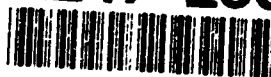


RL-TR-91-362  
Final Technical Report  
December 1991

AD-A247 286



6

# NATURAL LANGUAGE PROCESSING SYSTEMS EVALUATION WORKSHOP

Calspan-UB Research Center

Jeannette G. Neal and Sharon M. Walter



*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.*

92-06165



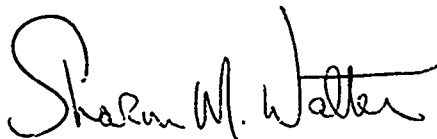
Rome Laboratory  
Air Force Systems Command  
Griffiss Air Force Base, NY 13441-5700

92 3 08 104

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-91-362 has been reviewed and is approved for publication.

APPROVED:



SHARON M. WALTER  
Project Engineer

FOR THE COMMANDER:



JOHN A. GRANIERO  
Chief Scientist  
Command, Control & Communications Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL(C3CA) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE December 1991		3. REPORT TYPE AND DATES COVERED Final May 91 - Nov 91	
4. TITLE AND SUBTITLE NATURAL LANGUAGE PROCESSING SYSTEMS EVALUATION WORKSHOP				5. FUNDING NUMBERS C - F30602-88-D-0026 Task B-1-3371 PE - 62702F PR - 5581 TA - 27 WU - P1	
6. AUTHOR(S) Jeannette G. Neal and Sharon M. Walter				7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Calspan-UB Research Center P O Box 400 Buffalo NY 14225	
8. PERFORMING ORGANIZATION REPORT NUMBER N/A				9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Rome Laboratory (C3CA) Griffiss AFB NY 13441-5700	
10. SPONSORING/MONITORING AGENCY REPORT NUMBER RL-TR-91-362				11. SUPPLEMENTARY NOTES Rome Laboratory Project Engineer: Sharon M. Walter/C3CA/(315) 330-7650	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.				12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) The Natural Language Processing Systems Evaluation Workshop, sponsored by Rome Laboratory and attended by over 60 people, was held in conjunction with the 29th Annual Meeting of the Association for Computational Linguistics (ACL) on 18 June 1991 at the Berkeley Campus of the University of California. The Workshop provided a forum for computational linguists to discuss current evaluation efforts and activities, research progress, and new approaches; promoted scientific interchange on important evaluation issues; and generated recommendations and directions for future investigations in the evaluation area. The papers in this report are formal records of the presentations made at the Workshop.					
14. SUBJECT TERMS Natural Language, Interface Evaluation, NLP, Workshop, Man-Machine Interface				15. NUMBER OF PAGES 184	
				16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED		18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED		19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	
				20. LIMITATION OF ABSTRACT U1	

# Natural Language Processing Systems Evaluation Workshop

## CONTENTS

Introduction.....	iii
Third Message Understanding Evaluation and Conference (MUC-3): Methodology and Test Results..... <i>Beth M. Sundheim</i>	1
MUC-3 Evaluation Metrics and Linguistic Phenomena Tests..... <i>Nancy Chinchor</i>	13
A Developing Methodology for the Evaluation of Spoken Language Systems..... <i>Madelaine Bates and Sean Boisen</i>	19
Multi-site Natural Language Processing Evaluation: MUC and ATIS..... <i>Deborah Dahl, Doug Appelt, and Carl Weir</i>	31
Benchmark Investigation/Identification Project: Phase I..... <i>Jeannette G. Neal, Elissa L. Feit, and Christine A. Montgomery</i>	41
Evaluating Syntax Performance of Parser/Grammars of English..... <i>Philip Harrison, Steven Abney, Ezra Black, Dan Flickinger, Claudia Gdaniec, Ralph Grishman, Donald Hindle, Robert Ingria, Mitch Marcus, Beatrice Santorini, and Tomek Strzalkowski</i>	71
A Diagnostic Tool for German Syntax..... <i>John Nerbonne, Klaus Netter, Abdel Kader Diagne, Ludwig Dickmann, and Judith Klein</i>	79
Preliminaries to the Development of Evaluation Metrics for Natural Language Semantic and Pragmatic Analysis Systems..... <i>James E. Hoard</i>	97
Corpus-Based Evaluation of the Sundial System..... <i>Norman M. Fraser</i>	103
Module-Level Testing: Lessons Learned..... <i>Sharon Flank, Aaron Temin, Hattje Blejer, Andrew Kehler, and Sherman Greenstein</i>	109
Maintaining and Enhancing a NL DBMS Interface..... <i>R. E. Cullingford and M. Graves</i>	121
Evaluation for Generation..... <i>Marie Meteer and David McDonald</i>	127
Evaluating Natural Language Generation Facilities in Intelligent Systems..... <i>Johanna D. Moore</i>	133
Measuring Compositionality in Transfer-Based Machine Translation Systems..... <i>Bjorn Gamback, Hiyun Alshawi, David Carter, and Manny Rayner</i>	141



Good Applications for Crummy Machine Translation.....147  
*Kenneth W. Church and Eduard H. Hovy*

Gross-Grained Software Evaluation: The Natural Language Software Registry.....159  
*Elizabeth A. Hinkelman*

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

## INTRODUCTION

The evaluation of natural language processing (NLP) systems is essential for continued, steady progress in the development and application of NLP technology. Its value encompasses the lifecycle of individual projects (steering the course of individual system developments and providing specifications for placing systems into service) and the spectrum from individual projects to the whole of the technology. For particular system development projects, the identification of strengths and weaknesses is necessary in order to chart progress, guide the course of evolution, and provide feedback into research and development cycles. With regard to applications, researchers must be able to measure the effectiveness of NLP systems and system components in order to appropriately combine them with other interface technologies and match them to the characteristics of specific tasks and user requirements in applications. Beyond the value to particular projects, a common and consistent basis for measurement, description, and comparison encourages the technical exchange and commingling of theories and ideas that will be required for the science of NLP technology to advance.

Interest in developing standard NLP evaluation methodologies is growing as the technology matures. Although there is keen interest in the problem of evaluation, there are no clear and obvious answers to questions regarding the basis of evaluation (e.g., task, corpus, capabilities), evaluation methods (black box versus glass box), metrics (e.g., recall, precision, averages, percentages, statistical measures), or the format and content of reported results (e.g., numerical scores, descriptions, profiles).

Corpus-based, task-based, and capability-based evaluation are three types of NLP evaluation. For corpus-based evaluation, a fixed body (or "corpus") of inputs is given to the system and measurements are made based on the system outputs. A number of standard corpuses for NL understanding system evaluation are in use today ([BBN, 1988]; [Flickinger et al., 1987]; [Hendrix, Sacerdoti, and Slocum 1976]; [Malhotra, 1975]). In a task-based evaluation, a specific task or tasks are to be completed using the system. The basis for judgement is how well the task was accomplished. Capability-based (or checklist-based) evaluation uses a list of individual capabilities to guide the evaluation. During evaluation each capability is assigned an indicator of whether or not (or, to what degree) the system demonstrates that capability.

