*

A grammar based calculation of predictions is too expensive since a full backward search (completer step) would have to be executed for each candidate. So we propose a generate and test algorithm for the computation of a set of predictions, based on the chart up to vertex I:[23]

Definition 5.5 *Computing predictions*

1. *Let ACTIVE be the nonempty active edges ending in* $vertex_I$, $P_I = \emptyset$
2. *Take the n best successors s of any* $w = last(A.intro)$, $A \in ACTIVE$, *maximizing* $logP(s|w, n\text{-}gram)$.
3. *Create* $vertex\__I$ *and insert inactive edges for all s into* $vertex_I$, $vertex\__I$, *thus filling* $agenda\__I$.
4. *Until* $below\_beam(agenda\__I)$, *let (A,S) be* $pop(agenda\__I)$:

   When $combine(A,S) \neq fail$, *remove all pairs (X,S) from* $agenda\__I$ *and set* $P_I = P_I \cup (I,0,S,last(A.intro),score)$.

The score of a quintuple (from, to, predicted word, predecessor, score) in P_I is the same as for the verification messages given in 5.2. Again, the actual score returned is globally selected. The reason for the n best technique in step 2 is, that we can precompute this step and hash the results. So the complexity of the whole calculation is determined by n times the number of active edges ending in vertex I in the worst case. We do not use a completer step to test the predictions, since one success with some active edge is sufficient to show that there is at least one spanning analysis according to the type skeleton of the grammar. Adding all the features to the test would be prohibitive. If we use a class based n-gram model where the classes correspond with types of our grammar we could reduce n considerably.[24]

6 Incremental Time Mapping using Edge Inheritance

The set of word hypotheses transferred bottom up from the decoder to the parser in a time synchronous coupling constitutes a *left connected word graph* seen a posteriori. In a non incremental architecture where word recognition ends before parsing starts, we can delete all dead ends in a word graph using an offline backward search phase through the set of word hypotheses. An abstracting step from frames to vertices can omit superfluos paths in the graph consisting of word ending hypotheses differing only in one frame. Chien et al. 90, 93 [6, 5] present such a mapping from frames to vertices.[25]

[23] A broad discussion of efficiency in chart based computation of predictions can be found in Weber 94 [20].

[24] We did not try that yet.

[25] Their method is a bit cryptical, but the only reference we found in the literature.

In our incremental architecture this does not work. When the decoder finds a word ending hypothesis we never know whether the word copy in question will still be above the beam in the next frame and if yes, whether its normalized acoustic score will increase or decrease. This leads to the effect, that sometimes the parser receives exactly the same set of word hypotheses in two successor frames, differing only with respect to their end vertex. The same completer actions are carried out two times in such a case. In order to implement a normalization without any lookahead, we developed a method we call incremental time mapping, which consists of a slight modification of the LR-ACP where some book-keeping is used to do all completer actions only once which stem from the same decoder's word copy. The basic idea is to inherit all those edges from vertex I to vertex I+1 which resulted from a word hypothesis in vertex I for which an identical one has later been found with ending time I+1. All word hypotheses found for the first time are parsed as usual.

For that issue, we add a little non monotonism to the chart parser:

- One edge can belong to a number of vertices. Seen from the vertices the edge is shared. Seen from the edges, an edge now possesses a list of end vertices the head of which is the actual one.
- We allow for the replacement of edges by others. This mechanism is only used on empty active edges. It is guaranteed that no inconsistencies arise, since we replace only edges ending in vertex I in cycle I.

The basic modified control loop for the LR-ACP can be paraphrased as follows.[26]

Definition 6.1 *Descripition of a cycle$_I$ in an LR-ACP with edge inheritance.*

READ HYPOS :

KNOWN, NEW, INH_EMPTIES be $\emptyset$.

Read in all hypotheses ending at I. Add those which have a predecessor at I-1 differing only in ending time to KNOWN.

Let pred(KNOWN) be the corresponding hypotheses in vertex$_{I-1}$. Add all others to NEW.

INHERITANCE :

For all edges A ending in vertex$_{I-1}$, A coming from some w $\in$ *pred(KNOWN):*

When score(w) < score(w)' in KNOWN, update the acoustic score and operations of A.

If empty_active(A), add A to INH_EMPTIES

Else, add A to vertex$_I$.edge-in, add vertex$_I$ to A.to. When A is active, perform a SEEK-DOWN on A in vertex$_I$.

For all the pairs (A,B) in save_agenda$_{I-1}$, B coming from some w $\in$ *pred(KNOWN), push (A,B) to agenda$_I$.*

PARSE :

Insert all word hypotheses in NEW into the chart. MAXVALUE := max(max(OS(A)|A $\in$ *vertex$_I$.edge-in), score(top(agenda$_I$)))*

BEAMVALUE := MAXVALUE - BEAMWIDTH

Apply fundamental rule to pop(agenda$_I$) until score(top(agenda$_I$)) $\leq$ BEAMVALUE.

Save agenda$_I$.

UPDATE EMPTIES :

For all A in INH_EMPTIES: If there exists no empty edge B in vertex$_I$, with A.rule = B.rule, add copy(A) to vertex$_I$. Else, if B.OS < A.OS, replace B by copy(A).

In order to keep the beam search mechanism working as in the original LR-ACP we have to take care of the inherited edges, when the BEAM is calculated. Furthermore, since the scores of edges inherited may be subject to an updating and since the new maximum in cycle I may differ from the maximum in cycle I-1 those pairs with inherited edges, which had been pruned in cycle I-1 must be pushed on the new agenda.

Nevertheless, scoring is different from the original version, since we only update an edges acoustic score until a maximum is reached.

In the original version, a sequence of say 5 word hypotheses coming from the same decoder,s word copy was represented by 5 edges[27]. One of those had the maximal acoustic score. Using incremental time mapping, we represent the 5 word hypotheses by only one edge, having 5 possible updates on the acoustic scores. In order to keep the global Viterbi search correct, we stop updating when the maximum is

[26] A more precise description can be found in Weber 94 [20].

[27] Or $5n$ edges, when n lexical entries where found.

reached, so the whole sequence of original hypotheses will be represented by its maximum. The mechanism is similar to the pruning used in Ney's 91 [15] dynamic programming parsing based on a CYK parser, but constitutes an incremental time synchonous version of the latter.

While the algorithm above works well for BUI and TDPI, in BUITDV some slight modification have to be made. In BUITDV the decoder sends all those hypotheses first which are to be verified, waiting for response, and sends all known hypotheses later. So the decoder can prune based on verified hypotheses only but the parser cannot use the maximum of all scores in the first phase, parsing the new ones. A version of the parser with time mapping using two distinct steps for the new and known hypotheses worked well in experiments although the first step did not have certain knowledge about the maximum in every cycle.

The calculation of predictions in TDPI can be subject to the inheritance technique too. All predictions in a cycle I will be pointed to by the edges they are based on. When the relevant edges are inherited, the predictions coming from them will be inherited with them then.[28]

7 Selected Experiments

In this section we present a couple of selected experiments with the LR-ACP. The n-gram model used in all of the experiments is word based and has a test set perplexity of 54.28. We used two unification grammars for our experiments. A small and rather restrictive one (GRS) was used in the experiments without the probabilistic grammar model. A larger, less restrictive grammar (GRB) with more structure building features was used with the model *gm* in order to have a more realistic test bed. A corpus of 200 sentences of the train information domain was used for training. Tests were carried out on 10 sentences with the small grammar and 5 sentences with the big one. The acoustic models used where well adapted and led to an acoustic word accuracy of 88.6 on the test sentences.

The experiments should be regarded with care. They are of a small scope they and preliminary in nature. Further testing of the architecture and its variants on a new domain with more realistic grammar and bi-gram are currently at work.

[28] Again, details can be found in Weber 94 [20].

The two tables below present the results of Hauenstein & Weber 94 [10, 11] for decoding alone (AC), BUI (BU), BUITDV (BV) and TDPI (TP), using GRS with the bigram only. We measured cpu time (T)[29], bottom up hypotheses (BUH), top down hypotheses (TDH), sentence recognition (SR), edges used (E) and maximal active gridpoints (GP).

The first table shows results for narrow beams, the second for wide beams.

	T	BUH	TDH	SR	E	GP
AC	47.4	–	–	0.5	–	790
BU	96.9	385	–	0.7	6195	790
BV	58.2	289	41	0.5	5827	1045
TP	220.5	223	7621	0.7	4954	78

	T	BUH	TDH	SR	E	GP
BU	2911.7	1465	–	0.7	19769	1712
BV	115.8	776	135	0.7	9765	2047
TP	282.7	415	8304	1.0	7519	184

The effect of the top down strategies cannot be overseen here. The amount of bottom up hypotheses is heavily reduced by the restriction supplied to the decoder.[30] For the relatively well adapted hmms, the TDPI strategy leads to a sentence recognition of 100 percent using wide beams. BUITDV is superior to BUI only in terms of efficiency.

The effect of the time mapping can be seen in the following table, which changes the overall impression given by the first results. All parameters besides the time mapping are as those used in the wide beam case. The table shows BUI, BUITDV and TDPI with incremental time mapping. TDPI has been tested with standard time mapping only (TM) and with time mapping for the computation of predictions (TMP). The comparison with time mapping has been done on five of the test sentences, which were well recognized by all the couplings.

	Time in Sek.	Edges
BUI, TM	68.8	6613
BUITDV, TM	51.4	4662
TDPI, TM	172.6	5004
TDPI, TM & TMP	58.2	4226

[29] Allegro Commom Lisp on a SUN Sparc10, parsing time.

[30] BUI without time mapping often exhausted the machines 48mb RAM causing swapping which explains the huge increase in cpu time.

We can see, that in terms of efficiency the strategies do not lead to similar results, when redundant completer operations are omitted. The TDPI case with standard time mapping gives a good impression of the enormous overhead caused by the calculation of predictions.

The first results with a larger grammar (GRB) and the model gm are presented below. The first column shows the weights for *n-gram* and *gm*. The single weights were optimized on the test sentences by hand, maximizing sentence recognition rate. The rows show *gm* only vs *n-gram* only vs a combination of both models. PR and TR stand for pruned and tried agenda items. T, SR and E are defined as above. The table shows a version of GRB without rules for performance phenomena and a BUI coupling.

ng/gm	PR	TR	T	SR	E
0.0 0.4	19920	2075	103	4/5	7774
0.2 0.0	7031	25796	137	3/5	8874
0.1 0.2	26381	2486	108	4/5	8192

The full GRB grammar and TDPI coupling with n-gram value supplied to the decoder with the prediction, led to the following results:

ng/gm	PR	TR	T	SR	E
0.0 0.4	69052	530	145	5/5	6627
0.2 0.0	22588	5343	171	5/5	6648
0.1 0.2	69858	599	157	5/5	6646

The most prominent observation is, that when the influence of *gm* is set to zero, much more agenda items are tried. The reason for this is, that on the basis of *gm* a lot of unifications can be pruned, that would fail with a type mismatch if tried. So these agenda items do not lead to additional edges. They are responsible for a growth of processing time, since the processing of the scores is cheaper than unification. Astonishing there is a slight increase in processing time and edges from *gm* alone to the combination of the models. We had expected the opposite effect. The weights, which were optimized by hand, might be the reason for that. A better tuning of the weights by an optimizing procedure, taking smaller steps than our hand tuning, should correct this.

8 Conclusions

We presented an active chart parser which combines different statistical knowledge sources in parsing speech in a time synchronous fashion. The beam search implemented on the agenda takes global scores into account.

On the basis of this parser, different couplings with a decoder in the style of Hauenstein & Weber 94 were explicated. The time mapping technique presented here leads to the effect, that differences in efficiency between the couplings observed by Hauenstein & Weber 94 become small.

TDPI is superior to the other couplings with respect to recognition rate in the experiments performed. This does still hold, when a statistical model of the unification grammar's derivations is included.

We introduced a simple method to supply a typed unification grammar with a statistical model of its derivations by observing type shifts on the one hand and type-rule pairs on the other hand. The method generalizes to most known type systems used for such applications.

Further experiments, larger in scale have to be performed in the future.

References

[1] X. Aubert, C. Dugast, H. Ney, and V. Steinbiss. Large vocabulary continuous speech recognition of wall street journal data. In *ICASSP*, 1994.

[2] J.K. Baker. Trainable grammars for speech recognition. In *Speech Communication Paper, 97th Meetingh of the Acoustical Society of America*, Cambridge, Mass., 1979. MIT Press.

[3] Ted Briscoe and John Carroll. Generalized probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59, 1993.

[4] Bob Carpenter. *Typed Feature Structures: Inheritance (In)equality and Extensionality.* CMU, 1991.

[5] Lee-Feng Chien, Keh-Jiann Chen, and Lin-Shan Lee. A best-first language processing model integrating the unification grammar and markov language model for speech recognition applications. In *IEEE Transactions on Speech and Audio Processing*, volume 1,2, pages 221–240, 1993.

[6] Lee-Feng Chien, K.J. Chen, and Lin-Shan Lee. An augmented chart data structure with efficient word lattice parsing scheme in speech recognition applications. In *Proceedings of COLING*, pages 60–65, 1990.

[7] Martin Emele and Rémi Zajac. A fixed-point semantics for feature type systems. In *Proc. of the 2nd International Workshop on Coditional and Typed Rewritung Systems (CTRS)*, Montreal, June 1990.

[8] T. Fujisaki, F. Jelinek, J. Cocke, E. Black, and T. Nishino. A probabilistic parsing method for sentence disambiguation. In Masura Tomita, editor, *Current Issues in Parsing Technology*, pages 139–148, Norvell, Mass., 1991. Kluver Akademic Publishers.

[9] Günther Görz. *Strukturanalyse natürlicher Sprache*. Addison Wesley, Bonn, 1988.

[10] A. Hauenstein and H. Weber. An investigation of tightly coupled time synchronous speech language interfaces using a unification grammar. In Paul McKevitt, editor, *Proceedings of the Workshop on Integration of Natural Language and Speech Processing at AAAI 94*, pages 42–49, Seattle, August 1994.

[11] A. Hauenstein and H. Weber. An investigation of tightly coupled time synchronous speech language interfaces. In *Proceedings of the CONVENTS 94*, Wien, September 1994.

[12] Charles Hemphill and Joseph Picone. Chart parsing of stochastic spoken language models. In *DARPA Workshop on Speech and Natural Language*, Philadelphia, Pensylvania, February 1989. DARPA.

[13] F. Jelinek. Basic methods of probabilistic context free grammars. In P. Laface and R. De Mori, editors, *Speech Recognition and Understanding: Recent Advances*, volume NATO ASI Series, F 75, pages 345–360, Berlin Heidelberg, 1992. Springer Verlag.

[14] D.M. Magermann and M.P. Marcus. Pearl: A probabilistic chart parser. In *Proc. of the European ACL*, March 1991.

[15] Hermann Ney. Dynamic programming parsing for context-free grammars in continuous speech recognition. *IEEE Transactions on Signal Processing*, 39(2):336–340, February 1991.

[16] Hermann Ney. *Architecture and Search Strategies for Large-Vocabulary Continuous-Speech Recognition.*, pages 59–84. NAT)-ASI Bubión, 1993.

[17] Annedore Paeseler. Modification of earleys algorithm for speech recognition. *NATO ASI Series: Recent Advances in Speech Understanding*, F46:465–472, 1988.

[18] S. Seneff. Tina: A probabilistic syntactic parser for speech understanding systems. In *DARPA Speech and Natural Language Workshop*, Philadephia, February 1989.

[19] Stuart M. Shieber. Using restriction to extend parsing algorithms for complex-feature-based formalisms. In *Proc. of the ACL 1985*, volume 23, pages 145–152, 1985.

[20] Hans H. Weber. *LR-inkrementelles probabilistisches Chartparsing von Worthypothesenmengen mit Unifikationsgrammatiken: Eine enge Kopplung von Suche und Analyse.* PhD thesis, Submitted to Universität Hamburg, FB Informatik, Dezember 1994.

[21] M. Wirén. *Studies in Incremental Natural-Language Analysis.* Number Dissertation No. 292 in Linköping Studies in Science and Technology. Linköping University, 1992.

Developing Natural Language Interfaces: a Test Case

Gert Veldhuijzen van Zanten and Rieks op den Akker
Department of Computer Science, University of Twente,
P.O.Box 217,
7500 AE Enschede

Abstract

Dutch is the best language there is. For the Dutch that is. Therefore they prefer to think in Dutch, not in a foreign language like English, SQL or some other computer language. Instead of learning people to use a strange language for communicating with machines, it may be preferable to learn machines to communicate with Dutch people in Dutch. This has led to the idea of making something "in between man and machine": an interface to bridge the gap between natural language and machine language. How do you make such a bridge and what kind of things do you need for making it? If you can't make a Natural Language Interface for data base querying there are more things you can't do. Making an NLI: a test case for the ability of making more sophisticated natural language processing systems.

1 What is a Natural Language Interface?

A natural language interface for a particular data base allows people who want to extract information from the data base to use their own "natural language", the language they think in and in which they express their questions immediately, i.e. without an explicit process of formulating. Is it possible to make a natural language interface for data base querying by typed queries? That is: an NLI the intended users can really use[1]. Maybe it is true that the ideal NLI allows its users to query the data base by speaking to it. But since you can't make a speech understanding NLI if you can't make an NLI that handles typed queries – although in written text we lack information provided by prosodic variations in spoken utterances – we first leave speech recognition out of discussion. We also assume an NLI is made for users using one natural language, Dutch let's say.

Being toolmakers we are interested in making tools, things that can be worked with effectively and efficiently and that make the work easier to do than without them. Of course as mathematicians, linguists, and/or computer scientist we love formal languages, grammars, and we like to write nice, efficient algorithms even if we haven't the slightest idea for what practical purposes they could be used. These things have their own intrinsic and attractive values. Maybe this is the main reason that in an academic environment it never comes to the making of a true NLI. (An informatics department of a technical university is something different as a pure mathematics or linguistics department at a university. The former one should have an open eye for the practical applications of their products.) A welcomed aspect of NLI's is that it confronts us with many interesting problems that need scientific research of very different kinds in order to be solved. For instance we need parsing theory, results from linguistic research, formal logics and software

[1]We distinguish between a "prototype" of a system that translates natural language sentences into formulas of a data base query language and a useful natural language user interface. Most prototypes, how promising they may be (like PHLIQA1 for instance [Phl79]) never reached the stage of a real NLI.

engineering methods. There is clearly a research management problem here that need to be solved before you can make a technological tool like an NLI. And there is a problem of financing. We will, however, abstract from these kinds of problems here.

If it was clear to us that practical NLI's for particular data bases are impossible this paper wouldn't have been written. The theorem

> Natural language interfaces for data base querying are impossible

cannot be proved in the mathematical sense[2]. This doesn't mean that you can't be convinced of its *correctness*. Some people are maybe more convinced of the impossibility of Artificial Intelligence than about the fact that 1 and 1 makes 2. Maybe they have the opinion that there is no thing "in between natural language and formal language" because these languages are creatures from heterological universes. There is definitively a point in this. But, this is not the time and place to hear philosophers. We try to explore the borders of language technology by trying to construct the bridge. The methods to prove the *incorrectness* of the theorem are both constructive. Either show one or, if you can't find one, make one. Since we didn't find one, (to be honest we didn't spend much time in searching one), and since we are not only interested in the product itself but also, and even more, in the process of making one (since this is the only way to become known about all finesses of how it works as it works) we decided to make one. NLI is a first test for how far we are in making computers understand people using natural language. It is a first test. Our interest in making an NLI for a data base doesn't stand for itself. We will use knowledge, techniques, experiences, contacts, and methods gained and developed for succesfully passing this test in the Schisma project (see the contribution about Schisma in these proceedings). If we don't succeed in passing the test we can always say that it is because of bad management or lack of finances. Or maybe we will find ourselves convinced of the fact that there is something wrong in principal with that "in between natural and machine language" or with the idea that this "in between" can be implemented as a "technological interface" between man and machine. We don't ignore the possibility that there might be something principally wrong with this idea.

In pondering about NLI's we should answer the question whether a natural language, like Dutch, is also a "natural" language for man-machine communication. Of course the answer depends on what you still consider to be Dutch, or to be "a natural language" in general. The fact is that people adapt their language use to the ones they are communicating with. In communicating with machines people don't have to show how well-behaved or well-educated they are, although some users may act as if they are communicating *by means of* the machine with someone inside or behind it. If a user of an NLI happens to experience that typing or uttering the phrase "employees Philips Research Eindhoven" (not a sponsor) has the same effect as typing the full sentence "Please show me all names of employees that work at Philips Research Department in Eindhoven", he will definitely prefer the shorter one in communicating with a machine, but most likely not in communicating with someone at Philips Research. In man-machine communication it is the effect of the question that counts, and nothing else. And if you're sure that saying *yep* to a machine has the same effect as saying *Please start up the database* you will say *yep*. Hence, in men-machine communication, more than in men-to-men communication, efficiency is a dominant factor in evaluating the communication language that is (to be) used. But does this mean that we are arguing in favour of using a formal language

[2] A mathematician could come up with a functional specification of a natural language interface, calling this function *nli*. The function *nli* can be either possible (in the sense of computable) or not. But a function is not a practical tool.

instead of a natural language? No. Although every NLI is based on some (formal) *model* of a natural language, we still believe the user of a data base should not be bothered with learning a formal language before he or she can phrase his first question. That is why the model on which the NLI is based should be a good model of a *natural language*. It means that an NLI should adapt it's language in communicating with the user. So, starting with a clear fragment of proper Dutch, the NLI and the user should be allowed to propose short phrases for complete sentences. Once both parties agree upon the use of these short phrases, having a well defined semantics in terms of a complete sentence, they can from that time on be used for the same purpose as the complete sentence. A natural language interface should have an adaptive natural language in order to be practical. It is often remarked that an NLI shouldn't stand on its own. It should be a part (mode) of a *flexible* multi-media interface. ("Flexible" in the sense that it is up to the user to decide in what form he wants to provide his information: by speech, by pointing at objects or by typing.) We agree with this but in this paper we will consider single-mode NLI's for data base querying. Moreover, we assume no references are made in questions to former queries or answers to former questions. Until now we assumed that the NLI leaves the user free in the way he formulates his question. A lot of problems introduced by this freedom can be "solved" by a syntax directed NLI. Such an NLI will show the user, each time he has typed a word, the possible continuations of the typed prefix. The possible continuations can be offered as a list of possible categories of words or phrases or in terms of concrete words (belonging to closed word groups, like *what, where, how, how many*) and the user can select the category or word he wants to use. The user should of course not be bothered with grammatical categories; not with NP's. Some typical examples of categories allowed and covered by the interface could be offered. The ordering of possible continuations offered can be dynamically updated depending on the history of use; either the last used continuation, or the most frequent used one is listed first. For uninitiated users and not dayly users an NLI that has such a syntax directed mode may be quite handy. Experiences from real users will decide whether this is really true. Existing tools for the generation of syntax-directed editors may be used for this mode of an NLI.

In this paper we present our experiences, interests, problems, and our plans for passing the test. We discuss several aspects of NLIs for data bases (like robustness and quality) and of the process of making them. Especially a parser for unification grammars [Shi86] and a tool for developing the front-end part of NLI's (i.e. the natural language parser) will be discussed in some more detail. The paper is organised as follows.

2 Stages of Handling Users Input: a categorization of users input according to the various reactions of the NLI. How do we handle typing errors?

3 User Requirements: what does the user expect from a practical NLI? (not completely hypothetical)

4 The Kernel of an NLI: in which the grammar and the intermediate languages between the natural query language and the data base query language are discussed.

5 Developing a Grammar for an NLI: how do we make a grammar, what is a good tool for developing one, and how do we specify the meaning of a natural language query? (presenting some ideas to be worked out in the future)

6 Left-Corner and Head-Corner Parsing: why? and how does it work? (not formal either but there are references given to more detailed specifications of this parser.)

7 Efficient Unification: how do we unify feature structures? In which a table is presented with experimental results from a

comparison between our unification algorithm and the best one we found in the literature.

8 Robustness of NLI's: what could we mean by this buzz word? Are stochastic methods useful for obtaining more robust natural language processing systems and why not?

9 Plans for the Future: what should we do without them?

2 Stages of Handling Users Input

Before we come to formulate user requirements for NLI's in terms of systems performance, we must categorize typed user input with respect to the possible reactions of the system.[3].

A user types a sequence of tokens. Sometimes this is a perfect sentence, but sometimes it contains typing errors. Typing errors can be destructive for people who read the "sentence", they can also be nondestructive to people, in the sense that human readers have no problem in recognizing a sequence of words and separators and punctuation marks. Typing errors sometimes are information destructive. They lead to a diversion in meaning among human readers about what the user intended to write. A typing error may result in an other word than the word that was meant. Such an error is not destructive in the sense defined above, but it is information destructive if it is not clear using only grammatical knowledge what the intended word was. Consider a user who types "female" instead of "male". This can be, of course, because of misthinking. Such an error does by itself not lead to an incorrect sentence and we cannot expect from a system that it recognizes such kind of errors, which are not typing errors. We cannot even expect that from human readers. But if a user types "fmale" we have to do with a destructive error because by itself we cannot know whether "male" or "female" was meant. However, in the sequence "male or fmale" the human reader will recognize the intended phrase "male or female". Even in the sequence "male or male" the human reader may recognize the intended phrase "male or female" or "female or male". There is semantic knowledge involved in recognizing this.

If a user types a sequence of tokens the system should correct nondestructive typing errors and show the user its result asking whether this is what he/she intended to write. The user can either confirm or correct the query. At this *first* stage only lexicographical word knowledge is used, no grammatical or semantical knowledge. The user seeing what he has typed has the possibility to correct his sentence. The result of this first stage is a sequence of tokens, t, not containing words that do not belong to a well defined set of words D. This set D is a superset of the words that the system can handle lexicographically, grammatically or semantically, and it may be a proper subset of the set of all English words. The token sequence t may contain spelling errors or ambiguities the user isn't aware of. This sequence is fed into the *second* part of the system, which is the morphological analyzer. No grammatical or semantical knowledge is involved in this process either. The result of this stage is a sequence of words, with their morphological analyses. There is no feedback to the user in this stage. (For the realization of this stage we can use results obtained by Vosse in making a Dutch spelling corrector, see [Vos94]). In the *third* phase words are looked up in the lexicon, a set of words with syntactical and semantical information. We will return to syntactical and semantical information later on. If a sentence contains a word w that cannot be built from a word in the lexicon at this stage the system responds to the user that it doesn't know the word. Let S denote

[3]Although Dutch is the best language for us, and although we intend to make a Dutch NLI, for the sake of communication we will adopt a kind of English in this paper; in the language used as well as in the examples.

the set of all sentences accepted by this stage for further processing. Sentences in S are fed into the following, the *fourth*, stage in which the sentences are analysed grammatically. A sentence will be called *correct* if it can be analyzed with respect to the formal grammar on which this parsing phase is based. [4] Correct sentences can be semantically ambiguous or unambiguous. If a sentence is not correct the system will respond to the user that it cannot handle the query: "I don't understand what you mean by this query". Ambiguities (i.e. semantic ambiguities– syntactic ambiguities that don't lead to semantic ambiguities fall outside this class) are reported to the user. The user can either be asked to choose from a small set of possible readings or to rephrase his question. Sentences that reach this stage of the process are correct and semantically unambiguous. They form the set U of correct unambiguous sentences. Notice that "correct" doesn't mean correct English. The "natural language" the NLI can handle may, and preferable will, contain sentences that are not correct English. (As we mentioned in our introduction, this language will cover some short sentences that don't have to be grammatically correct English, they are "private" formulas. And the system will also not stumble over spelling errors if they don't introduce ambiguities.) Not all sentences in U can be answered by the system. It may contain phrases like "the price of the employees" the system does not know how to handle since although it knows of prices and of employees, the type of the semantic entity the word "price" denotes is not consistent with the type of the entity the word "employee" denotes. These type restrictions not only depend on the linguistic category of the words (employees may have prices) but also on the semantic domain of the data base: in some domain the phrase may have a well-defined meaning. Type information may resolve syntactic ambiguities: "the names of companies that have more than 100 employees". The set US contains all sentences that are in S and also well-typed with respect to the semantic domain. Sentences that are in S but not in US will be answered by: "I can't answer this question".

The performance of an NLI can partly be given in terms of the relation between users inputs that result in a sentence belonging to the set S after the stages mentioned above and the systems response. A system *responds correctly* to a typed query in natural language if the response of the data base is the same as that of the system that is fed by the formal query found by experts who translated the natural language query into a formal query without using additional information to the information expressed in the users input. This definition (similar to the one found in [Bov93], discussing an NLI for the ATIS domain) can be used for obtaining an objective measure for measuring this aspect of the performance of the NLI. (In order to cope with the problems introduced by the informal notion of "information expressed in the users input", we should ask a number of experts to translate the natural language query into a formal query.) This makes it possible to compare different NLI's for a fixed data base. Comparing NLI's for data bases covering different domains is far more difficult, however, since the complexity of the formal model of the natural language part (defining syntaxis as well as semantics) depends on the complexity of the domain.

3 User Requirements

An NLI for information querying from a particular data base should meet the following user requirements.

1. The user doesn't have to know about the organisation of the data base, only about the kind of information it can provide in order to query it by using a natural lan-

[4] If one conclusion from experiments with commercial NLI's for data bases can be made it is that partial analyses (keyword recognition) of user input is simply out of the question if you want a robust system (see [Sij93] for a review of "commercial" NLI's for data bases).

guage.

2. The natural language provided by the NLI is
   (a) reliable: the answer to a question resulting in a sentence that belongs to the set US is the same as to the question phrased by experts in a formal query language (SQL lets say).
   (b) information-complete: all information that can be obtained by a formal query can be obtained by some natural language query.
   (c) broad covering: most natural language utterances used by an uninitiated user, who knows only about the kind of information stored in the data base, are covered by the natural query language.
   (d) adaptive: the NLI offers the possibility to communicate about the language that may be used for phrasing his questions.
3. The NLI doesn't accept semantically ambiguous sentences or phrases, but reports them to the user, offering alternatives or the possibility to rephrase his query.
4. An NLI should be robust. That is:
   (a) the NLI doesn't stumble over non-information-destructive typing, spelling, or grammatical errors.
   (b) The NLI accepts syntactically ambiguous sentences if they are not semantically ambiguous.
   (c) If a user request contains words (like *flower* or phrases like *beating about the bush* the NLI doesn't cover, because they haven't anything to do with the conceptual domain of the contents of the data base, it gives a message like "I don't know about *flowers, bushes*".
5. The time spend by the NL front-end on processing a well-phrased question is less than a small constant times the time spend by the actual querying of the data from the data base.

Requirement 1 will be clear. Some prototypes of NLI's expect from the user that he knows names of the tables of a relational data base. In our view such systems ask too much from the user. The user should of course know something about the data base: its contents, the kind of information it contains, but not about things like tables, keys, which have to do with the organisation of this contents. Notice the difference between requirements 2a) and 2c): a useful NLI should be 100% reliable (is 99% also reliable?), 2c) coverage has to do with how many percent of the sentences typed in by the users reach the stage of being answered properly, at one of the stages or by a reliable answer. We have already given arguments for requirement 2d) in the Introduction. If a user uses similar sentences quite often, the NLI should offer him the possibility to shorten the sentence. This can be implemented by storing questions, or schemes of questions. So the user may write "names of kids of Johns son" instead of "What are the names of the kids of Johns son?" In a further section we will come back to the issue of robustness. Sometimes requirement 5) is sorted under robustness-requirements. We don't because it depends too much on other things, like efficiency of algorithms, clever choice of data structures, and properties of the software and hardware environment in which the NLI operates, than on the method implemented for handling the users language proper. Efficiency is however an important aspect of performance: a system may use, for instance, some sophisticated method for semantical error correction but if the user has to pay an extra few minutes waiting before he gets his answer the tool will probably be put on the shelf. We will return to efficient parsing and translation later on.

4 The Kernel of an NLI

It will be clear by now that we have a linear architecture for the NLI in mind. The processes discussed before filter the input and sentences that pass these filters are fed

to the kernel of the NLI. These first four stages allow us to choose for online word by word processing, although some typing errors may destroy the intended word boundaries. The grammatical and semantical kernel of the natural language interface will be split up in two translation phases at least. The front-end translates NL into some Intermediate Language (IL) and the back-end translates IL into SQL, or some high level logical/knowledge representation language for data base querying. This makes it possible that the grammar, the lexicon and the IL become to a large extent independent of the contents and the organisation of the particular data base. The front-end becomes reusable for other data bases. This hasn't so much to do with the performance of a particular NLI for a particular data base, but more with the management of the process of developing an NLI and– not unimportant– the reusability and extendibility of parts of it. Moreover, further research and development may be directed towards the generation of NLI's for particular data bases using fixed parts of the grammar, the lexicon and the intermediate language(s).

Several candidates could be nominated for the price of the best intermediate language: QLF (the Core Language Engine), WML (from PHLIQA1), PTQ (Montague Grammars), to name a few. The Quasi Logical Formalisms used in the CLE project are not specifically for the semantics of natural query languages for data bases, making them quite, maybe too heavy for this purpose. The same holds for all other "general purpose" quasi logical languages. One characteristic of QLF [Als92] is the notion of event and state, event-variables and state-variables that allows to abstract over actions or events, and formulate higher order predicates for the representation of the semantics of phrases like "living in Paris is nice" in a systematic way. The World Model Language used in PHLIQA1 is an intermediate language between the English-oriented Formal Language (EFL) and a data base query language. EFL is completely independent of the subject domain (not considering the limitations of the lexicon; they are of course domain dependent). Hence references of words or phrases to objects in the semantic domain are not made in translating the natural language sentences into formulas in EFL. This is done in the second phase: the translation from EFL into WML. This "multi-level" approach clearly distinguishes the several steps in finding the proper semantics of a natural query. Adopting this distinction not necessarily implies that each step is implemented in separate modules that proces in line as it is done in PHILQA1. Unification grammars (we return to unification grammars in a later section) allow to specify the several aspect of semantics (rules or constraints that are "linguistic" and rules that are dependent on the domain) in one and the same formalism. The price to be paid is that words may occur more than once in the lexicon; one occurrence for each semantics. EFL assigns only one constant to each word. Disambiguation is done in a later phase, consisting of the translation into WML and the interpretation of WML-expressions. Most IL's proposed contain lambda-constructs for higher order predicates and set-denotation, and generalized quantifiers for denotation of phrases like "most employees" and "some department". For a motivation of using generalized quantifiers in the IL for a Dutch NLI for data base querying see [Spe93]. Experiments with IL's will have to decide what is an appropriate IL for data base query languages. We have chosen to use context-free grammar rules as the basis for specification of the semantics, and since the semantics of a sentence has to be defined compositionally, this choice may restrict us in the choices we have for an IL.

5 Developing a Grammar

The kernel of a Dutch NLI is based on a grammatical model of the Dutch language. You may think that there is someone somewhere who has such a thing. And indeed

there are but they can't offer you what you want. Either grammars are specified in a dreadful formalism it takes a year to comprehend, or the parser that goes with it, (sometimes the parser is the only "grammar") is written in Prolog or Lisp and after running a complete weekend on "list all the names of employees, that work in London" it outputs 135 parses. (There are people who have similar experiences, see [Ter93] and [Vet94].)

It should be remarked that we are not primarily interested in a "principle-based" grammar, a formalization or implementation of some linguistic theory, like GB. Since we are not primarily interested in describing a natural language, or the linguistic competence of an ideal language user, who knows the grammatical rules of Dutch. Nor are we interested in a "mentalistically motivated" theory of parsing, what ever that may be. We are only interested in methods, techniques and formalisms for the generation of robust NLI's. We decided to write a grammar. Since this is quite a job a tool for developing grammars is needed.