NLP System evaluations can occur at different levels of "detail". Black Box evaluation focuses on what a system does, measuring performance based on well-defined input/output

pairs without concern for how the system processes the input and generates its output. In contrast, a glass box evaluation "looks into" the system and examines the particulars of how it works. In a sense, the distinction between black and glass box evaluations can be a matter of perspective: a black box evaluation of one or more components of a particular system could be considered a glass box evaluation from the system level perspective.

Further complications arise when one considers that NLP systems evaluation can be performed from the perspective of the NLP system developer, that of the application system developer, or that of the end-user. Depending on one's perspective, varying levels of concern will be focused on issues such as system development cost, portability, extensibility, maintainability, linguistic functionality in the areas of syntax, semantics, discourse, pragmatics, and knowledge acquisition; reliability; help facilities; performance speed; and robustness. The papers presented in this collection focus on evaluation of linguistic functionality of NLP systems, in some instances on the integration of functionality evaluation during the development process.

### **Workshop Description**

The Natural Language Processing Systems Evaluation Workshop provided a forum for computational linguists to discuss current evaluation efforts and activities, research progress, and new approaches; promoted scientific interchange on important evaluation issues; and generated recommendations and directions for future investigations in the evaluation area. The Workshop, sponsored by Rome Laboratory and attended by over 60 people, was held in conjunction with the 29th Annual Meeting of the Association for Computational Linguistics (ACL) on 18 June 1991 at the Berkeley Campus of the University of California.

The Workshop Call for Participation sought presentations focused on evaluation-relevant issues that include: the identification of NLP capabilities requiring "measurement", evolving or contrived evaluation criteria, and descriptions of current evaluation practices and experiences. The papers in this volume are formal records of the presentations made at the Workshop.

DARPA-sponsored Message Understanding Conferences form a major program toward resolving issues of text understanding system evaluation. The original conference, in 1987, had participants training and testing their systems on Navy RAINFORM messages. Ten messages made up the training set and two additional messages were distributed when the participants assembled at the Naval Ocean Systems Center (NOSC) for system extension and evaluation. The second Message Understanding Conference, in 1989, presented 105 Navy OPREP messages for training and 25 for testing. Recently, MUC-3 used a training set of 1300

texts forming a corpus of over 2.5 MB and a test set of 100 previously unseen texts. The paper "Third Message Understanding Evaluation and Conference (MUC-3): Methodology and Test Results" presents the activities and results of the most recent encounter, held at NOSC in San Diego, California, on 21-23 May 1991. "MUC-3 Evaluation Metrics and Linguistic Phenomena Tests" delves into the details of the measurements made during that event.

The paper "A Developing Methodology for the Evaluation of Spoken Language Systems" discusses a program, supported by DARPA and the National Institute of Science and Technology (NIST), for evaluating Spoken Language Systems (SLS) on a database query task using the Air Travel Information System (ATIS). Both the ATIS and MUC evaluation programs collected a large data set, reserved a portion for testing purposes and used the major portion for training, developed agreement on judging correct system outputs, distributed the test set for administration at multiple sites, used automated scoring techniques, and report numerical scores (recall and precision for MUC-3; number right, wrong, and not answered as well as a weighted error percentage for ATIS). The paper "Multi-Site Natural Language Processing Evaluation: MUC and ATIS" further compares and contrasts the two evaluation programs.

In contrast to the task-specific and domain-dependent approach to evaluation used in MUC and ATIS, "The Benchmark Investigation/Identification Program" discusses a method of evaluating NLP systems that is being designed for applicability across task types, across domains, and to all different types of NLP systems. The developing evaluation methodology will not require the NLP systems to be re-engineered or adapted to a particular domain or text corpus, and will provide coverage of NLP capabilities across the categories of syntax, semantics, discourse, and pragmatics. Furthermore, the method provides detailed capability profiles of the NLP systems evaluated, with quantitative results for each item in the profile. The authors report on the development of the evaluation procedure and the results of actual use of the evaluation procedure with three NLP systems.

"Evaluating Syntax Performance of Parser/Grammars of English" reports on a collaborative effort on the part of representatives of nine institutions to develop criteria, methods, measures and procedures for evaluating the syntax performance of parser/grammars. The project has progressed to the point where the first version of an automated syntax evaluation procedure has been completed and is available for testing. The procedure judges a parse only on the basis of constituent boundaries and yields two principal measures: crossing parentheses and recall.

The paper "A Diagnostic Tool for German Syntax" describes an effort to construct a catalogue of syntactic data which should eventually exemplify the major syntactic patterns of the German language. The data set is being systematically constructed, rather than collected from naturally occurring text, so as to assure coverage of syntactic phenomena and optimize

the data set. In the future, the data set will be used for error detection and system evaluation. The effort has some similarity to the Benchmark Investigation/Identification program, mentioned above, in that the emphasis is on linguistic capabilities, categorized in a systematic manner, using "hand-constructed" test data.

The next two papers discuss evaluation of semantic analysis and dialogue performance, respectively. They do not report on methods that have been implemented but rather, discuss important issues and propose approaches to the evaluation process. "Preliminaries to the Development of Evaluation Metrics for Natural Language Semantic and Pragmatic Analysis Systems" discusses the categorization of semantic and pragmatic analysis capabilities that must precede the development of test suites or testing methods. The paper "Corpus-Based Evaluation of the Sundial System" discusses the importance that has been placed on the use of corpus collection and analysis in the development and evaluation of the Sundial System. The Sundial development project is a multi-national European Spoken Language project. Although the Sundial development group has not yet implemented a method of evaluating dialogue management performance, the author discusses the issues and proposes primary criteria for evaluation.

The papers "Module-Level Testing: Lessons Learned" and "Maintaining and Enhancing a NL DBMS Interface" address the issue of evaluation for the purpose of supporting the on-going development of a particular system. The first stresses the use of module-level testing on carefully constructed test suites that are representative of the actual corpus the system must handle and a scoring methodology that provided sufficient information to support system improvement and continued development. "Maintaining and Enhancing a NL DBMS Interface" discusses a maintenance facility that includes a large test query collection organized around their NLP system's general lexicon of words and phrases. Although the system is based on Conceptual Analysis and does not have a syntactic analysis module like the system of the previous paper, the maintenance facility does provide information concerning certain stages of processing via the storage of intermediate data structures. Both papers "speak from experience" and provide good insights into the problems of controlling and tracking the development of large NLP systems.

Evaluation programs for natural language generation (NLG) technology lag behind those for NLU, having neither funding nor a consensus on evaluation criteria and methodology. Issues hindering NLG evaluation are discussed in "Evaluation for Generation" and "Evaluating Natural Language Generation Facilities in Intelligent Systems". The impetus for, and the problems surrounding determination of, an NLG evaluation methodology are summarized in the first of these papers. The second NLG paper discusses the problems associated with devising a standard set of NLG evaluation criteria and supports the use of a

task-based approach instead. The paper outlines the approach in the context of a current study.