We decided to use our Head-Corner Chart Parser for unification grammars, as the basis for developing the tool for making the front-end. (We call this tool HCP, as long as we don't have a better name.) There is no better reason for choosing this parser than that it is a product of "our own", and that it is the best parser for context-free unification grammars available to us. What do we expect from a tool for developing a specification of a translation from a natural language into some intermediate language, when this translation is to be used for a natural query language? Here is our list of demands.

- The specification language for the grammar and the lexicon should be easy, high-level and declarative.
- The tool offers good diagnostics of syntactic errors in the specification.
- The parser should be fast
- The tool offers facilities to trace the parser for debugging the grammar.
- The tool offers the possibility to test modules of the grammar, like the part of the grammar for NP's.
- The tool offers a standard back-end language for semantics.

In [Ned92] the authors describe GWB, a grammar work bench developed at the Univerity of Nijmegen for affix grammars. HCP is not a grammar work bench in the sense GWB is. HCP doesn't offer the grammar writer technical information about the grammar he writes (like look-ahead sets of non-terminals). Maybe the tool should also be able to generate sentences, a functionality offered by GWB. But, it should be remarked that grammars for NLI may perfectly well overgenerate, or better there is no proper notion of "overgeneration" here (things change of course if you make a syntax-directed NLI based on the grammar).

HCP uses a unification grammar as a specification language for both the syntaxis and the semantics. This language is similar to PATR, except that it has a true context-free grammar backbone. The rules of which have left- and right-hand side symbols with associated feature structures. These structures are defined by a set of feature equations. Feature equations serve two purposes: they constrain the use of a rule in a particular node of a parse tree (static semantics, type-checking, agreement checking) and they are used for building the "semantic feature values" of the nodes of the parse tree from other feature values. These two purposes correspond to the two possible results of unifying the left-hand side and the right-hand side of a feature equation. Either unification is not possible, or unification results in a more informative feature structure. The lexicon is a set of words with associated feature structures.

A tool for developing a front end for natural language interfaces for data-base querying, should not only offer its users a specifi-

cation language for the syntaxis of the natural language, but also a target language, that can be used as an intermediate language for expressing the semantics of the natural language sentences. Notice that the formalism of unification grammars leaves the user free how to specify the meaning of its syntactic constructs. The target language should be general enough to serve as an IL for all kinds of data base contents. On the other hand it isn't necessary to translate all possible meanings of sentences of a natural language into it. We are trying to describe the contents of data bases, not the real world. This means that we "only" have to formalize the syntaxis and meanings of that "part" of our natural language that is used for data-base querying. So, we don't need a complete grammar for Dutch. And we don't want it either because a too large grammar would lay a great burden on the parser.

The functional meaning of a question (i.e. its result) is its answer. We distinguish the *functional* meaning from the *core* meaning of a sentence. The core meaning of the sentence "Which employees are working in Amsterdam?" is the set of all employees that form the semantics of the noun phrase "employees, that work in Amsterdam". The sentence "List all people, that work in Amsterdam." has the same core meaning, although for us it may have a different "meaning". Yes/no-questions like "Does John work in Amsterdam?" will have the same core meaning as its imperative form "John is working in Amsterdam". It is the *mood* of the sentences that makes the difference in their functional meanings, the answer to the question. Sentences like "Are there any men, that work in Amsterdam" will have as core meaning the meaning of the noun phrase "men, that work in Amsterdam". The conclusion is that, apart from the mood of sentences (Yes/No questions, WH-questions, declarative and imperative sentences) we have only two different kinds of sentences. Sentences that have as their core meaning the denotation of a noun phrase (Object Valued Sentences, OVS), and sentences that have a statement as their core meaning (Truth Valued Sentences, TVS). Similar distinctions can be found in the PHLIQA1-report [Phl79]. A sentence will have three semantic features.

- mood
- core, and
- assumptions

The feature assumptions will have as value the implicit assumptions concerning the existence of objects in the world of the data base. If we ask for "the brothers of John" we implicitly assume that there is a person in the data-base world whose name is "John". Using the noun phrase "the father of John" we also assume that there is someone who is the father of "John". Remark that this assumption is not made if we say: "Is there anybody named John?". The latter question should in fact be properly phrased as: "Is there anybody named 'John'?", and the kernel of a robust NLI should be able to see that this is meant. By distinguishing the assumptions from the core meaning we are able to generate better answers to questions: first the assumptions are validated and then, if all assumptions are true, the answer is produced.

Our IL thus has terms for denoting objects as meanings of noun phrases and OVS's, and assertions denoting truth-values for the meanings of TVS's. A grammar writer uses categories like NP, VP, PP and so on. The tool should provide the writer with standard feature structures for these categories. The same holds for the lexical entries: the category of an entry fixes its feature structure.

6 Left- and Head-Corner Parsing

In this section we explain the parser which is used in HCP and which will be used in the NLI as well. A head grammar is a context-free grammar in which each rule is assigned

a head. If a rule is S→NP $\widehat{\text{VP}}$ then VP is the head. S and VP are in the head-corner relation. The head-relation is the transitive closure of this head-corner relation. It is up to the grammar writer to assign heads to the rules of his grammar.

Suppose that S is the start symbol of the grammar, following a head-corner strategy, a parser starts with an assumption—the words on the input form a sentence. This is notified by recording on the chart that S is a goal symbol of the grammar for the whole sentence. Given this goal symbol, the parser determines the possible heads of the sentence[5]. Such a head is a word in the sentence that has a category C that is in the head relation with S. Then all rules are searched for that satisfy the following two conditions:

a) the head of the rule has the category C of the word selected, and

b) the left-hand side of the rule is in the head-relation with the start symbol.

Suppose that A→B $\widehat{\text{C}}$ D is one of the selected rules. A double dotted item of the form [A→B • C • D; i, j] is added to the chart, recording the fact that C has been recognized, i.e. C derives the part $a_i ... a_j$ of the sentence[6].

From this item, we derive two more goal symbols—B and D. These will play the same role as the start symbol in the beginning. The same proces starts but now possible heads are searched for in those parts of the sentence that are to the left (for B) and to the right (for D) of the recognized head with category C.

If B or D has been recognized, then an item is generated in which one of the dots is moved over the recognized symbol. When both dots are at the ends of the right-hand side, then the whole right-hand side has been recognized and thus, the left-hand side symbol A will now play the role of a recognized head.

When the item [S→ •α•, 0, n] is on the chart, then the sentence $a_0 ... a_n$ has completely been recognized and a complete parse has been found.

For a more formal specification of the head-corner chart parser we refer to the papers published elsewhere, like [Sik93, SiA93].

The symbols in the items have associated feature structures and during parsing these structures are built by unification according to the rules in the grammar and the lexicon. Since the number of items can be quite large, the space and time efficiency of the parsing is strongly influenced by the unification algorithm, and by the choice of the data-structures for the chart. In the next section the unification method is considered in some more detail.

Head-corner parsing can be seen as generalisation of left-corner parsing. In the sense that in left-corner parsing, parsing always starts with the left-corner that has been recognized. The advantage of left-corner parsing comes from the fact that it parsers strictly from left to right. Therefore only one index is needed whereas we need two in the head-corner parser. So why should we follow a head-corner strategy? A motive for head-corner parsing is that heads of rules and heads of sentences are those parts of the sentence or the tree that offer information on the basis of which possible rules can be ruled out if you look at the feature-constraints. This advantage, however, only pays off if heads are selected in a clever way. In the current implementation of the head-corner parser there is no top-down filtering by feature-unification.

[5]For each word w in the sentence, and possibly for some combinations of subsequent words in the sentence also (idiom), the lexical analyzer creates one or more (in case of multiple occurrence of a word in the lexicon) start items [i, F, j], where F is a feature structure for the word w if w occurs between positions i and j in the sentence. These start items are in fact initially put on the chart.

[6]If a lexicalized unification grammar is used, with more general trees of depth one in the lexicon, the lexical analyser will put for each word in the sentence this kind of double dotted items on the chart where C is a possible category of the word. Hence, the parser can not only be used for traditional context-free grammars but also for grammars that are completely specified by the lexicon.

This could be possible if symbols that are in the head-relation share their feature structures. Top-down filtering is only done by looking at the head relation. Therefore for most grammars we have seen left-corner parsing is more efficient than head-corner parsing.

The parser doesn't output parse trees but feature-values of the sentence (of course if the grammar writer wants he can define the parse trees as feature values). This implies that sentences that have more than one parse-tree but only one meaning in terms of features values, have only one output structure. Hence, also in case of cyclic grammars we may have only one output structure for an infinitely ambiguous sentence.

7 Efficient Unification

Feature structure unification is generally considered to be a very expensive operation, and, in many implementations of parsing systems it takes more than 80% of the total parse time. Therefore, we have taken special care to search for efficient implementation of this part of the parser.

In several papers, typed feature structures have been proposed to improve efficiency. The idea is that type checking can be done relatively fast and preempts many unifications that would have failed anyway.

In our current implementation, we haven't implemented
typed feature structures, mainly because we haven't had the time. Furthermore, we are using a context-free parsing strategy on top of the unification grammar. Therefore, the top-level feature structures do have a type in the guise of the grammar symbol attached. So, we do have some of the advantages of typed feature structures in our current system. In the future, we probably will implement typed feature structures, for there are many advantages other than efficiency.

The bulk of effort in unification is to do with copying of feature nodes. In a chart parser, we cannot use destructive unification, and therefore a copying scheme must be deviced. Tomabechi [Tom91] states that copying should be prevented for unifications that fail and describes how this can be achieved, without much additional overhead. His strategy is called quasi-destructive unification, and that is exactly what it is. When unifying two feature graphs, the result is built on top of one of the arguments. This is done in such a way that features from the second argument that are not in the first, are added to the first argument, in such a way that they can still be distinguished as being added in the current unification. If unification fails at some point, then the process is simply aborted and the additional features in the first argument are thrown away by a clever increment of a global generation counter. If unification succeeds, however, a copy is made of the first argument before the generation counter is incremented. This copy serves as the result.

Tomabechi's algorithm is considered to be quite efficient, but its virtues have been underestimated, as it allows for a slight modification that results in a major improvement in efficiency of our parser—when copying the first argument, we can share most of the substructures of the arguments. And by doing so, we have actually succeeded in decreasing the number of feature nodes used in the parser by a factor 4 to 10, in some example grammars. Also, because of the reduced copying, the cost of unification is significantly less than in other known implementations. An article in which this unification algorithm is described is forthcoming.

The tables show results from two small experiments for comparing our unification algorithm and the one from [Tom91]. The columns of the first table show: the number of the sentence (see below), the number of nodes created using our unification algorithm, and the number of nodes created using Tomabechi's algorithm, respectively.

nr	HCP	TOM
1	43	526
2	60	894
3	43	520
4	104	1279

In this experiment the following 4 sentences were used.

1. *Waar wonen de broers van Jan?*
2. *Wie zijn de zussen van de vader van Marie?*
3. *Hoe heet de vader van Marie?*
4. *Is de zus van Jan de tante van Marie?*

These 4 sentences were parsed with respect to a grammar for a small fragment of Dutch, having 30 context-free rules, with 5 features rules per production in the mean, and 3 right hand side symbols in the mean. The lexicon contains about 100 words. All these sentences are processed by HCP in less than a second.

n	parses	HCP	TOM
4	5	100	542
5	14	249	1.827
6	42	662	6.268
7	132	1897	22.187
8	429	5799	80.685

The second table shows results from parsing the "sentences" Jan^n, for $n = 4, 5, 6, 7$ and 8, respectively. These sentences were parsed using a grammar with the two rules $S \rightarrow S\ S \mid Jan$, so these sentences are rather ambiguous. The semantics defined in this grammar associates the parse trees to the sentences. The second column of this table shows the number of parses. For the last sentence, i.e. Jan^8, HCP outputs the 429 parse trees within less than 30 seconds.

8 Robustness of NLI's

More and more people seem to get convinced of the fact that even the most sophisticated model of a natural language can only partly account for the rich nature of natural language. Sooner or later a user types in a sentence the system can't handle in a correct, meaningful sense. This has led to the interest in what is now called "robust parsing". There is a lot of knowledge involved in the process of grasping the meaning of a question or an utterance as it was intended by the user. We use knowledge of the context, the dialogue for instance, we use also our knowledge about the speaker, in order to filter the relevant information from the utterances. Also filtering of noise in spoken language is an aspect of robust processing. Robust parsing aims at incorporating these different "knowledge recources" in finding the appropriate (intended) meaning of the utterance. The interest in (and the urge for) robust parsing comes from the principal shortcomings involved in any process of formalisation of natural languages and their use. In section 3 we have formulated some robustness requirements for NLI's. How can these requirements be met? What kind of solutions have been proposed for dealing with the problem of lack of robustness in existing NLI's?

Some people have advocated the use of statistical data in order to get more robust analyses of utterences. Hence, the interest and motivation for studying stochastic grammar models, Hidden Markov Models, and statistics on the occurrences of trigrams of syntactical categories. It is not clear whether these stochastic models could offer a solution for the robustness problem.

> "... on their own PCFGs [i.e. probabilistic context-free grammars] probably are not very useful in syntactic disambiguation.",

Charniak says in his book on statistical language models, [Cha93]. This also holds for more sophisticated probabilistic grammar

models of natural languages than the classical models studied in Charniak's book, as for instance DOP [Bod93] or weakly restricted CFGs [Doe94]. You need semantic information in addition. And very large corpora!

Sometimes the problem of robustness is almost identified with the problem of "overgeneration", the phenomenon of a parser that presents 135 parses for a simple sentence that has only one meaning (or maybe two). Stochastic grammars can be used to deliver the most likely parse of a sentence. But sometimes you don't want the likely one, and if you always want this one why not remake the grammar, so it only outputs this one? Overgeneration is not the most important problem and it isn't even a robustness problem in itself. We reserve the term robustness to describe a quality or property of a system (like a natural language interface)—how well does it behave in case of unpredicted input. In that sense, you cannot say that a grammar that produces 135 analyses for a simple sentence is not robust. It depends on what the grammars is used for. A user of an NLI will say that it is not robust, if he types in a good sentence but the parser cannot recognize it or, he makes a typing error or a grammatical error and the parser cannot recognize this error and restore it, while human beings have no problem in recognizing the intended meaning. One can, of course, extend the formal grammar so the parser can handle more correct sentences and also most frequent typing errors and grammatical errors, but this is unfeasable for larger domains.

A solution to some problems of robustness consist of skipping unparsable parts of a sentence. Then the problem is to find a most complete parse of the sentence, i.e. a partial parse that covers most of the sentence. The idea is that this partial parse may offer enough information about the semantics of the sentence. To this end existing parsers are "made more robust": they skip unparsable parts of the input and resume parsing as soon as possible. Pure bottom-up parsing, like GLR-parsing (Tomita) or a generalisation of CYK-chart parsing for general context-free grammars, is the best strategy for this kind of robust parsing, although the results are not very promising for reliable NLIs. Head-corner parsing is bad for obtaining the best partial parse because only heads that are predicted top-down are recognized. The same holds for left-corner parsing. They don't build as much incomplete parse trees as a pure bottom-up parser. We think that for particular domains we can use head-corner parsing and extend the grammar for dealing with common "grammatical" errors that do not introduce semantical ambiguities.

9 Plans for the Future

We will continue to work on developing an NLI and on HCP, the tool, simultaneously. HCP is being used for developing natural language front-ends for database querying for educational purposes. Problems discussed are: how do you specify a small fragment of a natural language by means of a context-free unification grammar and how can you specify the semantics of the sentences of this fragment in a systematic way? Special attention is paid to reusability and extendibility of the grammar. The system is also used for the Plinius project at our department. This project aims to develop a system able to acquire knowledge semi automatically from natural language abstracts about the mechanical properties of ceramics. For the reasons that have led to choose for using HCP and for more information about the Plinius grammar and this project in general we refer to [Vet94].

HCP is not in its final form, although rather efficient it hasn't a user-friendly interface yet. The grammar writer is not supported in writing well-structured orthogonal grammar specifications. Using a unification system with typed feature structures could be a good step in this direction. Standard types for agreement features and even some standard typed semantic features (related to a standard intermediate formalism)

can help the grammar writer in writing well-typed and well-structured grammar specifications. As a side-effect the introduction of standard typed feature structures may have a positive effect on the efficiency of unification ([Car91, Car92]). As far as the specification language is concerned, the tool doesn't support disjunct feature structures. The question is whether the disadvantages of disjunct feature unification (complexity, [Kas87], [EiD88]) outweights the advantages for the grammar writer. HCP not only allows us to develop grammars but also to study several candidates for the intermediate language to be used in an NLI.

There is no morphological analyser, no error-correcting phase implemented and so in using small prototypes currently all word forms have to occur in the lexicon and have to be typed in without errors.

Although we haven't said anything about answer generation it is an important part of a practical NLI. Also this part will be the subject of further investigation. A next step towards a natural language dialogue system then consist in looking at problems introduced by offering the user the possibility to refer to phrases mentioned in a previous query: "can you give me more information about these things?"

References

[Als92] Alshawi, H. (ed.) (1992), *The Core Language Engine*, The MIT Press, Cambridge, Mass., 1992.

[Bod93] Bod, R. (1993). Data Oriented Parsing as a General Framework for Stochastic Language Processing. *Twente Workshop on Lang. Techn. 6* (*TWLT6*), 37–46.

[Bov93] Boves, L. (1993). Spoken Language Systems: An Overview. *Twente Workshop on Lang. Techn. 5* (*TWLT5*), 9–13.

[Car91] Carpenter, B., C. Pollard and A. Franz
(1991). The Specification and Implementation of Constraint-based Unification Grammars. *Proc. Second Internat. Workshop on Parsing Technol.*, 143–153.

[Car92] Carpenter, B. (1992). *The Logic of Typed Feature Structures.* Cambridge University Press.

[Cha93] Charniak, E. (1993). *Statistical Language Learning.* A Bradford Book, The MIT Press, Cambridge Mass., London, England.

[Doe94] ter Doest, H. and R. op den Akker (1994). Weakly Restricted Stochastic Grammars. *15th Internat. Conf. Comput. Ling., COLING'94* , 929-934. (see also: Ta! nr. 4 jaargang 2, 84-98)

[EiD88] Eisele, A. and J. Dörre (1988). Unification of Disjunctive Feature Descriptions. *Proc. 26th Annual Meeting of the Association of Computational Linguistics*, 286–294.

[Kas87] Kasper, R.T. (1987). A Unification Method for Disjunctive Feature Descriptions. *Proc. 25th Annual Meeting of the Association of Computational Linguistics.*

[Ned92] Nederhof, M.-J., C.H.A. Koster, C. Dekker, A. van Zwol (1992). The Grammar Workbench: a first step towards lingware engineering. *Twente Workshop on Lang. Techn. 2* (*TWLT2*), 103–115.

[Phl79] Bronnenberg, W.J.H.J. et al. (1979). The Question Answering System PHLIQA1. In: L.Bolc (ed.), Natural Language Question Answering Systems. (Natural Communication with Computers, Vol. II) Carl Hanser Ver-

lag, Muenchen, Wien; Macmillan, London, 1979.

[Shi86] Shieber, S.M. (1986). *An Introduction to Unification-Based Approaches to Grammar. CSLI Lecture Notes* 4, Center for the Study of Language and Information, Stanford University, Stanford, CA.

[Sij93] Sijtsma, W. and O. Zweekhorst (1993). Comparison and Review of Commercial Natural Language Interfaces. *Twente Workshop on Lang. Techn. 5 (TWLT5)*, 43–57.

[Sik93] Sikkel, K. (1993). *Parsing Schemata*. Ph.D. Thesis, Dept. of Computer Science, University of Twente, Enschede, the Netherlands.

[SiA93] Sikkel, K. and R. op den Akker (1993). Predictive Head-Corner Chart Parsing. *Int. Workshop on Parsing Technologies (IWPT'93)*, 267–276.

[Spe93] Speelman, D. (1993). A Natural Language Interface that uses Generalized Quantifiers, *Twente Workshop on Lang. Techn. 5 (TWLT5)*, 69–74.

[Ter93] Stefanova, M. and W. ter Stal (1993). A Comparison of ALE and PATR: Practical Experiences. *Twente Workshop on Lang. Techn. 6 (TWLT6)*, 47–62.

[Tom91] Tomabechi, H. (1991). Quasi-Destructive Graph Unification. *Proc. 29th Annual Meeting of the Association of Computational Linguistics*, Berkeley, 315–322.

[Vet94] Van der Vet, P.E. et al. (1994). Plinius intermediate report. Memorandum Informatica 94-35, University of Twente, Enschede.

[Vos91] Vosse, T.G. (1991). Detection and Correction of Morho-Syntactic Errors in Shift-Reduce Parsing. *Twente Workshop on Lang. Techn. 1 (TWLT1)*, 69–77.

[Vos94] Vosse, T.G. (1994). *The Word Connection; Grammar based error correction in Dutch* . Uitgeverij Neslia Paniculata, Enschede.

SCHISMA: A NATURAL LANGUAGE ACCESSIBLE THEATRE INFORMATION AND BOOKING SYSTEM.

G.F. v.d. Hoeven, J.A. Andernach, S.P. v.d. Burgt, G-J.M. Kruijff, A. Nijholt,
J. Schaake, and F.M.G. de Jong.
Parlevink Group
University of Twente, Dept. INF/SETI
P.O. Box 217
7500AE Enschede, The Netherlands.
e-mail: {vdhoeven,andernac,stan,kruijff,anijholt,schaake,fdejong}@cs.utwente.nl

Abstract

This paper gives an overview of activities in the SCHISMA project now and in the near and further future. Main points of discussion on current activities are collecting data on dialogues, dialogue analysis and dialogue modelling. Some attention is also paid to experiences using a the commercially available product Natural Language.
In the discussion of future plans and developments, topics such as speech, integration of speech and language, and the possible contribution of neural networks and fuzzy logic in the project are considered.

1. INTRODUCTION.

SCHISMA is a collaborative project of the University of Twente and PTT Research. The participating groups are the Parlevink group of the Department of Computer Science in Twente, and the speech and language group at PTT Research.

The aim of the project is to develop a prototype of a natural language dialogue system. The envisaged system is capable of providing a user with information about theatre performances, and it should allow the user to book seats for such performances. In addition to the goal of building a prototype of some quality there is the equally important goal of gaining a deeper insight in the problems one encounters in the process of building a natural language dialogue system. Getting experienced is considered one of the prerequisites for a successful follow-up of the project.

In SCHISMA *dialogue management* has been identified as the key issue to be addressed. As a consequence emphasis is put on the proper combination of the various sources of information, on the current status in a dialogue, on the process of judging that status, and on the subsequent steps. Parsers, semantic representation formalisms, knowledge representation formalisms, etc. are tools to be used, rather than products to be delivered. Of course improving these tools could also lead to improvement of the capability of our system. But in our view the key to enhanced functionality and better performance lies with better dialogue control.

Characteristic for SCHISMA in comparison to projects with similar objectives is the choice of the application domain: theatre performances and bookings. Furthermore one could probably characterize SCHISMA as 'eclectic': it is tried to find the proper mix of ideas, new ideas are developed only when necessary, and special attention is given to integration.

In this paper an overview will be given of major issues in current SCHISMA research. The project plan distinguishes between three related lines of research and development, each with their own deliverables.

1. **The collection and analysis of a corpus of real dialogues.** This aspect of the project will be addressed in sections 2 and 3 which contain a discussion of the Wizard of Oz environment that was built, a discussion of various forms of dialogue tagging, and of our views on obtaining a useful corpus of annotated dialogues.
2. **Investigating notions of dialogue state and state transitions.** Next topic, discussed in sections 4 and 5, is dialogue control and functionality of a natural language dialogue system. Under this heading there is attention for on a suitable notion of discourse semantics and representation,

and for the possible role of discourse transition matrices.

3. **The realisation of a theatre information system with the use of a commercially available tool set for building natural language interfaces to databases.** As is discussed in section 6, the exploration of the possibilities and shortcomings of available tools was considered useful. In addition, it could provide a possibility for a comparison with the envisaged SCHISMA prototype and demonstrate the added value of real dialogue over question answering.

The paper concludes with an outline of future developments in section 7. Here it will be argued that SCHISMA is suited for aims which are less restricted than the current aims formulated above, and that it may turn into a much broader research umbrella, with more and continuous attention for issues like speech, speech and language integration, robustness and neural and fuzzy approaches.

2. THE WIZARD OF OZ EXPERIMENTS FOR SCHISMA.

As mentioned above, one of the SCHISMA objectives is to conduct Wizard of Oz experiments in order to obtain data on dialogues between a human user and a (simulated) automated natural language dialogue system. Stated more precisely, the objectives of the experiments are threefold:

1. to gather dialogues which can serve as a basis for the design of a dialogue manager for our prototype man-machine dialogue system
2. to develop and refine dialogue models derived from dialogues obtained from a relatively free subject-wizard dialogue
3. to find out how the subject-wizard dialogues are evaluated by the subjects compared to normal man-man dialogues, with regard to robustness and user-friendliness.

2.1. GENERAL DESCRIPTION.

In order to conduct the experiments, one needs people to participate and a wizard environment to work with. The involvement of humans will be discussed first.

There are four actors involved in each experiment session: a subject, the subject's instructor, the wizard's assistant and the wizard. The instructor and the assistant have auxiliary roles. The main roles are played by the subject and the wizard.

The subject is the person who acts as (or is) a client using the system. The subject is not aware of the fact that there is no real automated system, and that (s)he is communicating through the Subject's Interface with a person that provides the system's functionality. The number and kind of subjects depend on the experimental phase (see below).

The wizard is the person who simulates the functionality of the system. The simulation involves

- the elicitation, recognition and understanding of the user's utterances,
- acquiring the data needed to query the database,
- interpreting the output of the database query,
- formulating adequate responses to the user's utterances.

The wizard is one of the SCHISMA researchers, who is supported in his task by the Wizard Interface.

Conducting dialogues in a wizard environment through a wizard has an additional goal, which involves especially the wizard and his assistant. Besides simulating the system's functionality, the wizard and his assistant are also supposed to monitor the wizard's behaviour in order to provide input for the design and development of the automated dialogue manager. One way to obtain relevant data in this respect, is to let the wizard work under various restraining conditions. For example, the wizard can be given the instruction to follow a restrictive protocol that forbids him to reply to a question with another question, or to select all his reactions from a very restricted set of possibilities, etc.

Another parameter in the experimental setting, is the wizard environment. It consists of the Wizard Interface and the Subject Interface. The Wizard Interface helps the wizard to control the dialogue simulation session. The Subject Interface allows the subject to have a keyboard conversation with the wizard.

More specifically, the Simulation Environment allows both the wizard and the subject to enter, edit and transfer their utterances. It lets the wizard select for its responses (partially) pre-defined utterances, enter query data and execute queries. Furthermore, it keeps track of the course of the dialogue by storing in a log file all utterances, their time stamps, and some other useful information,

like whether wizard's utterances were selected from a menu, or self-invented and typed.

The screen of the Wizard Interface is composed of a number of windows: The most important ones are the wizard-subject communication window and the database interface window. The figures 1–5 and their captions show these windows and explain the operations they allow.

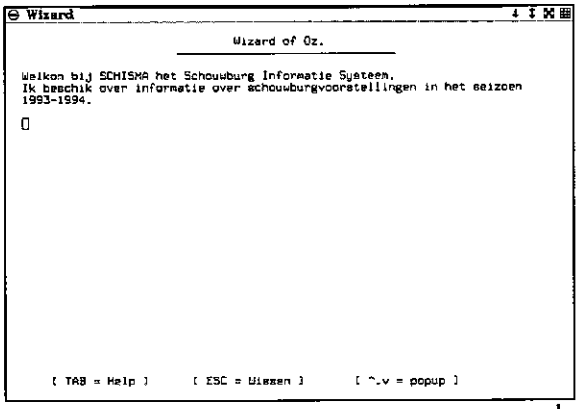


figure 1: initial wizard-subject communication window[1]

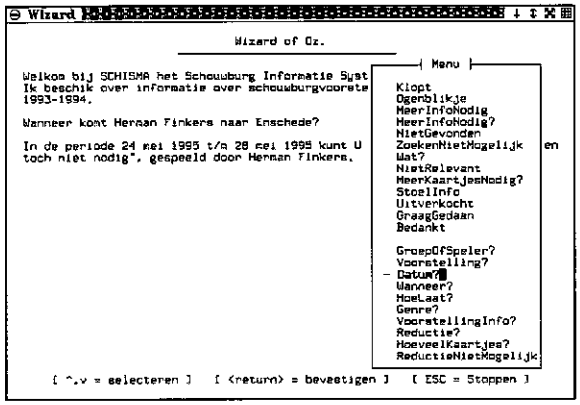


figure 2: wizard-subject communication window with pop-up menu of possible utterances.

The Subject Interface has one window, which looks a lot like the wizard-subject communication window the wizard has. In this window both the subject's own utterances and the wizard's appear. The wizard's utterances are displayed per utterance to the subject to obtain a computer-like way of presenting utterances. The subject does not have the option to select an utterance from a menu of canned sentences, as the wizard has (cf. figure 2).

[1]Figures 1 to 5 were produced by Jos Buis, who designed and implemented the wizard environment.

All programs are written in C. The Subject and Wizard Interface both use the socket mechanism to exchange information.

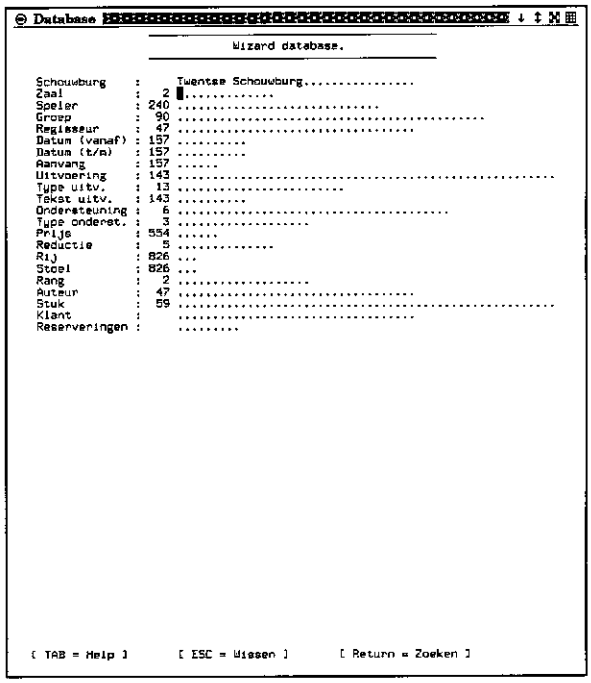


figure 3: the database interface window in its initial state.

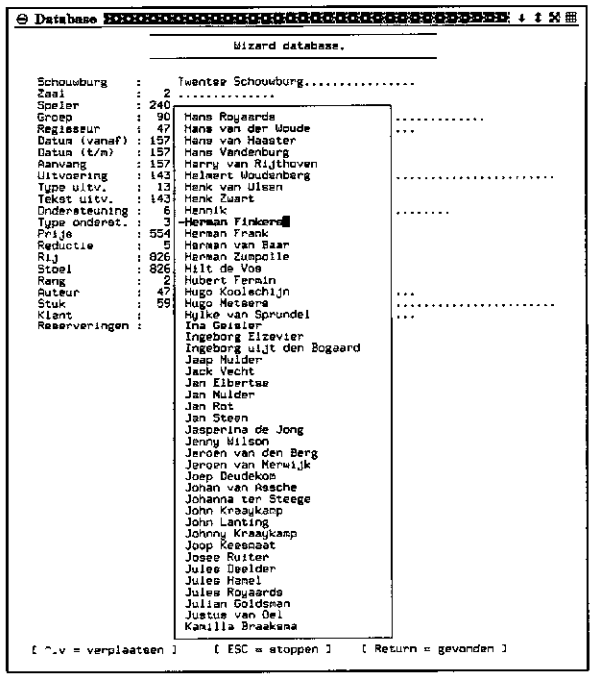


figure 4: the database interface with pop-up menu of possible actors.

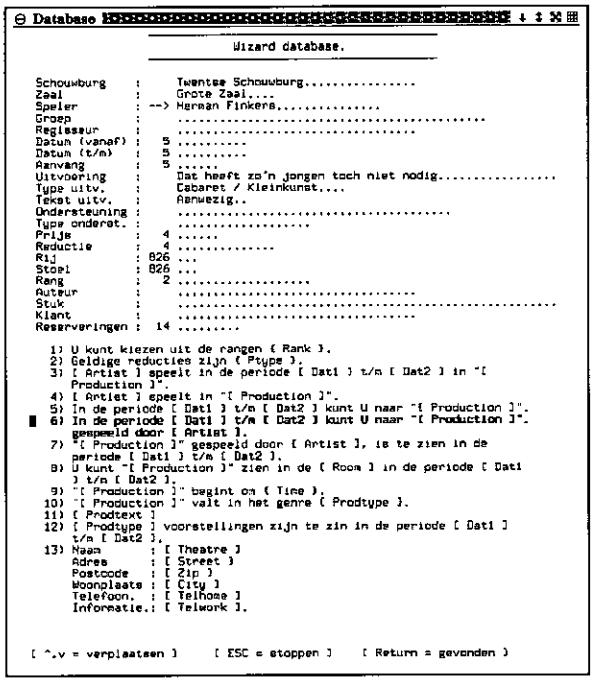


figure 5: the database interface window with templates for utterances the wizard can choose from.

In addition to its contribution to collecting a corpus of relevant dialogues, the Simulation Environment serves another goal in the project. It is implemented in such a way that it can easily be extended with components which take over some task of the wizard.

E.g. the wizard's task may switch from providing data on a fill-in form for database queries to providing data on a fill-in form for parse results. For this purpose, a component has been added to the environment that may turn the parse result into the intended query. This component could be one that is designed for the completely automatic system. Thus, the Simulation Environment is also a testbed for parts of the final system.

2.2. SET-UP OF EXPERIMENTS.

It will be clear that it is not an easy task to let experimental dialogues evolve smoothly. Every experiment will take 'tuning' of environment as well as wizard, wizard's assistant and instructor in a number of phases.

Also the order of experiments is crucial. Guyomard and Siroux explain in [11] how they conducted spoken Wizard of Oz experiments in two steps. In the first step of their experiments, data were gathered using a strict dialogue model and in the second phase those data were used for a more free dialogue model. In SCHISMA a different order is chosen. Insight in the more varied linguistic behaviour of subjects under a more free condition is considered a prerequisite for the design of dialogue models. So, first there will be an experiment in which no dialogue models are used and then an experiment in which a stricter dialogue model is used which is derived from the previous experiments.

Each experiment consists of the following phases:

The pre-experimental phase. In this phase, the experimental set-up was determined and the Simulation Environment was designed and implemented. The scenarios to be used in the pilot experiment were written, as well as the instruction sheets, which should both help and constrain the subject in his dialogue with the wizard.

The pre-experimental tests were carried out with the Computer Science Department of the University of Twente as the test site for the wizard and his assistant, and PTT Research as the test site for the subjects and their instructor.

The pilot phase. The pilot phase will be carried out at the University of Twente and several other universities. It serves the purposes of testing the Simulation Environment on the ease to use it, the consequent behaviour it exposes and on the kind of dialogues it yields, and letting the wizard get acquainted with the Simulation Environment.

The main experimental phase. In this phase, the actual dialogue sessions take place. These sessions will be carried out in a realistic environment in which the subjects are people who actually planned to inquire about a theatre performance or to make a reservation.