A transfer-based (or direct transfer) Machine Translation system translates from one human language to another without using an intermediate representation language. Transfer-based systems select word translations and then use rules to adjust the result of those word replacements to form a sentence in the target language. MT systems with "good compositionality" are those requiring fewer adjustment rules. The paper "Measuring Compositionality in Transfer-Based Machine Translation Systems" describes a methodology applied to objectively evaluate the compositionality of a transfer-based MT system.

Next, under the intriguing title of "Good Applications for Crummy Machine Translation," prevailing MT metrics are reviewed and a convincing argument is presented endorsing the use of task-dependent evaluation criteria. To demonstrate this approach, the appropriateness of various metrics for Machine (Aided) Translation for a range of applications is noted.

Our final paper in the Proceedings contrasts with the others in that it reports on a community resource and addresses the evaluation of NLP systems based on criteria that differ from those of the other papers. "Gross-Grained Software Evaluation: The Natural Language Software Registry" reports on the NL Software Registry sponsored by the Association for Computational Linguistics and recently established at the University of Chicago's Center for Information and Language Studies. Its purpose is to facilitate the exchange and evaluation of NLP software, both commercial and noncommercial. A concise and uniform summary of the 33 software items that have been submitted has provided a better understanding of some current evaluation criteria. This gross-grained approach to evaluation was supplemented by extensive testing and review of selected software.

### **Summary/Conclusions**

The Workshop well-achieved its goal of gathering researchers representing a wide range of NLP-related interests. The papers and active support of the workshop participants attest to research community awareness that NLP technology, having matured to the point of utility for certain circumscribed applications, has a growing need for the formulation of evaluation methodologies.

We were fortunate to have one of the earliest reports on the results of MUC-3. Having taken the initiative in developing evaluation methodologies, MUC events have played an inestimable role in the development of criteria and methodology for text processing systems. Comments from MUC-3 participants reinforce the notion of value in evaluating NLP systems for stimulating ideas and identifying high payoff processing techniques or problem areas. Results

indicate, for example, that the majority of the high-scoring systems in MUC-3 use robust parsing. And, when asked to identify the key problem their system faced, that is, where their energies would best be focused during the next year (until MUC-4), "discourse analysis" was the popular response [Iwanska et. al., 1991].

In reading the Workshop papers, the relationship between the efforts cataloging specific NLU capabilities and the optional appositives test described in the second of the two MUC-3 papers should be noted. There may be synergistic benefits to be gained from the insinuation of such cataloging efforts into MUC-4 or similar evaluation activities of the future.

Many new evaluation projects and activities have begun since the December 1988 Workshop on the Evaluation of NLP Systems held at Wayne, PA [Palmer, et al; 1989]. The response to the call within the report for that workshop for "rigorous accounts" of linguistic phenomena, such as that offered for discourse by [Webber, 1988], is strongly reflected within this volume. Also, large collections of text and other resources (such as the Software Registry described in the last paper here) have been recognized as crucial and cost-effective for future NLP development and evaluation efforts. Workshop discussion was interspersed with mention of a number of text collection initiatives (which were more fully described later in the week to ACL attendees during the business meeting). For example, recognizing the need for large amounts of linguistic data for speech and text processing research, DARPA is in the process of establishing the Linguistic Data Consortium (LDC) to develop and distribute speech and text corpuses, lexicons, and grammars. The goal of the ACL Data Collection Initiative (ACL/DCI) to acquire large text corpuses with a total of over 100 million words has been surpassed; and, the TREEBANK Project at the University of Pennsylvania operates to provide linguistic analyses of vast quantities of spoken and written text. Further information on the ACL/DCI, the TREEBANK Project, and the Text Encoding Initiative (TEI), an international project to "formulate and disseminate guidelines for the preparation and interchange of machine-readable texts," can be found in [Walker, 1990].

Workshop attendees commented on the lack of MUC like evaluation programs for NLG technology. This deficiency was attributed to insufficient funding support and the lack of an evaluation methodology. Efforts toward defining feasible measures and testing methods are being made, however, as evidenced by the mention in the enclosed paper, "Evaluation for Generation", of recent workshops focused on those goals.

Workshop participants concluded that some important and appropriate steps are being taken, but much more remains to be done toward developing evaluation methodologies for NLP. A fact to be kept in mind is that evaluation standards are not created or defined by decree but must evolve and earn community acceptance over time. The work described in the papers in this Proceedings are taking some critical steps in that process. Workshop participants, in

particular, the authors of this introductory paper, seek feedback from the research community on their projects and research directions. We encourage reader comments.

Our sincere thanks go to all who participated in the Workshop: the Workshop speakers/authors; fellow members of the Workshop Organizing Committee, namely, Christine Montgomery, Tim Finin, and Ralph Grishman; the Association for Computational Linguistics, particularly Peter Norvig and Don Walker; and the University of California at Berkeley.

### Bibliography

BBN Systems and Technologies Corporation; Draft Corpus for Testing NL Data Base Query Interfaces, NL Evaluation Workshop, Wayne, PA, Dec. 1988.

Flickinger, D., Nerbonne, J., Sag, I., and Wasow, T.; Toward Evaluation of Natural Language Processing Systems; Hewlett-Packard Laboratories Technical Report, 1987.

Hendrix, G.G., Sacerdoti, E.D. and Slocum, J.; Developing a Natural Language Interface to Complex Data, Artificial Intelligence Center Technical Report, SRI International, 1976.

Iwanska, Lucja, Doug Appelt, Damaris Ayuso, Kathy Dahlgren, Bonnie Glover Stalls, Ralph Grishman, George Krupka, Christine Montgomery, and Ellen Riloff; Computational Aspects of Discourse in the Context of MUC-3, (to be published).

Lehrberger, John and Laurent Bourbeau; Machine Translation: Linguistic Characteristics of MT systems and General Methodology of Evaluation, Philadelphia: John Benjamins Publishing Company, 1988.

Malhotra, A.; Design Criteria for a Knowledge-Based Language System for Management: An Experimental Analysis, MIT/LCS/TR-146, 1975.

Palmer, Martha, Tim Finin, and Sharon M. Walter; Workshop on the Evaluation of Natural Language Processing Systems, RADC Final Technical Report #89-302, 1989.

Walker, Donald E.; Collecting Texts, Tagging Texts, and Putting Texts in Context; Working Notes: AAAI Spring Symposium Series (Text-Based Intelligent Systems), 1990.

Webber, Bonnie Lynn; Discourse Canon, Presented at the Mohonk Darpa Workshop, 1988.

Wetschedel, Ralph M.; Evaluating Natural Language Interfaces to Expert Systems, Proceedings of 1986 IEEE International Conference on Systems, Man, and Cybernetics.



# THIRD MESSAGE UNDERSTANDING EVALUATION AND CONFERENCE (MUC-3): METHODOLOGY AND TEST RESULTS

*Beth M. Sundheim*

Naval Ocean Systems Center  
Code 444  
Decision Support and AI Technology Branch  
San Diego, CA 92152-5000  
sundheim@nosc.mil

## INTRODUCTION

The Naval Ocean Systems Center (NOSC) has conducted the third in a series of evaluations of English text analysis systems. These evaluations are intended to advance our understanding of the merits of current text analysis techniques, as applied to the performance of a realistic information extraction task. The latest one is also intended to provide insight into information retrieval technology (document retrieval and categorization) used instead of or in concert with language understanding technology. The inputs to the analysis/extraction process consist of naturally-occurring texts that were obtained in the form of electronic messages. The outputs of the process are a set of templates or semantic frames resembling the contents of a partially formatted database.

The premise on which the evaluations are based is that task-oriented tests enable straightforward comparisons among systems and provide useful quantitative data on the state of the art in text understanding. The tests are designed to treat the systems under evaluation as black boxes and to point up system performance on discrete aspects of the task as well as on the task overall. These quantitative data can be interpreted in light of information known about each system's text analysis techniques in order to yield qualitative insights into the relative validity of those techniques as applied to the general problem of information extraction.

This paper presents an overview of the evaluation and its results. A MUC-3 conference proceedings will be published by Morgan Kaufmann that includes the complete set of overall test scores, some analysis by the participants of their test results, and system descriptions.

## OVERVIEW

The third evaluation began in October, 1990. A dry-run phase was completed in February, 1991, and final testing was carried out in May, 1991, concluding with the Third Message Understanding Conference (MUC-3). This evaluation was significantly broader in scope than previous ones in most respects, including text characteristics, task specifications, performance measures, and range of text understanding and information extraction techniques. The corpus and task are sufficiently challenging that they will be used again (with a new test set) in a

future evaluation, which will seek to measure improvements in performance by MUC-3 systems and establish performance baselines for any new systems.

A call for participation was sent to organizations in the U.S. that were known to be engaged in system design or development in the area of text analysis or information retrieval. Twelve of the sites that responded participated in the dry run and reported results at a meeting held in February, 1991, which also served as a forum for resolving issues that affected the test design, scoring, etc. for the final testing in May. One site dropped out after the dry run, and four new sites entered. The Defense Advanced Research Projects Agency (DARPA), which funded NOSC to conduct the evaluation, also provided partial financial support to two-thirds of the participating sites.

Pure and hybrid systems based on a wide range of text interpretation techniques (e.g., statistical, key-word, template-driven, pattern-matching, in-depth natural language processing) were represented in the MUC-3 evaluation. The fifteen sites that completed the evaluation are Advanced Decision Systems (Mountain View, CA), BBN Systems and Technologies (Cambridge, MA), General Electric (Schenectady, NY), GTE (Mountain View, CA), Intelligent Text Processing, Inc. (Santa Monica, CA), Hughes Research Laboratories (Malibu, CA), Language Systems, Inc. (Woodland Hills, CA), McDonnell Douglas Electronic Systems (Santa Ana, CA), New York University (New York City, NY), PRC, Inc. (McLean, VA), SRI International (Menlo Park, CA), Synchronetics, Inc. together with the University of Maryland (Baltimore, MD), Unisys Center for Advanced Information Technology (Paoli, PA), the University of Massachusetts (Amherst, MA), and the University of Nebraska (Lincoln, NE) in association with the University of Southwest Louisiana (Lafayette, LA). In addition, an experimental prototype of a probabilistic text categorization system was developed by David Lewis, who is now at the University of Chicago, and was tested along with the other systems.

## CORPUS AND TASK

The corpus was formed via a keyword query to an electronic database containing articles in message format that had originated from open sources worldwide. These articles had been compiled, translated (if necessary), edited, and disseminated by the Foreign Broadcast Information Service of the U.S. Government. A training set of 1300 texts was identified, and additional texts were set aside for use as test data. The corpus presents realistic challenges in terms of its overall size (over 2.5 megabytes), the length of the individual articles (approximately a half-page each on average), the variety of text types (newspaper articles, TV and radio news, speech and interview transcripts, rebel communiques, etc.), the range of linguistic phenomena represented (both well-formed and ill-formed), and the open-ended nature of the vocabulary (especially with respect to proper nouns).

The task was to extract information on terrorist incidents (incident type, date, location, perpetrator, target, instrument, outcome, etc.) from the relevant texts in a blind test on 100 previously unseen texts. Approximately half of the articles were irrelevant to the task as it was defined; scoring penalties were exacted for failures to correctly determine relevancy (see following section). The extracted information was to be represented in a template in one of several ways, according to the data format requirements of each template slot. Some slot fills were required to be categories from a predefined set of possibilities (e.g., for the various types of

terrorist incidents such as BOMBING, ATTEMPTED BOMBING, BOMB THREAT); others were required to be canonicalized forms (e.g., for dates) or numbers; still others were to be in the form of strings (e.g., for person names).

A relatively simple article and corresponding answer-key template are shown in Figures 1 and 2. Note that the text in Figure 1 is all upper case, that the dateline includes the source of the article ("Inravisión Television Cadena 1") and that the article is a news report by Jorge Alonso Sierra Valencia. In Figure 2, the left-hand column contains the slot labels, and the right-hand column contains the correct answers as defined by NOSC. Slashes mark alternative correct responses (systems are to generate just one of the possibilities), an asterisk marks slots that are inapplicable to the incident type being reported, a hyphen marks a slot for which the text provides no fill, and a colon introduces the cross-reference portion of a fill (except for slot 16, where the colon is used as a separator between more general and more specific place names).

The participants collectively created the answer key for the training set, each site manually filling in templates for partially overlapping subset of the texts. This task was carried out at the start of the evaluation; it therefore provided participants with good training on the task requirements and provided NOSC with good early feedback. Generating and cross-checking the templates required an investment of at least two person-weeks of effort per site. These answer keys were updated a number of times to reduce errors and to maintain currency with changing template fill specifications. In addition to generating answer key templates, sites were also responsible for compiling a list of the place names that appeared in their set of texts; NOSC then merged these lists to create the options for filling the TARGET: FOREIGN NATION slot and LOCATION OF INCIDENT slot.

TST1-MUC3-0080

BOGOTA, 3 APR 90 (INRAVISION TELEVISION CADENA 1) -- [REPORT] [JORGE ALONSO SIERRA VALENCIA] [TEXT] LIBERAL SENATOR FEDERICO ESTRADA VELEZ WAS KIDNAPPED ON 3 APRIL AT THE CORNER OF 60TH AND 48TH STREETS IN WESTERN MEDELLIN, ONLY 100 METERS FROM A METROPOLITAN POLICE CAI (IMMEDIATE ATTENTION CENTER). THE ANTIOQUIA DEPARTMENT LIBERAL PARTY LEADER HAD LEFT HIS HOUSE WITHOUT ANY BODYGUARDS ONLY MINUTES EARLIER. AS HE WAITED FOR THE TRAFFIC LIGHT TO CHANGE, THREE HEAVILY ARMED MEN FORCED HIM TO GET OUT OF HIS CAR AND GET INTO A BLUE RENAULT.

HOURS LATER, THROUGH ANONYMOUS TELEPHONE CALLS TO THE METROPOLITAN POLICE AND TO THE MEDIA, THE EXTRADITABLES CLAIMED RESPONSIBILITY FOR THE KIDNAPPING. IN THE CALLS, THEY ANNOUNCED THAT THEY WILL RELEASE THE SENATOR WITH A NEW MESSAGE FOR THE NATIONAL GOVERNMENT.