The analysis phase. In this phase the subject-wizard dialogues will be analyzed on several levels. The section below is entirely devoted to this subject.

3. THE ANALYSIS OF DIALOGUES.

In the analysis of dialogues we will distinguish three aspects.

1. Identification of the components a dialogue is built from (things like utterances and turns).
2. Tagging (the components of) dialogues.
3. Evaluating the annotated corpus, validating or rejecting hypotheses about dialogue structure, developing new hypotheses about dialogue structure.

In this section we shall not discuss the third aspect. We concentrate on aspects 1 and 2.

3.1. DIALOGUE COMPONENTS.

The first aspect of analysis is not really a separate analysis issue. It influences decisions that are made in implementing the wizard environment, and it is in turn influenced by such decisions.

We shall not dwell upon the decisions about what is an utterance and what is a turn. Let us simply state that we have taken a very pragmatic stand point here, relying on the fact that we deal with keyboard input. An utterance is anything that ends with pressing the return key. A turn is defined in terms of the time stamps associated to consecutive utterances. If no utterance of the other party is timed in between, then two utterances belong to the same turn. The wizard environment provides the timing information.

```
---Dialoog.Jos Buis.941118.03
---Version 0, created by the system.

{!client_age 20}
{!client_gender M}
{!client_assignment 3}
{!client_start_time 15:54:08}

C.1.1:
wanneer zijn er musicals?
{time 15:54:18}

S.2.1:
{generated}
Musical voorstellingen zijn te zien in de periode woensdag 28 december 1994 t/m vrijdag 19 mei 1995.
{time 15:54:46}
{continue 1}
{generated}
U kunt naar de volgende musical voorstellingen : Bubbling Brown Sugar, Candide, Little voice, Sweeney Todd, The marriage of Figaro en Under a blue Roof.
{time 15:55:19}

C.3.2:
wanneer is candide
{time 15:55:49}

three turns deleted

C.7.4:
reserveer vier kaartjes voor candide
{time 15:57:14}

S.8.4:
{generated}
De Grote Zaal heeft de volgende rangen : 1e balkon 1 en zaal 1. Welke rang wilt u zitten?
{time 15:57:28}

C.9.5:
zaal 1
{time 15:57:37}

S.10.5:
{generated}
Dit theater kent de volgende reducties : CJP, Geen, JTK en Pas 65. Heeft u een reductiekaart?
{time 15:57:46}

C.11.6:
nee
{time 15:57:52}

S.12.6:
{edited}
De totale kosten zijn f 104,00. Moet ik de kaartjes reserveren?
{time 15:58:28}

C.13.7:
ja
{time 15:58:34}

S.14.7:
{generated}
U heeft gereserveerd voor "Candide", gespeeld door Voltaire op woensdag 1 februari 1995. Aanvang van de voorstelling is 20:00. U wordt verzocht deze kaartjes
minimaal een half uur voor aanvang van de voorstelling af te halen.
{time 15:58:39}

{!total_time 00:04:59}
{!total_turns 14}
```

figure 6: a dialogue as logged by the Wizard.

In figure 6 a part of a subject-wizard dialogue is shown. One may notice that turns are distinguished by a header line which signals who the 'speaker' was, namely S for system or C for client. The header also shows the turn count, in total as well as per speaker. A time stamp is given at the end of each utterance, if two or more utterances belong to the same turn there are additional *continue* labels. Every utterance of the wizard (system) is marked as either typed, or selected from a menu. The figure shows the contents of the text file exactly as it was logged by the wizard environment.

There are more components to be distinguished in a dialogue: there are turns and utterances, but also sentences, phrases, and words. It suffices here to say that our definition of these concepts must take into account that we deal with keyboard dialogues. A phrase which is just a single question mark (as a cry for help) must be considered as well.

3.2. DIALOGUE TAGGING.

Assigning annotations or tags to dialogues is important for other parts of the project. We mention three grounds for our interest in obtaining a tagged corpus of theatre information dialogues.

Firstly, without annotations it is impossible to move to the third aspect of analysis: developing dialogue models and testing hypotheses about dialogue structure.

Secondly, the annotated dialogues can serve as material to run tests of parts of the final system on.

Finally, some parts of the system will not run without data obtained from the final system. This is in particular the case for the discourse transition matrices that we discuss in section 5.

In this paper any tagging that has to do with the lower levels in the hierarchy of components, such as syntactic tagging, will be neglected. Such tagging is interesting and relevant, but the emphasis in this paper is on dialogue management, so here we look only at dialogue structure tagging.

The tagging related to dialogue structure is found mainly at the utterance and turn level. Some of these utterance and turn tags will be considered now in more detail.

One kind of dialogue structure tagging is derived from the topic-focus approach, and tries to capture something like *subject matter* of an utterance. The tags are highly domain dependent, there is a limited number of attributes which qualify as possible subject matter in a relevant dialogue. Things that count are *performance, place, date, time, number of seats, rank, price* and a few more. The tags are sets of such attributes. Dialogues with this subject matter annotation will be used in three ways.

Firstly, it is assumed that all dialogues will show a common progression in subject matter. Stated otherwise, the hypothesis is that there is a preferred order of dealing with the relevant aspects of the booking topic. The availability of a tagged corpus will help to gain insight on this hypothesis. We return to this preferred order in section 5 below. Note that we talk here about an ordering among subject matters which are all equally possible as next subject when viewing the 'information state' and the current 'theme' in the dialogue. Restrictions on the choice of a next subject that have to do with the status of available information are discussed in section 4.

Secondly, the final system must be capable of deriving the subject matter of an utterance, possibly using various sources of knowledge. The components of the system which should perform this task can be tested if an annotated corpus is available.

Finally, although this is dependent on the development of our view with respect to the 'preferred order of subject' hypothesis as formulated above, we envisage that counts of subject changes will appear in discourse transition matrices in the final system. This point is further elaborated in section 5.

Another kind of dialogue structure tagging is the initiative-response tagging. It has been pointed out by others (Bilange [5], Stubbs [19]) that every dialogue can be subdivided into initiative-response, or maybe initiative-response-confirmation units, and that such units could be viewed as the minimal constituents of a dialogue. Part of dialogue analysis is finding and marking these constituents. We do so by tagging at the turn level for initiative or response. This kind of tagging has been applied and extensively investigated by others also, in particular by Dahlbäck (cf. [1], [6] and [7]).

The objectives we pursue by providing this kind of tagging are largely the same as the ones we mentioned before. We return to the issues of initiative and response in section 5.

There are yet other kinds of tagging, at the turn and utterance level, that we consider, but we shall refrain from giving a complete list. In fact there is no complete list at the moment. A serious effort at tagging dialogues has started only recently, and the hypotheses about regularities and irregularities in

dialogues, as well as the variety of possibilities for tagging and their relevance are subject of discussion.

4. THE INFORMATION CONTENT OF UTTERANCES AND THE INFORMATION STATE OF A DIALOGUE.

One could say that the remarks in the previous section about the analysis of dialogues are superficial in the sense that they deal with surface structure only. So far we have avoided any discussion on semantics, intentions, or plans. To put it bluntly: so far we have not tried to capture in any formal setting or model what the dialogues we consider are really about. This aspect of functionality and its formal model is the theme of this section.

In general, the functionality of a dialogue is described as a series of transitions with respect to a finite set of possible states (cf. [2,5,14,21]). Each state corresponds to some intentional state of the actors: being (dis)satisfied by an answer, having a question, believing something not known by the other actor to be true, etc. According to this approach, the purpose of a dialogue will be the fulfilment of the intentions of the actors raising and raised by the discourse.

In our experimental system for theatre information and booking, the continuation of the dialogue also depends on the information state resulting from the most recent dialogue act. The identification process of a performance is less straightforward than a sub dialogue identifying a train or a flight. Being aware of the vagueness and indistinctness of the keywords related to this domain, users utter additional information in order to check whether they were right in their expectations. For example, instead of using the identifying phrase *the performance next Tuesday* or *the premiere of The Tempest*, users utter *the **opera** next Tuesday* or *the premiere of **Shakespeare's** Tempest.*

Not only the user, however, but also the system is aware of the vagueness of some technical terms. For instance, when selecting the seats the user would desire, the system first asks in what part of the theatre the seats are to be located and consequently presents a proposal. It turns out that in this particular domain it is far less clear when the selection criteria to be given by the user have already identified a unique performance or not and that appropriate action has to be performed as soon as certain criteria do exclude each other.

This analysis leads to an approach towards the functionality of dialogues in which information states play a more important role than in the approaches generally based on Searle's Speech Acts theory ([17]). In this theory information states have been simplified to the felicity condition that with respect to that particular information state the current speech act may be performed. Only a change of this condition, not a change of the information state, is represented and plays a role in determining the sequel of the dialogue.

It will be shown that the role information states play cannot simply be added to a speech acts based approach. In order to represent the role of information states, we must use information state changes as the foundation of our approach and regard the purpose of a dialogue to be the transfer of information. Therefore, the functionality of discourses, from the point of view of their objective, can be described in terms of conveying information. This change of approach gives rise to at least the following questions:

1. what is information and what exactly is the difference from intentional states?
2. in what way is information conveyed?
3. how is such information dealt with?

These questions will be subject of the following subsections.

4.1. WHAT IS INFORMATION?

At first sight it seems to be very odd that the term 'information' can be used both for *secure knowledge* (common use) as well as for an *increase of certainty* but still related to some probability and uncertainty (technical use). Both uses, however, are based on the common notion that information is knowledge to act upon. Knowledge, we are not entirely sure of that it is true, but which we assume to be so.

In this sense, Groenendijk et al [10] are right in perceiving the increase of information both as an increase of things we have to know something about as well as a decrease of possible state of things by knowing more about those things.

The contemporary use of the term 'information' originates with the American philosopher C.S. Peirce (1839–1914) ([15]). Confronted with the Kantian rule that the extension and intension of a term are reciprocal (that is, the more attributes a term has got, the less objects it refers to, and vice versa),

Peirce states that according to that rule we are not able to learn something new.

According to him, we must be able to increase the number of attributes (depth), with respect to the same number of objects (breadth), while, on the other hand, it must also be possible to increase the breadth the depth remaining unchanged. The result of both processes will be an increase of information. And so, 'information' is defined by Peirce as *the amount of comprehension (=intension) a symbol has over and above what limits its extension.* That is to say, information, as defined by Peirce, consists of attributes which are additional with respect to the attributes needed in order to determine the selected data. Combined with the Peircean formula

Extension × Comprehension = Information

expressing his rejection of the Kantian formula

Extension × Comprehension = Constant

according to which indeed the attributes contained by a predicate will be represented by the selected extension properly, we have to represent information as a tuple containing both a set of attributes and a set of entities the attributes may be predicated to.

4.2. In what way is information conveyed?

In order to get a clear view on the matter, one needs to distinguish between the manners in which information can be conveyed, and the way in which such information is actually managed. Let us first attend the issue of the manners in which information can be conveyed.

Central to our treatment of the communication of information are the familiar notions of *topic* and *focus*, as developed by the Prague School of Linguistics. Topic, in general, signifies the already available, or 'given', information, while focus expresses additional, or 'new', information.

Due to the articulation of topic and focus in a sentence, and the relations between topics and foci of different sentences, a certain story-line or thematic progression is developed in the discourse.

Four kinds of thematic progression are distinguished by Daneš[2]:

1. Sequential progression: $T_1 \rightarrow F_1$, $T_2 \rightarrow F_2$, ...; T_2 is constituted by F_1
2. Parallel progression: $T_1 \rightarrow F_1$, $T_2 \rightarrow F_2$, ...; $T_1 \approx T_2$, i.e. are highly similar
3. Hyper theme: $T_1 \rightarrow F_1$, $T_2 \rightarrow F_2$, ...; T_1, T_2 refer to hyper theme T_H

The interesting fact of defining the kinds of thematic progression as such, is that we arrive at building blocks for the communication of information. Each block is characterized by conditions, induced information change, and a relation between the sentences. The relation indicates whether the information is negative or positive, while the conditions state the applicability of the building block – i.e. whether a certain kind of information can actually, sensibly, be communicated in a certain situation. This is outlined in some more detail in [13].

4.3. How is such information dealt with?

As already stated above, the dynamics of interpretation can be perceived to lay in the change of information conveyed by the discourse. In their theory of Dynamic Semantics, which is currently under development, Groenendijk et al. [10] emphasize that the dynamics of interpretation should be reflected in the formal representation. In particular, meaning is to be understood as information change potential. According to Groenendijk et al, the communicated information is strictly positive. Hence, each utterance in the discourse is conceived of as an update of the hearer's information. Revisions or downdates are not allowed, which might be marked as a first shortcoming of Dynamic Semantics. The formal representation of Dynamic Semantics, for which we refer to Groenendijk et al. [10], is characterized by the use of possibilities making up information states.

The possibilities intuitively stand for those alternatives open according to the present information. Formally, possibilities consist of pegs each of which is associated to an object in some possible world. A second shortcoming might thus be noted. The predicates, of which mentioned objects are the subjects, are not represented in the possibilities. Holding on to the original ideas of Groenendijk et al, we have proposed a revision of the representation of Dynamic Semantics that is based on Peirce's logical theory of information ([16]). For our revised representation we define a similar notion, namely that of

[2]The fourth kind, Prosody is not defined very clearly by Daneš and hence it will not be treated in the sequel of this section.

the spot, consisting of a set of attributes and a set of entities the attributes are predicated to.

Each predicate has an epistemological modal attached to it, where ◊ is a *may-be* or *possibility*, O an *is* or *actuality*, and □ a *should-be* or *tendentiality*. By these modals we are enabled to distinguish the essential and additional predicates as well as the constraints related to the domain.

Rather than being an index to an entity perceived actual in a possible world, the spot is a symbol, that matures by comprehending more accurate information about the world discoursed of. By 'more accurate information' we understand information states arrived at by using a combination of updates and downdates, depending on the discourse. Hence, discourse is perceived of as a process of conveying information being either positive or negative. Recall that Groenendijk et al state that discourse is a process of conveying strictly positive information. Our perception leads to a series of definitions concerning information states, updates, downdates, revisions, etc. which is rather similar to, though more extensive than, Groenendijk's.

5. THE GUIDING ROLE OF DISCOURSE TRANSITION MATRICES.

In the previous section we talked about the functionality of a dialogue in terms of a set of possible states, and transitions between them. It was pointed out that state in SCHISMA dialogues is largely information state. Ways of representing such states, with emphasis on the state change potential, were proposed and discussed.

In this section we leave the functionality aspect and concentrate more on form. The main observations of this section are the following.

Firstly, state in our dialogues may be, as we saw, largely information state, but it is not exclusively so; there are other relevant aspects to the notion of current state.

Secondly, it is worthwhile to consider and represent the other aspects of state independently of the more important information aspect; determining and effectuating state transitions is in our view the process of determining the optimal combination of two or more independent choices.

So we return to states and transitions, but we look at other aspects and, for the moment, ignore the aspect of information state and conveying information.

One could say that what we talk about here are the protocols that play a role in conducting a, or even any, dialogue. Things that fall under this heading are:

the natural sequences of initiative and response and the disruption of such sequences,

the choice of subject and the ordering of subjects to be discussed,

etc.

One can imagine how large parts of a dialogue with SCHISMA consist of simple question answer pairs, where the initiative (question) lies with the user, and the system's behaviour is purely responsive (answering). But once it has become clear for what performance on what day the user wants a booking, the initiative and response roles may interchange. Now it can be the system questioning the user about his preference for seats in the theatre, or for his possible rights on a reduced price. There are other points in a conversation where the initiative may change from one partner to the other. E.g. questions by the system about preference for seats could well be countered by the client posing a question about prices first. The other way round, an open question by the client about performances this season could lead to a counter question by the system, urging the user to be more specific. The same may happen if the client poses a question which is too vague, or one that is in some way inconsistent in itself.

It is essential that the system is constantly aware of which partner holds the initiative in the dialogue, and how the initiative changes. This will help understanding as well as generating utterances. It may also help to prevent dialogues going wrong because both partners start claiming the initiative, or both partners keep refusing to take the initiative.

Moreover, being aware of who has the initiative is important for a consistent behaviour of the system. Such consistency can be realized in different ways. One way to be consistent is to go for the initiative as much as possible. But it is also consistent to leave the initiative with the user as much as possible.

This is not the place to discuss the proper choice between the two (or may be even more) behaviours. The point is, that initiative is a factor in conducting a dialogue. Initiative is related to the conveying of information, but initiative is not part of the 'information state'.

Another aspect of consistent behaviour, which is has to do with conveying information, but which is a matter of protocol rather than being a part of an information state, is the ordering of subject matters (foci) in a conversation. We mentioned this point already in our discussion on the tagging of dialogues in section 3.

An example of a situation where this ordering seems to be relevant is, when in a dialogue the client has established a date and a performance (s)he wants to go to, and the initiative is left for the system to guide the client through the actual booking process. Although part of the booking will certainly be to inform the client about the price (s)he has to pay for the tickets, and although this price may very well depend on whether the client is entitled to some sort of reduction, it seems extremely awkward to open the booking process with a question about the user's claims for such a reduction. The number of persons that want a ticket, and preference for a rank or a particular position of the seats in the theatre seem far more natural first topics.

Apparently a smooth dialogue needs a smooth ordering of issues to be discussed.

The system must be constantly aware of the most natural (or most probable) next subject to come into the dialogue. This will contribute to an easier understanding of utterances by the client, as well as to a better choice of a question or a remark to be generated, if the initiative is at the system's side.

The initiative in a dialogue and the subject-matter of utterances that we discuss here, were also a theme in the section on dialogue tagging. The point of what was said in this section so far has been that being aware of various kinds of states and being consistent in state transitions is important for smooth dialogues.

In the concluding remarks we will now present a link between the tagging, dialogue states and transitions between them. This link should at the same time explain the title of this section.

It is our idea to equip the final system with one or more transition matrices based on n-gram counts of tagging as found in our annotated corpus. The 'protocol-state' one could say is the history of the dialogue restricted to its last n events. The transition matrix gives the likelihood for every next event in every state.

The data needed to fill the matrix are derived from the numbers as found by counting occurrences of states in the corpus. Note however that we are not obliged to follow these numbers exactly. Consistent behaviour could be improved by tuning some values by hand.

In any case, the transition matrix or matrices are simply Markov models for aspects of the protocol in SCHISMA dialogues!

It must be re-emphasized however, that state as we discuss it here is of restricted importance. There is information state as well, and the final state transition of the dialogue system will be chosen primarily on the basis of that state. The protocol state helps, it should make the flow of information and things like the changes of roles between dialogue partners, the awareness of mental state (like disappointment or discontentness) go smoother.

6. USING NATURAL LANGUAGE.

In SCHISMA, we put quite some effort in the development of a natural language interface using a commercial tool set for building such interfaces: the Intelligent CONnector (ICON, also known as Natural Language) from Natural Language Inc.

The reasons to build such an interface were:

- to evaluate this tool set which according to Sijtsma and Zweekhorst [18], is one of the most sophisticated;
- to investigate to which extent a dialogue can be handled by question answering systems like this one;
- to have a question answering system that could function in a comparison of results of the SCHISMA project with a similar functionality.

Although the final version and the evaluation report will be ready by the end of 1994 (Komen [12]), some preliminary observations can be reported already.

Firstly, our expectations about the ease of use of the tool set and the quality of the resulting system diminished during the development. While 70% of interface was built in two or three weeks, the next ten to fifteen percent was a lot harder to accomplish. At 85% percent the process of steady progress changed into the contrary: with each new addition, the system could collapse to a state in which it misinterpreted questions that it answered correctly earlier.

Secondly, the resulting system is clearly not of the kind that could function as a co-operative partner in a quest for information. The interpretation which is the basis for the transformation of a question to SQL, treats questions (almost) completely

isolated from its history. (The only exceptions are anaphoric expression such as *it* or *he*.) The system cannot handle open, explorative questions or statements like *I'd like to go to an opera tonight* and it is not possible to stick to a particular subject for a while. Furthermore it is impossible to model the effects of reservations such as *Please, reserve two seats at the performance tonight.*

As a last remark we would like to mention the (sometimes extreme) bias towards financial and administrative domains.

Although we still think it valuable to have comparable systems at hand, we can already conclude that any dialogue system will outperform systems built with ICON, simply because they cannot handle dialogues at all.

7. TOWARDS SCHISMA 2001.

As explained in the previous sections, different research lines are pursued in the SCHISMA project. As announced in the introduction, we shall use this final section to provide a broader perspective on things to come.

It is the intention of the Parlevink research group in Twente, that also in the future SCHISMA will allow and stimulate different and changing goals. SCHISMA should be a vehicle in which research results of researchers and Ph.D. students will be embedded.

This does not mean that there will be no end products or tools that can be offered to interested partners. Such tools are available already at this moment (maybe not as a fully developed product, but certainly usable).

To mention a few, there are:

- a system to get information about theatre performances which is developed using Natural Language;
- a Wizard of Oz tool, which may find application in quite different domains;
- an environment which allows investigation of feature structures in a user-friendly way;
- a PROLOG implementation of the information state model as described in section 4.

However, in addition to current activities, many research directions, with further reaching technical aims, will be followed. Below some of them are listed.

Speech. Obviously, the most natural extension to SCHISMA is making it accessible by speech, preferably by telephone. It does not seem difficult to extend our Wizard of Oz tool in such a way that a spoken language corpus for the theatre domain can be obtained. More knowledge about spoken language access to a kind of SCHISMA system will be obtained from interviews that have been and will be conducted at some theatres in some Dutch cities. We have not planned yet to record real-world dialogues by telephone of theatre information or booking services. It is hoped that this can be done in a similar way as is being planned in the framework of the Dutch "Prioriteitsprogramma Taal en Spraak" for train travel information. Presently we plan to investigate speech especially with the aim of integrating speech and language. But there are options for more extensive neural network research into speech. They are discussed below.

Integrating speech and language. Traditional speech and language research have led to different research communities. Speech research is very much concerned with signal processing, noise filtering, phoneme and word recognition. The use of higher-level resources (syntax and semantics) is seldomly employed. The use of statistical approaches is advocated. In SCHISMA research corpus-based and statistical methods will play an important role. Apart from that, we expect it to be possible to integrate our methods of syntactic analysis and unification with HMM speech recognition. The same holds true for knowledge of topic-focus articulation, prosodic information and integration of these aspects in a parser.

Robustness. It is well known that in the near future no comprehensive formal and effective models for all kinds of knowledge sources that play a role in language interpretation will be available. Clearly, in this case analysis with respect to a certain knowledge source can not always lead to a desired result. More in general, if the input is not according to the model of the knowledge source it cannot be given a correct interpretation with respect to this knowledge. Since a complete analysis with respect to one particular kind of linguistic or extra-linguistic knowledge will never be goal in practical applications it is useful to see whether incomplete analysis with respect to a certain type of knowledge can be compensated by using a different type of knowledge. This compensation can be considered as adding robustness to a certain level (or viewpoint) of analysis (and therefore to the whole trajectory of analysis). For example, the use of unknown words may be compensated by syntactic knowledge, but also by knowledge about punctuation or the way humans make errors in typing or speaking. Similarly, not allowable syntactic constructs can be repaired using certain semantic knowledge. Research

on robustness, also including probabilistic methods will remain part of the project and the results will be incorporated into (the) system(s) that will be built.

Evaluation issues. Measuring the quality with which software performs a certain task is an issue that has also emerged in the field of natural language processing. Clearly, due to incomplete models of nearly all aspects of natural language evaluation is a difficult question. Evaluation in the SCHISMA context, for example, can mean evaluation of the system as a whole, evaluation of the different parts or evaluation with respect to different groups of users. Evaluation standards are hardly available but we expect that in the near future the attention that is going to this subject (e.g., the ARPA speech contests, the MUC contest, the morpholympics, the parseolympics) will pay off. Although there is no real competition we hope that in the future at least comparisons can be made between parser systems for Dutch (head and left corner (unification) parsers, data-oriented parsers, AGFL parsers, etc.) and natural language interfaces for Dutch (the primitive SCHISMA obtained from the commercial 'Natural Language' tool, SCHISMA under development and the OVIS system that will be built in the framework of the Dutch "Prioriteitsprogramma Taal en Spraak"). For literature on these issues see Thompson [20].

Neural and Fuzzy Approaches. Probably the most straightforward application of neural network techniques in SCHISMA is in speech recognition. Although some modest excursions into this area have been made (e.g., M.Sc. research by students) there is insufficient expertise and manpower to enter this field. A joint research proposal with, among others, the Max Planck Institute of Nijmegen, will hopefully give us the opportunity to make a start in this field and to use results in SCHISMA. For a more comprehensive approach to neural network techniques for speech and language that deserves consideration for use in the SCHISMA context see Drossaers ([9],and also this volume). Presently it is far from clear whether complete integrated linguistic analysis with neural networks for the SCHISMA domain can ever be realized. We nevertheless assume that the sequence recognition capabilities of the network that has been developed can support decision making in several sub tasks (robustness, disambiguation, dialogue modelling, etc.) for SCHISMA. In addition to the neural network research some modest efforts are being made to develop language theory using a fuzzy logic approach.

At this moment there are no plans to make SCHISMA multi-modal. The emphasis is on keyboard input and if possible we would like to add the possibility to access SCHISMA using speech input. In order to continue SCHISMA research in a satisfactory way it will be necessary to strengthen the cooperation with PTT Research and to embed SCHISMA research in several Dutch and European research programs.

LITERATURE.

[1] Ahrenberg, L., A. Jönsson, and N. Dahlbäck. 1990. Discourse representation and discourse management. In: Dahl and Fraurud (eds.) *Papers from the 2nd Nordic Conference on Text Comprehension in Man and Machine*, Institute of Linguistics, Stockholm.

[2] Allen, J.F. and C.R. Perrault. 1980. Analyzing Intention in Utterances. *Artificial Intelligence 15.*

[3] Andernach, T., G. Deville, and L. Mortier. 1993. The design of a real world wizard of oz experiment for a speech driven telephone directory information system. In: *Proceedings of Eurospeech.*

[4] Andry, F., E. Bilange, F. Charpentier, K. Choukri, M. Ponamale, and S. Soudoplatoff. 1990. Computerised simulation tools for the design of an oral dialogue system.

[5] Bilange, E. 1991. A Task Independent Oral Dialogue Model. *Proceedings of the 5th Conference of the European Chapter of the Association for Computational Linguistics.*

[6] Dahlbäck, N. and A. Jönsson. 1986. A system for studying human-computer dialogues in natural language. Research report, NLPLAB IDA Linköping University, Linköping, Sweden.

[7] Dahlbäck, N. and A. Jönsson. 1989. Empirical studies of discourse representation for natural language Interfaces. In: *Proceedings of the 4th Conference of the EACL.*

[8] Daneš, F. 1974. Functional sentence perspective and the organization of text. In: F. Daneš (ed.), *Papers on Functional Sentence Perspective.* Prague: Academia.

[9] Drossaers, M.F.J. 1995. *Neural Networks for Integrated Linguistic Analysis.* Ph.D. Thesis, Parlevink Research Group, University of Twente, (to appear).

[10] Groenendijk J., M. Stokhof, and F. Veltman. 1994. Dynamic Semantics. *Course Material Sixth European Summer school on Logic, Language and Information.*

[11] Guyomard, M. and J. Siroux. 1988. Experimentation in the specification of an oral dialogue. In: H. Niemann, M. Lang, and G. Sagerer (eds.), *Recent Advances in Speech Understanding and Dialog Systems.* Springer Verlag, Berlin.

[12] Komen, E. 1994. Building SCHISMA in Natural Language. *Memoranda Informatica.* University of Twente, Enschede. (to appear)

[13] Kruijff G-J.M. and J. Schaake. 1994. Discerning Relevant Information in Discourses Using TFA. Presentation at the Fifth Symposium on Computational Linguistics In the Netherlands CLIN94.

[14] Lochbaum, K.E. 1991. An Algorithm for Plan Recognition in Collaborative Discourse. *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics.*

[15] Peirce, C.S. 1982. *Writings of Charles S. Peirce: a Chronological Edition.* Edited by M. Fisch et al. Vol. 1: 1857-1866. Indiana University Press, Bloomington.

[16] Schaake, J. and G-J.M. Kruijff. 1994. Information States Based Analysis of Dialogues. Presentation at the Fifth Symposium on Computational Linguistics In the Netherlands CLIN94. 1994

[17] Searle, J.R. 1969. *Speech Acts, An Essay in the Philosophy of Language.* Cambridge University Press, Cambridge.

[18] Sijtsma, W. and O. Zweekhorst. 1993. Comparison and Review of Commercial Natural Language Interfaces. In: F.M.G. de Jong and A. Nijholt (eds.), *Natural Language Interfaces: From Laboratory to Commercial User Environment. Proceedings of TWLT5.* University of Twente, Enschede.

[19] Stubbs, M. 1983. *Discourse Analysis.* Blackwell, Oxford, 1983.

[20] Thompson, H. (ed.). The Strategic Role of Evaluation in NLP. Proceedings, Edinburgh, Scotland, April-May 1992.

[21] Wachtel, T. 1986. Pragmatic Sensitivity in NL Interfaces and the Structure of Conversation. *Proceedings of the 11th International Conference on Computational Linguistics.*

On the Intersection of Finite State Automata and Definite Clause Grammars

Gertjan van Noord
Vakgroep Alfa-informatica
Rijksuniversiteit Groningen

Abstract

An elegant framework for robust parsing was presented by Bernard Lang on last year's TWLT. In Lang's approach parsing is viewed as the calculation of the intersection of a FSA (the input) and a CFG. Viewing the input for parsing as a FSA rather than as a string combines well with standard approaches in speech understanding systems in which typically a word lattice is produced as an intermediate result. Furthermore, certain techniques for robust parsing can be modelled as finite state transducers.

In this paper we investigate how we can generalize this approach for unification grammars. In particular we will concentrate how we might calculate the intersection of a FSA and a DCG. It is shown that existing parsing algorithms can be easily extended for FSA inputs. However, we also show that the termination properties change drastically: we show that it is undecidable whether the intersection of a FSA and an off-line parsable DCG is empty.

Furthermore we discuss techniques to cope with the problem.

1 Introduction

In this paper we are concerned with the syntactic analysis phase of a speech analysis system. The input for the syntactic analysis phase in such a system is a *word lattice*, and the output is a packed representation of all possible parse trees. Syntactic analysis is performed on the basis of some grammar G. Following Bernard Lang we can thus characterize the syntactic analysis phase as the computation of the intersection of a finite state automaton (the word-lattice) and the grammar G. This intersection is a grammar that derives exactly all parse trees for the input.

Note that we allow the input to be a full FSA (possibly including cycles, etc.) in order for certain techniques to handle ill-formed input to be applied. Whereas an ordinary word-graph always defines a finite language, a FSA of course can easily define an infinite number of sentences.

It can be shown that the computation of the intersection of a FSA and a CFG requires only a minimal generalization of existing parsing algorithms. We simply replace the usual string positions with the names of the states in the FSA. It is therefore straightforward to show that the complexity of this process is cubic in the number of states of the FSA (in the case of ordinary parsing the number of states equals $n + 1$) (Lang, 1974; Billot and Lang, 1989) (assuming the grammar is in bilinear format).

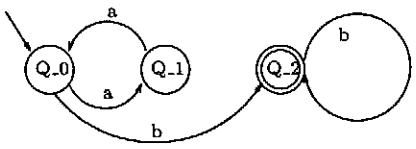
In this paper we investigate whether the same techniques can be applied in case the grammar G is a constraint-based grammar rather than a CFG. For specificity we will take G to be a *Definite Clause Grammar* (DCG) Pereira and Warren (1980). A DCG is a simple example of a family of grammar formalisms that are widely used in natural language analysis (and generation). The main findings of this paper can be extended to the other members of that family of constraint-based grammar formalisms.

2 The intersection of a CFG and a FSA

The calculation of the intersection of a CFG and a FSA is very simple (Bar-Hillel, Perles, and Shamir, 1961). However this construction yields an enor-

mous amount of rules that are 'useless'. Furthermore the resulting parse forest grammar might define an empty language (if the intersection was empty). Luckily 'ordinary' recognizers/parsers for CFG can be easily generalized to construct this intersection (usually yielding a much smaller grammar than in the construction of Bar-Hillel, Perles, and Shamir (1961)). Checking whether the intersection is empty or not is then usually very simple as well: only in the latter case will the parser terminate succesfully.

Consider the following definite clause specification of a FSA. The following conventions apply. We define the transition relation using the relation `trans/3`. For start states, the relation `start/1` should hold, and for final states the relation `final/1` should hold. Thus the following FSA, defining the regular language $L = (aa)^*b^+$ (i.e. an even number of a's followed by at least one b) is given as:




```
start(q0).  final(q2).

trans(q0,a,q1).  trans(q1,a,q0).
trans(q0,b,q2).  trans(q2,b,q2).
```

A context-free grammar is represented as a definite-clause specification as follows. We do not wish to define the sets of terminal and non-terminal symbols explicitly, these can be understood from the rules that are defined using the relation `rule/2`, and where symbols of the rhs are prefixed with '-' in the case of terminals and '+' in the case of non-terminals. The relation `top/1` should hold for the start symbol. The language $L' = a^n b^n$ is defined as:

```
top(s).

rule(s,[-a,+s,-b]).  rule(s,[]).
```

In order to illustrate how ordinary parsers can be used to compute the intersection of a FSA and a CFG consider the definite-clause specification of a top-down parser. This parser runs in cubic time if implemented using Earley deduction or XOLDT resolution. It is assumed that the input string is represented by the `trans/3` predicate.

```
parse(P0,P) :-
    top(Cat), parse(+Cat,P0,P).

parse(-Cat,P0,P) :-
    trans(P0,Cat,P),
    side_effect(p(Cat,P0,P) --> Cat).
parse(+Cat,P0,P) :-
    rule(Cat,Ds),
    parse_ds(Ds,P0,P,His),
    side_effect(p(Cat,P0,P) --> His).

parse_ds([],P,P,[]).
parse_ds([H|T],P0,P,[p(H,P0,P1)|His]) :-
    parse(H,P0,P1),
    parse_ds(T,P1,P,His).
```

The predicate `side_effect` is used to construct the parse forest grammar. The predicate always succeeds, and as a side-effect asserts that its argument is a rule of the parse forest grammar. For the sentence 'a a b b' we obtain the parse forest grammar:

```
p(s,2,2) --> [].
p(s,1,3) -->
    [p(-a,1,2),p(+s,2,2),p(-b,2,3)].
p(s,0,4) -->
    [p(-a,0,1),p(+s,1,3),p(-b,3,4)].
p(a,1,2) --> a.
p(a,0,1) --> a.
p(b,2,3) --> b.
p(b,3,4) --> b.
```

The reader easily verifies that indeed this grammar generates (a isomorphism of) the single parse tree of this example, assuming of course that the start symbol is `p(s,0,4)`. In the parse-forest grammar, complex symbols are non-terminals, atomic symbols are terminals.

Interestingly, nothing needs to be changed to use the same parser for the computation of the intersection of a FSA and a CFG. If our input 'sentence' now is the definition of `trans/3` as given above, we obtain the following parse forest grammar (where the start symbol is `p(s,q0,q2)`):

```
p(s,q0,q0) --> [].
```

```
p(s,q1,q1) --> [].
p(s,q1,q2) -->
    [p(-a,q1,q0),p(+s,q0,q0),p(-b,q0,q2)].
p(s,q0,q2) -->
    [p(-a,q0,q1),p(+s,q1,q2),p(-b,q2,q2)].
p(s,q1,q2) -->
    [p(-a,q1,q0),p(+s,q0,q2),p(-b,q2,q2)].
p(a,q0,q1) --> a.
p(a,q1,q0) --> a.
p(b,q0,q2) --> b.
p(b,q2,q2) --> b.
```

Thus, even though we now use the same parser for an infinite set of input sentences (represented by the FSA) the parser still is able to come up with a parse forest grammar. A possible derivation for this grammar is the following (abbreviated) parse tree in figure 1.