LAST WEEK, FEDERICO ESTRADA VELEZ HAD REJECTED TALKS BETWEEN THE GOVERNMENT AND THE DRUG TRAFFICKERS.

Figure 1. Article from MUC-3 Corpus<sup>1</sup>

<sup>1</sup>This article has serial number PA0404072690 in the Latin America volume of the Foreign Broadcast Information Service Daily Reports, which are the secondary source for all the texts in the MUC-3 corpus.

0. MESSAGE ID	TST1-MUC3-0080
1. TEMPLATE ID	1
2. DATE OF INCIDENT	03 APR 90
3. TYPE OF INCIDENT	KIDNAPPING
4. CATEGORY OF INCIDENT	TERRORIST ACT
5. PERPETRATOR: ID OF INDIV(S)	"THREE HEAVILY ARMED MEN"
6. PERPETRATOR: ID OF ORG(S)	"THE EXTRADITABLES" / "EXTRADITABLES"
7. PERPETRATOR: CONFIDENCE	CLAIMED OR ADMITTED: "THE EXTRADITABLES" / "EXTRADITABLES"
8. PHYSICAL TARGET: ID(S)	•
9. PHYSICAL TARGET: TOTAL NUM	•
10. PHYSICAL TARGET: TYPE(S)	•
11. HUMAN TARGET: ID(S)	"FEDERICO ESTRADA VELEZ" ("LIBERAL SENATOR" / "ANTIOQUIA DEPARTMENT LIBERAL PARTY LEADER" / "SENATOR" / "LIBERAL PARTY LEADER" / "PARTY LEADER")
12. HUMAN TARGET: TOTAL NUM	1
13. HUMAN TARGET: TYPE(S)	GOVERNMENT OFFICIAL / POLITICAL FIGURE: "FEDERICO ESTRADA VELEZ"
14. TARGET: FOREIGN NATION(S)	-
15. INSTRUMENT: TYPE(S)	•
16. LOCATION OF INCIDENT	COLOMBIA: MEDELLIN (CITY)
17. EFFECT ON PHYSICAL TARGET(S)	•
18. EFFECT ON HUMAN TARGET(S)	-

Figure 2. Answer Key Template

## MEASURES OF PERFORMANCE

All systems were evaluated on the basis of performance on the information extraction task in a blind test at the end of each phase of the evaluation. It was expected that the degree of success achieved by the different techniques in May would depend on such factors as whether the number of possible slot fillers was small, finite, or open-ended and whether the slot could typically be filled by fairly straightforward extraction or not. System characteristics such as amount of domain coverage, degree of robustness, and general ability to make proper use of information found in novel input were also expected to be major factors. The dry-run test results were not assumed to provide a good basis for estimating performance on the final test in May, but the expectation was that most, if not all, of the systems that participated in the dry run would show dramatic improvements in performance. The test results show that some of these expectations were borne out, while others were not or were less significant than expected.

A semi-automated scoring program was developed under contract for MUC-3 to enable the calculation of the various measures of performance. It was distributed to participants early on during the evaluation and proved invaluable in providing them with the performance feedback necessary to prioritize and reprioritize their development efforts as they went along. The scoring program can be set up to score all the templates that the system generates or to score subsets of templates/slots. User interaction is required only to determine whether a mismatch between the system-generated templates and the answer-key templates should be judged completely or partially correct. (A partially correct filler for slot

11 in Figure 2 might be "VELEZ" ("LEADER"), and a partially correct filler for slot 16 would be simply COLOMBIA.) An extensive set of interactive scoring guidelines was developed to standardize the interactive scoring. The scoring program maintains a log of interactions that can be used in later scoring runs and augmented by the user as the system is updated and the system-generated templates change.

The two primary measures of performance were completeness (recall) and accuracy (precision). There were two additional measures, one to isolate the amount of spurious data generated (overgeneration) and the other to determine the rate of incorrect generation as a function of the number of opportunities to incorrectly generate (fallout). The labels "recall," "precision," and "fallout" were borrowed from the field of information retrieval, but the definitions of those terms had to be substantially modified to suit the template-generation task. The overgeneration metric has no correlate in the information retrieval field, i.e., a MUC-3 system can generate indefinitely more data than is actually called for, but an information retrieval system cannot retrieve more than the total number of items (e.g., documents) that are actually present in the corpus.

Fallout can be calculated only for those slots whose fillers form a closed set. Scores for the other three measures were calculated for the test set overall, with breakdowns by template slot. Figure 3 presents a somewhat simplified set of definitions for the measures.

MEASURE	DEFINITION
RECALL	$\frac{\# \text{correct fills generated}}{\# \text{fills in key}}$
PRECISION	$\frac{\# \text{correct fills generated}}{\# \text{fills generated}}$
OVERGENERATION	$\frac{\# \text{spurious fills generated}}{\# \text{fills generated}}$
FALLOUT	$\frac{\# \text{incorrect+spurious generated}}{\# \text{possible incorrect fills}}$

Figure 3. MUC-3 Scoring Metrics

The most significant thing that this table does not show is that precision and recall are actually calculated on the basis of points -- the term "correct" includes system responses that matched the key exactly (earning 1 point each) and system responses that were judged to be a good partial match (earning .5 point each). It should also be noted that overgeneration is not only a measure in its own right but is also a component of precision, where it acts as a penalty by contributing to the denominator. Overgeneration also figures in fallout by contributing to the numerator. Further information on the MUC-3 evaluation metrics, including information on three different ways penalties for missing and spurious data were assigned, can be found elsewhere in this volume in the paper by Nancy Chinchor.

## TEST PROCEDURE

Final testing was done on a test set of 100 previously unseen texts that were representative of the corpus as a whole. Participants were asked to copy the test

package electronically to their own sites when they were ready to begin testing. The testing had to be conducted and the results submitted within a week of the date when the test package was made available for electronic transfer. Each site submitted their system-generated templates, the outputs of the scoring program (score reports and the interactive scoring history file), and a trace of the system's processing (whatever type of trace the system normally produces that could serve to help validate the system's outputs). Initial scoring was done at the individual sites, with someone designated as interactive scorer who preferably had not been part of the system development team. After the conference, the system-generated templates for all sites were labeled anonymously and rescored by two volunteers in order to ensure that the official scores were obtained as consistently as possible.

The system at each site was to be frozen before the test package was transferred; no updates were permitted to the system until testing and scoring were completed. Furthermore, no backing up was permitted during testing in the event of a system error. In such a situation, processing was to be aborted and restarted with the next text. A few sites encountered unforeseen system problems that were easily pinpointed and fixed. They reported unofficial, revised test results at the conference that were generally similar to the official test results and do not alter the overall picture of the current state of the art.

The basic test called for systems to be set up to generate templates that produced the "maximum tradeoff" between recall and precision, i.e., templates that achieved scores as high as possible and as similar as possible on both recall and precision. This was the normal mode of operation for most systems and for many was the *only* mode of operation that the developers had tried. Those sites that *could* offer alternative tradeoffs were invited to do so, provided they notified NOSC in advance of the particular setups they intended to test on.

In addition to the scores obtained for these metrics on the basic template-generation task, scores were obtained of system performance on the linguistic phenomenon of apposition, as measured by the template fills generated by the systems in particular sets of instances. That is, sentences exemplifying apposition were marked for separate scoring if successful handling of the phenomenon seemed to be required in order to fill one or more template slots correctly for that sentence. This test was conducted as an experiment and is described in the paper by Nancy Chinchor elsewhere in this volume.

## TEST RESULTS AND DISCUSSION

Scatter plots for selected portions of the final test results are shown in the appendix. The data points are labeled with abbreviated names of the 15 sites, and optional test runs are marked with the site's name and an "O" extension. The plots present an interesting picture of the MUC-3 results as a whole, but the significance of the numbers for each of the tested systems needs to be assessed on the basis of a careful reading of the MUC-3 proceedings papers that were submitted by each of the sites. The level of effort that could be afforded by each of the sites varied considerably, as did the maturity of the systems at the start of the evaluation. All sites were operating under time constraints imposed by the evaluation schedule. In addition, the evaluation demands were a consequence of the intricacies of the task and of general corpus characteristics such as the following:

- \* The texts that are relevant to the MUC-3 task (comprising approximately 50% of the total corpus) are likely to contain more than one relevant incident.

- \* The information on a relevant incident may be dispersed throughout the text and may be intertwined with accounts of other (relevant or irrelevant) incidents.

- \* The corpus includes a mixture of material (newspaper articles, TV news, speeches, interviews, propaganda, etc.) with varying text structures and styles.

The scoring program produces four sets of overall scores, three of which are based on different means of assessing penalties for missing and spurious data. The set called Matched/Missing is a compromise between two others and is used as the official one for reporting purposes. Figure A1 is based on the Matched/Missing method of assessing penalties. The fourth method does the scoring only for those slots that require set fills, i.e., fills that come from predefined sets of categories. Figure A2 is based on that method of scoring. The various methods are described more fully in the paper by Nancy Chinchor.

Figure A1 gives the most general picture of the results of MUC-3 final testing. It shows that precision always exceeds recall and that the systems with relatively high recall are also the ones that have relatively high precision. The latter fact inspires an optimistic attitude toward the promise of at least some of the techniques employed by today's systems -- further efforts to enhance existing techniques and extend the systems' domain coverage may lead to significantly improved performance on both measures. However, since all systems show better precision than recall, it appears that it will be a bigger challenge to obtain very high recall than it will be to achieve higher precision at recall levels that are similar to those achievable today.

The distribution of data points tentatively supports at least one general observation about the technologies that underly today's systems: those systems that use purely stochastic techniques or handcrafted pattern-matching techniques were not able to achieve the same level of performance for MUC-3 as some of the systems that used parsing techniques. The "non-parsing" systems are ADS, HU, MDC, UNI, UNL, UNL-O1, and UNL-O2, and the "parsing" systems are BBN, BBN-O, GE, GTE, ITP, LSI, NYU, NYU-O1, NYU-O2, PRC, SRI, SYN, UMA, and UMA-O.

Further support for this observation can be found in Figure A2, where the scores are computed for all slots requiring set fills, and in Figure A3, which shows the scores for just one of those set-fill slots. In these cases, one might expect the non-parsing systems to compare more favorably with the parsing systems, since the fill options are restricted to a fairly small, predefined set of possibilities. However, none of the non-parsing systems appears at the leading edge in Figure A2, and the only non-parsing system in the cluster at the leading edge in Figure A3 is ADS (which shares a data point with NYU-O2), although a few non-parsing systems have extremely high precision scores (UNI, UNL, UNL-O1, and UNL-O2).

On the other hand, there is quite a range in performance even among the systems in the parsing group, all of which had to cope with having limited coverage of the domain. One thing that is apparent from the sites' system descriptions is that all the ones on the leading edge in Figure A1 have the ability to make good use of partial sentence parses when complete parses cannot be obtained. Level of effort is also an indicator of performance success, though not a completely

reliable one: GE, NYU, and UMass all reported investing more than one person-year of effort on MUC-3, but several other sites with lower overall performance also reported just under or over one person-year of effort.

It must be said that there were some extremely immature systems in the non-parsing group and the parsing group alike, so any general conclusions must be taken as tentative and should certainly not be used to form opinions about the relative validity of isolated techniques employed by the individual systems in each group. It could be that the relatively low-performing systems use extremely effective techniques that, if supplemented by other known techniques or supported by more extensive domain coverage, would put the system well out in front. One should also not assume that the systems at the leading edge are similar kinds of systems. In fact, those systems have quite different architectures and have varying sizes of lexicons, kinds of parsers and semantic interpreters, etc.

In addition to showing how system performance varies from one slot to another, Figures A3, A4 and A5 show how spurious data generation combines with incorrect data generation to affect the precision scores in different kinds of slots. Figure A4 is for the **TEMPLATE ID** slot. The fillers of this slot are arbitrary numbers that uniquely identify the templates for a given message. The scoring program disregards the actual values and finds the best match between the system-generated templates and the answer key templates for a given message based on the degree of match in fillers of other slots in the template. Since there is no such thing as an *incorrect* template ID, only a *spurious* or *missing* template, and since missing data plays no role at all in computing precision, the only penalty to precision for the **TEMPLATE ID** slot is due to spurious data generation. In contrast to the **TEMPLATE ID** slot, the **TYPE OF INCIDENT** slot (Figure A3) shows no influence of spurious data on precision at all. This is because the **TYPE OF INCIDENT** slot permits only one filler. The **HUMAN TARGET: ID(S)** slot (Figure A5) can be filled with indefinitely many fillers and thus shows the impact of both incorrect and spurious data on precision.

Four sites submitted results for the optional test runs mentioned in the previous section -- BBN Systems and Technologies (BBN-O), New York University (NYU-O1 and NYU-O2), the University of Massachusetts (UMA-O), and the University of Nebraska/University of Southwestern Louisiana (UNL-O1 and UNL-O2). These sites conducted radically different experiments to generate templates more conservatively. The BBN-O experiment largely involved doing a narrower search in the text for the template-filling information; the NYU-O1 and NYU-O2 experiments involved throwing out templates in which certain key slots were either unfilled or were filled with information that indicated an irrelevant incident with good probability; the UMA-O experiment bypassed a case-based reasoning component of the system; and the UNL-O1 and UNL-O2 experiments involved the usage of different thresholds in their connectionist framework. The experiments resulted in predicted differences in the Matched/Missing scores compared to the basic test. In almost all cases the experiments had the overall effect of lowering recall; in all cases they lowered overgeneration and thereby raised precision. Figure A4 shows the marked difference the experiments made in spurious template generation; Figure A1 shows the much smaller difference they made in overall recall and precision.