3 The intersection of a DCG and a FSA

First note that the problem of calculating the intersection of a DCG and a FSA can be solved trivially by a generalization of the construction used by Bar-Hillel, Perles, and Shamir (1961). However, if we use that method we will end up with an enormously large forest grammar that is not even guaranteed to contain solutions. Therefore, we are interested in methods that only generate a small subset of this; if the intersection is empty we want an empty parse-forest grammar. The straightforward approach is to generalize existing recognition algorithms.

The same techniques that are used for calculating the intersection of a FSA and a CFG can be applied in the case of DCGs. In order to compute the intersection of a DCG and a FSA we assume that FSA's are represented as before. DCGs are represented using the same notation we used for context-free grammars, but now of course the category symbols can be first-order terms of arbitrary complexity (note that without loss of generality we don't take into account DCGs having external actions defined in curly braces).

3.1 Undecidability

This confronts us with an undecidability problem: the recognition problem for DCGs is undecidable. A fortiori the problem of deciding whether the intersection of a FSA and a DCG is empty or not is undecidable.

In the case of parsing, this undecidability result is usually circumvented by considering subsets of DCGs which can be recognized effectively. For example, we can restrict the attention to DCGs of which the context-free skeleton does not contain cycles. Recognition for such 'off-line parsable' grammars is decidable.

Most existing constraint-based parsing algorithms will terminate for grammars that exhibit the property that for each string there is only a finite number of possible derivations. Note that off-line parsability is one possible way of ensuring that this is the case.

This observation is not very helpful in establishing insights concerning interesting subclasses of DCGs for which termination can be guaranteed (in the case of FSA input). The reason is that there are now two sources of recursion: in the DCG and in the FSA (cycles). As we saw earlier: even for CFG it holds that there can be an infinite number of analyses for a given FSA (but in the CFG this of course does not imply undecidability). In the appendix we give a simple proof that the question whether the intersection of a FSA and an off-line parsable DCG is empty or not is undecidable.

3.2 What to do?

The following approaches towards the undecidability problem can be taken:

- limit the power of the FSA
- limit the power of the DCG
- compromise completeness
- compromise soundness

These approaches are discussed now in turn.

Limit the FSA Rather than assuming the input for parsing is a FSA in its full generality, we might assume that the input is an ordinary word graph (a FSA without cycles).

Thus the techniques for robust processing that give rise to such cycles cannot be used. One example is the processing of an unknown sequence of

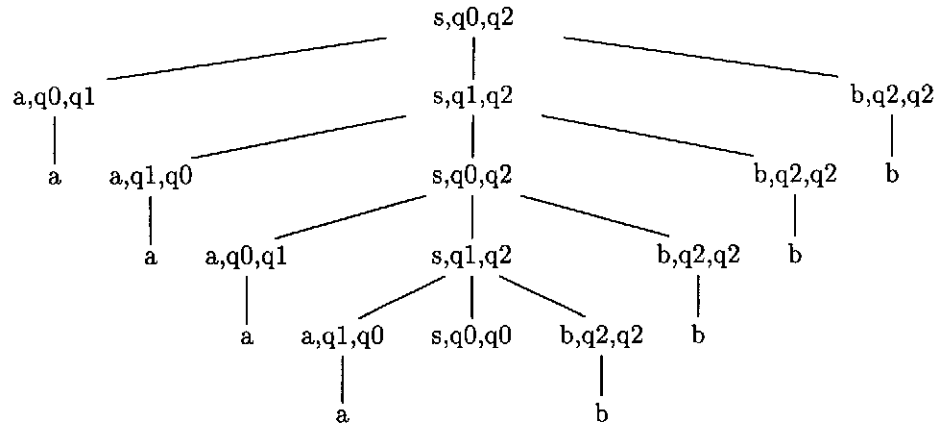


Figure 1: A parse-tree extracted from the parse forest grammar

words, e.g. in case there is noise in the input and it is not clear how many words have been uttered during this noise.

It is not clear to me right now what we loose (in practical terms) if we give up such cycles.

Note that it is easy to verify that parsing is decidable on the basis of a FSA without cycles and a DCG that is off-line parsable.

Limit the DCG Another approach is to limit the size of the categories that are being employed. This is the GPSG and F-TAG approach. In that case we are not longer dealing with DCGs but rather with CFGs (which have been shown to be insufficient in general for the description of natural languages).

Compromise completeness Completeness in this context means: the parse forest grammar contains all possible parses. It is possible to compromise here, in such a way that the parser is guaranteed to terminate.

For example, if we assume that each edge in the FSA is associated with a probability it is possible to define a threshold such that each partial result that is derived has a probability higher than the threshold. Thus, it is still possible to have cycles in the FSA, but anytime the cycle is 'used' the probability decreases and if too many cycles are encountered the threshold will cut of that derivation.

For any threshold it is the case that the intersection problem of off-line parsable DCGs and FSA is decidable.

Compromise soundness Soundness in this context should be understood as the property that all parse trees in the parse forest grammar are valid parse trees. A possible way to ensure termination is to remove all constraints from the DCG and parse according to this context-free skeleton. The resulting parse-forest grammar will be too general most of the times.

A practical variation can be conceived as follows. From the DCG we take its context-free skeleton. This skeleton is obtained by removing the constraints from each of the grammar rules. Then we compute the intersection of the skeleton with the input FSA. This results in a parse forest grammar. Finally, we add the corresponding constraints from the DCG to the grammar rules of the parse forest grammar.

This has the advantage that the result is still sound, although the size of the parse forest grammar is not optimal (as a consequence it is not guaranteed that the parse forest grammar contains a parse tree).

References

Bar-Hillel, Y., M. Perles, and E. Shamir. 1961. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprach-Wissenschaft und Kommunicationsforschung*, 14:143–172. Reprinted in Bar-Hillel's Language and Information – Selected Essays on their Theory and Application, Addison Wesley series in Logic, 1964, pp. 116-150.

Billot, S. and B. Lang. 1989. The structure of shared parse forests in ambiguous parsing. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 143–151, Vancouver.

Grosz, Barbara, Karen Sparck Jones, and Bonny Lynn Webber, editors. 1986. *Readings in Natural Language Processing.* Morgan Kaufmann.

Hopcroft, John E. and Jeffrey D. Ullman. 1979. *Introduction to Automata Theory, Languages and Computation.* Addison Wesley.

Lang, Bernard. 1974. Deterministic techniques for efficient non-deterministic parsers. In J. Loeckx, editor, *Proceedings of the Second Colloquium on Automata, Languages and Programming.* Also: Rapport de Recherche 72, IRIA-Laboria, Rocquencourt (France).

Pereira, Fernando C.N. and David Warren. 1980. Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13. reprinted in (Grosz, Jones, and Webber, 1986).

A Intersection of FSA and off-line parsable DCG is undecidable

Here I show that the question whether the intersection of a FSA and an off-line parsable DCG is empty is undecidable. A yes-no problem is *undecidable* (cf. Hopcroft and Ullman (1979, pp.178-179)) if there is no algorithm that takes as its input an *instance* of the problem and determines whether the answer to that instance is 'yes' or 'no'. An instance of a problem consists of a particular choice of the *parameters* of that problem.

I use Post's Correspondence Problem (PCP) as a well-known undecidable problem. I show that if the above mentioned intersection problem were decidable, then we could solve the PCP too. The following definition and example of a PCP are taken from Hopcroft and Ullman (1979)[chapter 8.5].

An instance of PCP consists of two lists, $A = v_1 \dots v_k$ and $B = w_1 \dots w_k$ of strings over some alphabet Σ. This instance has *a solution* if there is any sequence of integers $i_1 \dots i_m$, with $m \geq 1$, such that

$$v_{i_1}, v_{i_2}, \dots, v_{i_m} = w_{i_1}, w_{i_2}, \dots, w_{i_m}.$$

The sequence $i_1, \dots, i_m$ is a solution to this instance of PCP. As an example, assume that $\Sigma = \{0, 1\}$. Furthermore, let $A = \langle 1, 10111, 10 \rangle$ and $B = \langle 111, 10, 0 \rangle$. A solution to this instance of PCP is the sequence 2,1,1,3 (obtaining the sequence 101111110). For an illustration, cf. figure 2.

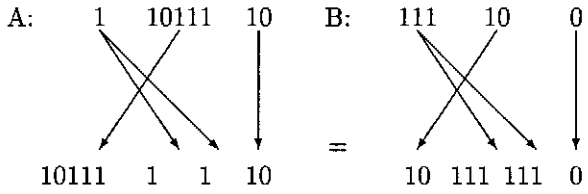


Figure 2: Illustration of a solution of a PCP problem.

Clearly there are PCP's that do not have a solution. Assume again that $\Sigma = \{0, 1\}$. Furthermore let $A = \langle 1 \rangle$ and $B = \langle 0 \rangle$. Clearly this PCP does not have a solution. In general, however, the problem whether some PCP has a solution or not is not decidable. This result is proved by Hopcroft and Ullman (1979) by showing that the halting problem for Turing Machines can be encoded as an instance of Post's Correspondence Problem.

First I give a simple algorithm to encode any instance of a PCP as a pair, consisting of a FSA and an off-line parsable DCG, in such a way that the question whether there is a solution to this PCP is equivalent to the question whether the intersection of this FSA and DCG is empty.

Encoding of PCP.

1. For each $1 \leq i \leq k$ (k the length of lists A and B) define a DCG rule (the $i-th$ member of A is $a_1 \ldots a_m$, and the i−th member of B is $b_1 \ldots b_n$): $r([a_1 \ldots a_m|A], A, [b_1 \ldots b_n|B], B) \rightarrow [x]$.

2. Furthermore, there is a rule $r(A_0, A, B_0, B) \rightarrow r(A_0, A_1, B_0, B_1), r(A_1, A, B_1, B)$.

3. Furthermore, there is a rule $s \rightarrow r(X, [\,], X, [\,])$. Also, s is the start category of the DCG.

4. Finally, the FSA consists of a single state q which is both the start state and the final state, and a single transition $\delta(q, x) = q$.

Observe that the DCG is off-line parsable.

The underlying idea of the algorithm is really very simple. For each pair of strings from the lists A and B there will be one lexical entry where these strings are represented by a difference-list encoding. Furthermore there is a general combination rule that simply concatenates A-strings and concatenates B-strings. Finally the rule for s states that in order to construct a succesful top category the A and B lists must match.

The resulting DCG, FSA pair for the example PCP is:

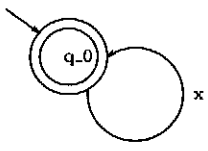
```
s --> r(X,[],X,[]).

r(A0,A,B0,B) -->
    r(A0,A1,B0,B1),
    r(A1,A, B1,B ).

r([1|A],A,[1,1,1|B],B) --> [x].

r([1,0,1,1,1|A],A,[1,0|B],B) --> [x].

r([1,0|A],A,[0|B],B) --> [x].
```



Proposition The question whether the intersection of a FSA and an off-line parsable DCG is empty is undecidable.

Proof. Suppose the problem *was* decidable. In that case there would exist an algorithm for solving the problem. This algorithm could then be used to solve the PCP, because a PCP π has a solution if and only if its encoding given above as a FSA and an off-line parsable DCG is not empty. The PCP problem however is known to be undecidable. Hence the intersection question is undecidable too.

Prediction and Disambiguation by means of Data-Oriented Parsing

Rens Bod* and Remko Scha
Institute for Logic, Language and Computation
Department of Computational Linguistics
University of Amsterdam
Spuistraat 134
1012 VB Amsterdam
The Netherlands
{rens|scha}@alf.let.uva.nl

1 Introduction

The design of today's experimental spoken language understanding systems usually encompasses two distinct statistical language models. First of all, there is a model which is used to choose the most probable string among the strings suggested by the speech recognition component; this is often an n-gram Markov model (cf. Waibel and Lee, 1990). Secondly, there is a model which is employed for resolving the ambiguities that arise in analyzing that string; this is often a stochastic grammar (cf. (D)ARPA Proceedings, 1991-1993).

It is obvious that a hybrid system does not represent and apply its knowledge of the input language in an optimal way. If speech recognition and natural language processing are to be effectively integrated, it is preferable to have one statistical model that is invoked in guessing the input string as well as in guessing its analysis. This model should integrate probabilistic knowledge about lexical items with probabilistic knowledge about syntactic structure.

This note describes how a spoken language understanding system based on the Data-Oriented Parsing approach (Scha, 1990; Bod, 1992) uses a sample corpus of syntactic subtrees to achieve both functionalities at once. We assume a speech recognition component which provides, for every string under consideration, the probability for this string to be realized as the acoustic signal that was actually observed. The language component of the system then uses Bayesian statistics to compute the most probable one among all candidate parses of all candidate strings.

In the following, we start with a short introduction to the Data-Oriented Parsing approach, after which we deal with the prerequisites of a statistical language model that integrates sentence prediction and sentence disambiguation.

*Also: Research Institute for Language and Speech, Utrecht University, Trans 10, 3512 JK Utrecht.

2 The Data-Oriented Parsing Approach

A Data-Oriented Parsing (DOP) model is characterized by a corpus of analyzed language utterances, together with a set of operations that combine subanalyses from the corpus into new analyses. We will limit ourselves in this paper to corpora with purely syntactic annotations; for the semantic dimension of DOP the reader is referred to (van den Berg et al., 1994). Consider the following imaginary example of a corpus consisting of only two trees:

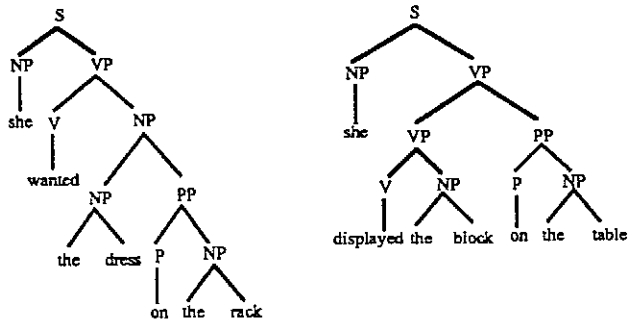


Figure 1. Imaginary corpus of two trees.

We will assume one operation for combining subtrees. This operation is called "composition", and is indicated by the infix operator o. The composition of t and u, t o u, yields a copy of t in which its leftmost nonterminal leaf node has been identified with the root node of u (i.e., u is *substituted* on the leftmost nonterminal leaf node of t). For reasons of simplicity we will write in the following $(t \text{ o } u) \text{ o } v$ as: $t \text{ o } u \text{ o } v$.

Now the (ambiguous) sentence *"She displayed the dress on the table"* can be parsed by combining subtrees from the corpus. For instance:

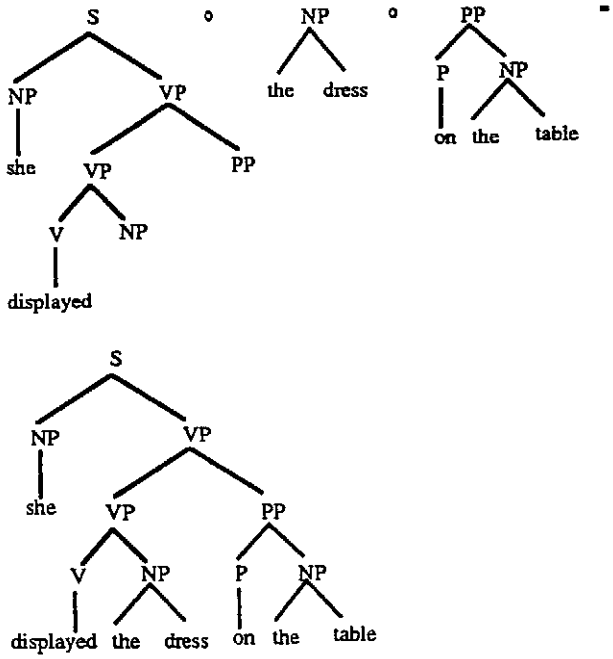


Figure 2. Derivation and parse tree for *"She displayed the dress on the table"*

As the reader may easily ascertain, a different derivation may yield a different parse tree. However, a different derivation may also very well yield the same parse tree; for instance:

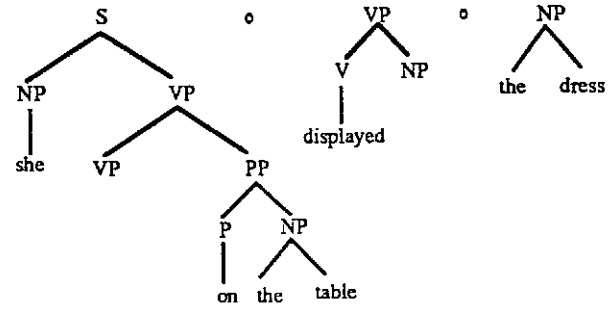


Figure 3. Different derivation generating the same parse tree for *"She displayed the dress on the table"*

or

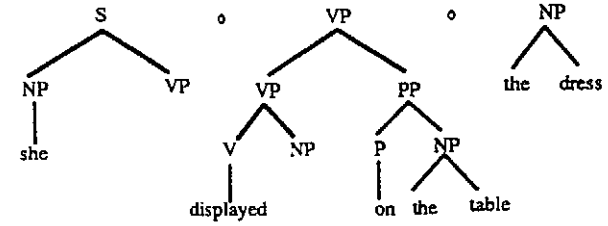


Figure 4. Another derivation generating the same parse tree for *"She displayed the dress on the table"*

Thus, a parse tree can have several derivations involving different subtrees. Using the corpus for our stochastic estimations, we estimate the probability of substituting a certain subtree on a specific node as the probability of selecting this subtree among all subtrees in the corpus that could be substituted on that node.[1] The probability of a derivation can be computed as the product of the probabilities of the substitutions that it involves. The probability of a parse tree is equal to the probability that any of its derivations occurs, which is the sum of the probabilities of all derivations of that parse tree. Finally, the probability of a word string is equal to the sum of the probabilities of all its parse trees.