## CONCLUSIONS

The MUC-3 evaluation established a solid set of performance benchmarks for systems with diverse approaches to text analysis and information extraction. The MUC-3 task was extremely challenging, and the results show what can be done with today's technologies after only a modest domain- and task-specific development effort (on the order of one person-year). On a task this difficult, the systems that cluster at the leading edge were able to generate in the neighborhood of 40-50% of the expected data and to do it with 55-65% accuracy. Breakdowns of performance by slot show that performance was best on identifying the type of incident -- 70-80% completeness and 80-85% accuracy were achieved, and accuracy figures in the 90-100% range were possible with some sacrifice in completeness.

All of the MUC-3 system developers are optimistic about the prospects for seeing steady improvements in system performance for the foreseeable future. This feeling is based variously on such evidence as the amount of improvement achieved between the dry-run test and the final test, the slope of improvement recorded on internal tests conducted at intervals during development, and the developers' own awareness of significant components of the system that they had not had time to adapt to the MUC-3 task. The final test results are consistent with the claim that most systems, if not all, may well be still on a steep slope of improvement. However, they also show that performance on recall (completeness) is not as good as performance on precision (accuracy), and they lend support to the possibility that this discrepancy will persist. It appears that systems cannot be built today that are capable of obtaining high overall recall, even at the expense of outrageously high overgeneration. Systems can, however, be built that will do a good job at potentially useful subtasks such as identifying terrorist incidents of various kinds.

The results give at least a tentative indication that systems incorporating robust parsing techniques show more long-term promise of high performance than non-parsing systems. However, there are great differences in techniques among the systems in the parsing and non-parsing groups and even among those robust parsing systems that did the best in optimizing the overall tradeoff between recall and precision. Further variety was evident in the optional test runs conducted by some of the sites. Those runs show promise for the development of systems that can be "tuned" in various ways to generate data more aggressively or more conservatively, yielding tradeoffs between recall and precision that respond to differences in emphasis in real-life applications.

## ACKNOWLEDGEMENTS

This work was funded by DARPA under ARPA order 6359. The author is indebted to all the organizations that participated in MUC-3 and especially to certain individuals who contributed extra time and energy to ensure the evaluation's success, among them Laura Balcom, Nancy Chinchor, Ralph Grishman, Pete Halverson, Lynette Hirschman, Jerry Hobbs, Cheryl Kariya, George Krupka, David Lewis, Lisa Rau, John Sterling, Charles Wayne, and Carl Weir. Thanks are also due to Catherine MacLeod, Steve Dennis, and Mary Ellen Okurowski, who commented on an earlier version of this paper, and to Eric Scott, who assembled the appendix.

## APPENDIX

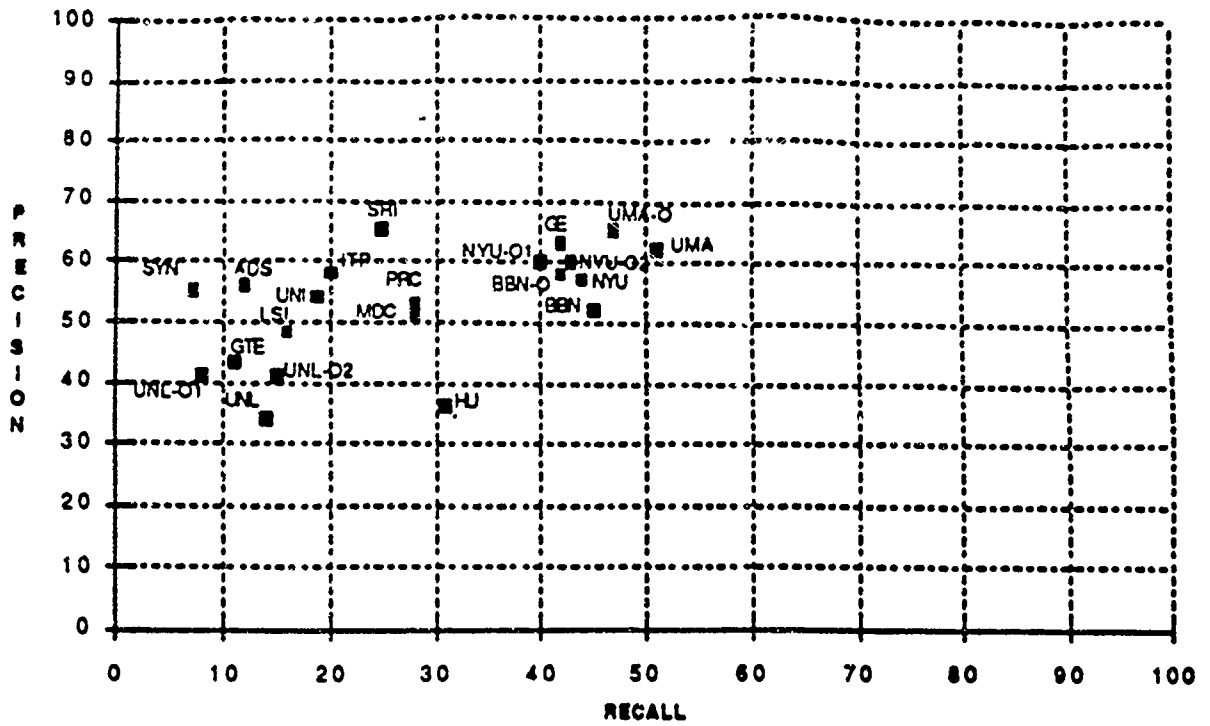


Figure A1. Overall Recall vs Precision (Matched/Missing Row)