Notice that DOP combines the best of two worlds: the lexical sensitivity of Markov models and the structural sensitivity of phrase structure grammars. Moreover, by taking into account all corpus-subtrees, no lexical/structural relationship that might possibly be of statistical interest is ignored. For computational and experimental evaluations of DOP we refer to (Bod, 1993a, 1993b and 1993c). Here it may suffice to mention that our experiments on the Air Travel Information System (ATIS) corpus as analyzed in the Pennsylvania Treebank (Marcus et al., 1993) resulted in a 96% parse tree accuracy, which is substantially better than the results achieved by other stochastic parsing models on this corpus (cf. Pereira and Schabes, 1992; Brill, 1993).

3 An Overall Statistical Model for Prediction and Disambiguation

Let us consider the task of finding the analysis tree T that is most probable to underly the word string that gave rise to the acoustic signal A that was observed.[2] We are thus interested in the conditional probability of T given A, i.e.: P(T|A).

Direct application of Bayes' rule gives

$$P(T|A) = P(T)\,P(A|T) / P(A).$$

Now we introduce the assumption that the acoustic realization of a word string is independent of its syntactic analysis. This assumption is probably false, but it makes for a simple interface between the speech recognition component and the natural language processing component of the system. It

[1] Very small frequencies are smoothed by the Good-Turing method.

[2] In an actual application system, we wish to find the most probable interpretation I of an acoustic evidence A. The statistical model developed here can also be used for that situation, if we substitute I for T everywhere. In (van den Berg et al., 1994), we investigated what is involved in estimating the probability of an interpretation I given a word string W.

follows that the acoustic realization of a tree T is completely determined by its yield W(T). Therefore:

$P(T|A) = P(T)\, P(A|W(T)) / P(A)$.

Now we are interested in the tree T that maximizes the value of P(T|A), given a particular acoustic signal A. We write this as $\max_T P(T|A)$.

$$\max_T P(T|A) = \max_T P(T)\, P(A|W(T)) / P(A)$$
$$= \max_T P(T)\, P(A|W(T))$$

P(T) is the *a priori* probability estimated by the DOP model. P(A|W(T)) is the acoustic probability estimated by the speech recognition component.

If we are only interested in that part of the tree which represents the word string, we may replace T by W, yielding the formula for predicting the word string W given an acoustic evidence A.

Conclusion

We presented a formal model that integrates speech and natural language in a Bayesian framework. The lexical and structural information exploited by the Data-Oriented Parsing model, make it possible to accomplish the prediction of a word string and its disambiguation at the same time. The next step is the design of an efficient algorithm that estimates the most probable analysis from a word lattice. We conjecture that DOP algorithms for finding the most probable analysis of a string (Sima'an et al., 1994) can be adapted to the speech analysis problem.

References

M. van den Berg, R. Bod & R. Scha, 1994. "A Corpus-Based Approach to Semantic Interpretation", *Proceedings Ninth Amsterdam Colloquium*, Amsterdam.

R. Bod, 1992. "A Computational Model of Language Performance: Data Oriented Parsing", *Proceedings COLING'92*, Nantes.

R. Bod, 1993a. "Using an Annotated Corpus as a Stochastic Grammar", *Proceedings European Chapter fo the ACL'93*, Utrecht.

R. Bod, 1993b. "Monte Carlo Parsing", *Proceedings Third International Workshop on Parsing Technologies*, Tilburg/Durbuy.

R. Bod, 1993c. "Data Oriented Parsing as a General Framework for Stochastic Language Processing", in: K.Sikkel & A. Nijholt (eds.), *Parsing Natural Language*, TWLT6, Twente University.

E. Brill, 1993. "Transformation-Based Error-Driven Parsing", *Proceedings Third International Workshop on Parsing Technologies*, Tilburg/Durbuy.

(D)ARPA Proceedings on Speech and Natural Language, 1991, 1992, 1993. Morgan Kaufmann, San Mateo (CA).

M. Marcus, B. Santorini and M. Marcinkiewicz, 1993. "Building a Large Annotated Corpus of English: the Penn Treebank", *Computational Linguistics* 19(2).

F. Pereira and Y. Schabes, 1992. "Inside-Outside Reestimation from Partially Bracketed Corpora", *Proceedings ACL'92*, Newark.

R. Scha, 1990. "Language Theory and Language Technology; Competence and Performance" (in Dutch), in Q.A.M. de Kort & G.L.J. Leerdam (eds.), *Computertoepassingen in de Neerlandistiek*, Almere: Landelijke Vereniging van Neerlandici (LVVN-jaarboek).

K. Sima'an, R. Bod, S. Krauwer and R. Scha, 1994. "Efficient Disambiguation by means of Stochastic Tree Substitution Grammars", *Proceedings International Conference on New Methods in Language Processing*, UMIST, Manchester.

A. Waibel and K. Lee, 1990. *Readings in Speech Recognition*, Morgan Kaufmann, San Mateo (CA).

PROJECT PARLEVINK

Language Engineering
University of Twente

PARLEVINK Research Topics

The Parlevink Project started in January 1992. It did not start from scratch. In previous years research took place in the area of theory of formal and programming languages (theoretical computer science, compiler construction) and more and more this research became influenced by potential applications in the area of natural language processing. Currently the following three research directions are distinguished:

- research which concentrates on syntactic formalisms and where syntax is the starting point for studying the description and processing of semantic and pragmatic aspects of language;
- research which concentrates on the representation of meaning in dialogue modelling and where syntax is of secondary importance;
- research which concentrates on modelling language behaviour with the help of neural networks and where language learning and integrated use of syntactic, semantic and pragmatic knowledge are the main characteristics.

In 1993 a start has been made with the integration of the different research tracks in the design of a natural language interface that allows a user to ask information about theatre performances in a city. This so-called SCHISMA subproject has taken the form of joint research with PTT Research.

PARLEVINK Researchers

More than ten researchers, including Ph.D. students, are involved in the project. In 1995 at least four Ph.D. students will be involved in the research. Programming support is provided by the Computing Laboratory of the Department of Computer Science. A large number of computer science students are performing their M. Sci. work in the project. It is not unusual that they spend part of their education in companies in the Netherlands or with research groups in the USA.

PARLEVINK Activities

PARLEVINK participates in the Centre of Telematics and Information Technology (CTIT) of the University of Twente. Research is published in books, journals and proceedings of (international) workshops and conferences (COLING, ICANN, KONVENS, IWPT, etc.). A complete list of publications is available on request. Twice a year a workshop (TWLT: Twente Workshop on Language Technology) is organised. Proceedings of these workshops are available. In 1991 there were workshops on Generalised LR Parsing and on Linguistic Engineering. In 1992: Connectionist Natural Language Processing and Pragmatics in Language Technology. In 1993: Natural Language Interfaces and Parsing Natural Language. In 1994: Computer-Assisted Language Learning and Speech and Language Engineering. In 1995 there will be workshops on Corpus-Based Dialogue Modelling and on Algebraic Methods in Language Processing. Students and project members are informed about research, lectures and other activities during weekly meetings and in the PARLEBODE, a monthly newsletter.

Twente Workshops on Language Technology

The TWLT workshops are organised by the PARLEVINK project of the University of Twente. The first workshop was held in Enschede, the Netherlands on March 22, 1991. It was attended by about 40 participants. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 1 (TWLT 1)
Tomita's Algorithm: Extensions and Applications
Eds. R. Heemels, A. Nijholt & K. Sikkel, 103 pages.

Preface and Contents
A. Nijholt *(University of Twente, Enschede).* (Generalised) LR Parsing: From Knuth to Tomita.
R. Leermakers *(Philips Research Labs, Eindhoven).* Recursive Ascent Parsing.
H. Harkema & M. Tomita *(University of Twente, Enschede & Carnegie Mellon University, Pittsburgh).* A Parsing Algorithm for Non-Deterministic Context-Sensitive Languages.
G.J. van der Steen *(Vleermuis Software Research, Utrecht).* Unrestricted On-Line Parsing and Transduction with Graph Structured Stacks.
J. Rekers & W. Koorn *(CWI, Amsterdam & University of Amsterdam, Amsterdam).* Substring Parsing for Arbitrary Context-Free Grammars.
T. Vosse *(NICI, Nijmegen).* Detection and Correction of Morpho-Syntactic Errors in Shift-Reduce Parsing.
R. Heemels *(Océ Nederland, Venlo).* Tomita's Algorithm in Practical Applications.
M. Lankhorst *(University of Twente, Enschede).* An Empirical Comparison of Generalised LR Tables.
K. Sikkel *(University of Twente, Enschede).* Bottom-Up Parallellization of Tomita's Algorithm.

The second workshop in the series (TWLT 2) has been held on November 20, 1991. The workshop was attended by more than 70 researchers from industry and university. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 2 (TWLT 2)
Linguistic Engineering: Tools and Products.
Eds. H.J. op den Akker, A. Nijholt & W. ter Stal, 115 pages.

Preface and Contents
A. Nijholt *(University of Twente, Enschede).* Linguistic Engineering: A Survey.
B. van Bakel *(University of Nijmegen, Nijmegen).* Semantic Analysis of Chemical Texts.
G.J. van der Steen & A.J. Dijenborgh *(Vleermuis Software Research, Utrecht).* Lingware: The Translation Tools of the Future.
T. Vosse *(NICI, Nijmegen).* Detecting and Correcting Morpho-syntactic Errors in Real Texts.
C. Barkey *(TNO/ITI, Delft).* Indexing Large Quantities of Documents Using Computational Linguistics.
A. van Rijn *(CIAD/Delft University of Technology, Delft).* A Natural Language Interface for a Flexible Assembly Cell.
J. Honig *(Delft University of Technology, Delft).* Using Deltra in Natural Language Front-ends.
J. Odijk *(Philips Research Labs, Eindhoven).* The Automatic Translation System ROSETTA3.
D. van den Akker *(IBM Research, Amsterdam).* Language Technology at IBM Nederland.

M.-J. Nederhof, C.H.A. Koster, C. Dekkers & A. van Zwol *(University of Nijmegen, Nijmegen).* The Grammar Workbench: A First Step Toward Lingware Engineering.

The third workshop in the series (TWLT 3) was held on May 12 and 13, 1992. Contrary to the previous workshops it had an international character with eighty participants from the U.S.A., India, Great Britain, Ireland, Italy, Germany, France, Belgium and the Netherlands. The contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 3 (TWLT 3)
Connectionism and Natural Language Processing
Eds. M.F.J. Drossaers & A. Nijholt, 142 pages.

Preface and Contents
L.P.J. Veelenturf *(University of Twente, Enschede).* Representation of Spoken Words in a Self-Organising Neural Net.
P. Wittenburg & U. H. Frauenfelder*(Max-Planck Institute, Nijmegen).* Modelling the Human Mental Lexicon with Self-Organising Feature Maps.
A.J.M.M. Weijters & J. Thole *(University of Limburg, Maastricht).* Speech Synthesis with Artificial Neural Networks.
W. Daelemans & A. van den Bosch *(Tilburg University, Tilburg).* Generalisation Performance of Back Propagation Learning on a Syllabification Task.
E.-J. van der Linden & W. Kraaij *(Tilburg University, Tilburg).* Representation of Idioms in Connectionist Models.
J.C. Scholtes *(University of Amsterdam, Amsterdam).* Neural Data Oriented Parsing.
E.F. Tjong Kim Sang *(University of Groningen, Groningen).* A connectionist Representation for Phrase Structures.
M.F.J. Drossaers *(University of Twente, Enschede).* Hopfield Models as Neural-Network Acceptors.
P. Wyard *(British Telecom, Ipswich).* A Single Layer Higher Order Neural Net and its Application to Grammar Recognition.
N.E. Sharkey & A.J.C. Sharkey *(University of Exeter, Exeter).* A Modular Design for Connectionist Parsing.
R. Reilly *(University College, Dublin).* An Exploration of Clause Boundary Effects in SRN Representations.
S.M. Lucas *(University of Essex, Colchester).* Syntactic Neural Networks for Natural Language Processing.
R. Miikkulainen *(University of Texas, Austin).* DISCERN: A Distributed Neural Network Model of Script Processing and Memory.

The 4th workshop: "Pragmatics in Language Technology" was held on September 23, 1992. It was visited by more than 50 researchers. Contents of the proceedings are given below.

Proceedings Twente Workshop on Language Technology 4 (TWLT 4)
Pragmatics in Language Technology
Eds. D. Nauta, A. Nijholt & J. Schaake, 114 pages.

Preface and Contents
D. Nauta, A. Nijholt & J. Schaake *(University of Twente, Enschede).* Pragmatics in Language Technology: Introduction.
Part 1: Pragmatics and Semiotics
J. van der Lubbe & D. Nauta *(Delft University of Technology & University of Twente, Enschede).* Semiotics, Pragmatism, and Expert Systems.

TWLT 5 took place on 3 and 4 June 1993. It was devoted to the topic "Natural Language Interfaces". The aim was to provide an international platform for commerce, technology and science to present the advances and current state of the art in this area of research.

Proceedings Twente Workshop on Language Technology 5 (TWLT 5)
Natural Language Interfaces
Eds. F.M.G. de Jong & A. Nijholt, 124 pages.

TWLT 6 took place on 16 and 17 December 1993. It was devoted to the topic "Natural Language Parsing". The aim was to provide an international platform for technology and science to present the advances and current state of the art in this area of research, in particular research that aims at analysing real-world text, real-world speech and keyboard input.

Proceedings Twente Workshop on Language Technology 6 (TWLT 6)
Natural Language Parsing: Methods and Formalisms
Eds. K. Sikkel & A. Nijholt, 190 pages.

The seventh workshop in the series took place on 15 and 16 june 1994. It was devoted to the topic "Computer-Assisted Language Learning" (CALL). The aim was to present both the state of the art in CALL and the new perspectives in the research and development of software that is meant to be used in a language curriculum. By the mix of themes addressed in the papers and demonstrations, it was hoped to bring about the exchange of ideas between people of various backgrounds.

Proceedings Twente Workshop on Language Technology 7 (TWLT 7)
Computer-Assisted Language Learning
Eds. L. Appelo, F.M.G. de Jong, 133 pages.

C. Schwind (*Universite de Marseille, France*). Error analysis and explanation in knowledge based language tutoring.
J. Thompson (*CTI, Hull, United Kingdom/EUROCALL*). TELL into the mainstream curriculum.
M. Zock (*Limsi, Paris, France*). Language in action, or learning a language by whatching how it works.

The eighth workshop in the series took place on 1 and 2 December 1994. It was devoted to speech, the integration of speech and natural language processing and the application of this integration in natural language interfaces. The program emphasized research of interest for the themes in the framework of the Dutch NWO programme on Speech and Natural Language that started in 1994.

Proceedings Twente Workshop on Language Technology 8 (TWLT 8)
Speech and Language Engineering
Eds. L. Boves, A. Nijholt, 167 pages.

Chr. Dugast (*Philips, Aachen, Germany*). The North American Business News Task: Speaker Independent, Unlimited Vocabulary Article Dictation.
P. van Alphen, C. in't Veld & W. Schelvis (*PTT Research, Leidschendam, the Netherlands*). Analysis of the Dutch Polyphone Corpus.
H.J.M. Steeneken & D.A. van Leeuwen (*TNO Human Factors Research, Soesterberg, The Netherlands*). Assessment of Speech Recognition Systems.
J. M. McQueen (*Max Planck Insitute, Nijmegen, The Netherlands*). The Role of Prosody in Human Speech Recognition.
L. ten Bosch (*IPO, Eindhoven, the Netherlands*). The Potential Role of Prosody in Automatic Speech Recognition.
P. Baggia, E. Gerbino, E. Giachin, & C. Rullent (*CSELT, Torino, Italy*). Spontaneous Speech Phenomena in Naive-User Interactions.
M.F.J. Drossaers & D. Dokter (*University of Twente, Enschede, the Netherlands*). Simple Speech Recognition with Little Linguistic Creatures.
H. Helbig & A. Mertens (*FernUniversität Hagen, Germany*). Word Agent Based Natural Language Processing.
Geunbae Lee et al. (*Pohang University, Hyoja-Dong, Pohang, Korea*). Phoneme-Level Speech and Natural Language Integration for Agglutinative Languages.
K. v. Deemter, J. Landsbergen, R. Leermakers & J. Odijk (*IPO, Eindhoven, the Netherlands*). Generation of Spoken Monologues by Means of Templates.
D. Carter & M. Rayner (*SRI International, Cambridge, U.K.*). The Speech-Language Interface in the Spoken Language Translator.
H. Weber (*University of Erlangen, Germany*). Time-synchronous Chart Parsing of Speech Integrating Unification Grammars with Statistics.
G. Veldhuijzen van Zanten & R. op den Akker (*University of Twente, Enschede, the Netherlands*). More Efficient Head and Left Corner Parsing of Unification-based Grammar Formalisms.
G.F. van der Hoeven et al. (*University of Twente, Enschede, the Netherlands*). SCHISMA: A Natural Language Accessible Theatre Information and Booking System.
G. van Noord (*University of Groningen, the Netherlands*). On the Intersection of Finite State Automata and Definite Clause Grammars.
R. Bod and R. Scha (*University of Amsterdam, the Netherlands*). Prediction and Disambiguation by Means of Data-Oriented Parsing.

The proceedings of the workshops can be ordered from Vakgroep SETI, Department of Computer Science, University of Twente, P.O. Box 217, NL-7500 AE Enschede, The Netherlands. E-mail orders are possible: bijron@cs.utwente.nl. Each of the proceedings costs Dfl. 30.