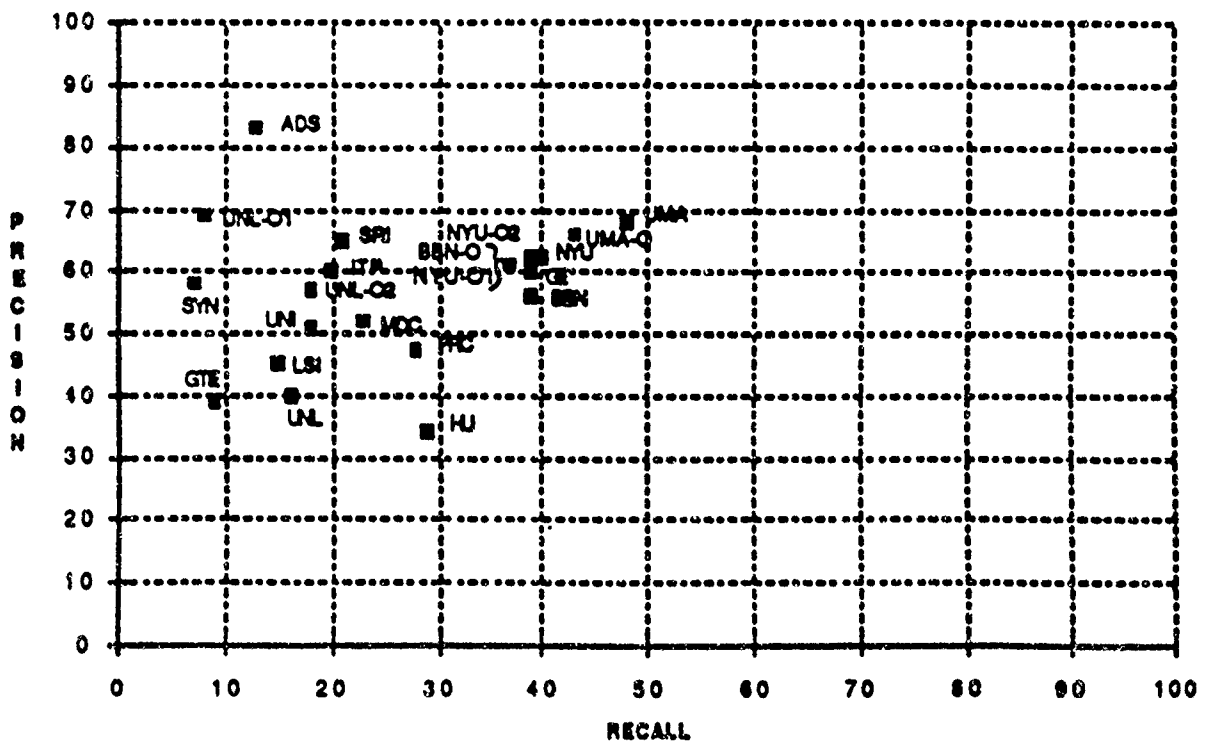


Figure A2. Overall Recall vs Precision (Set Fills Only Row)

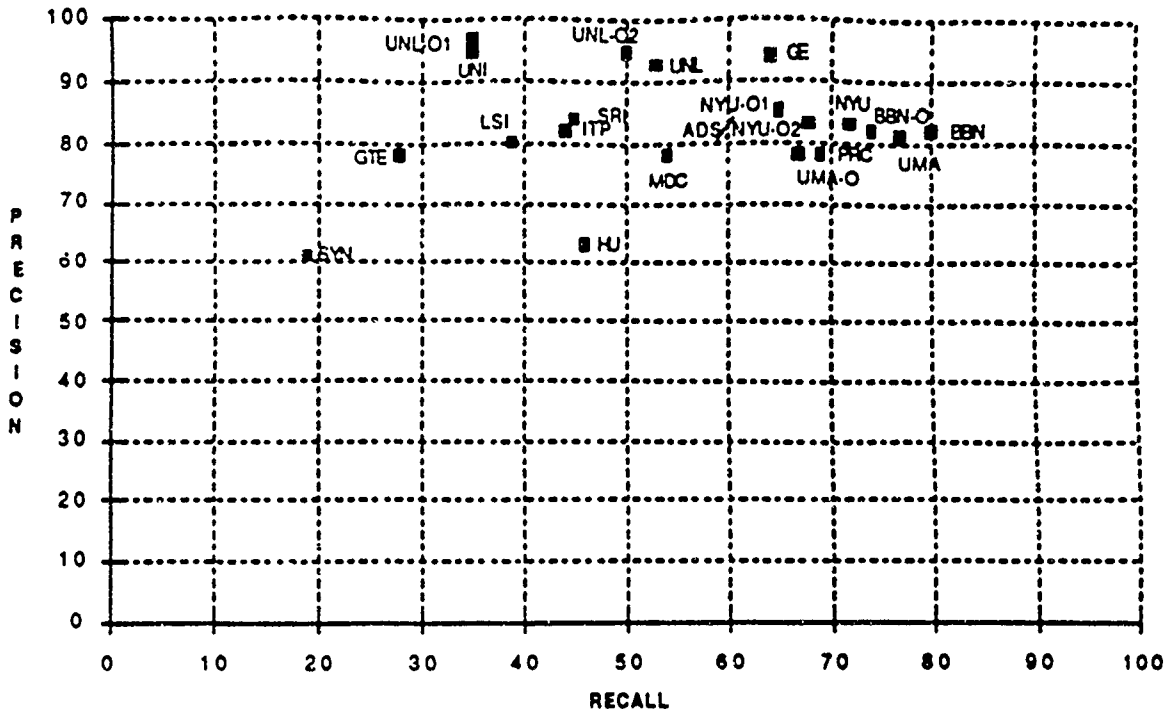


Figure A3. Recall vs Precision for Incident Type Slot

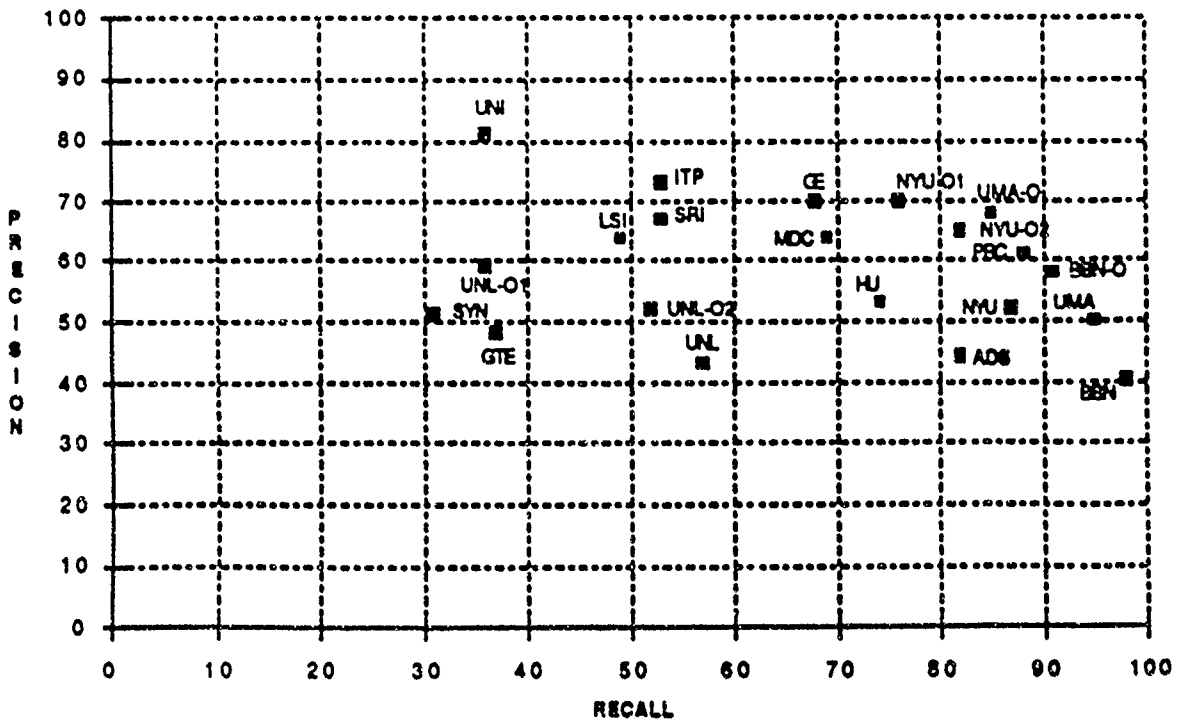


Figure A4. Recall vs Precision for Template ID Slot

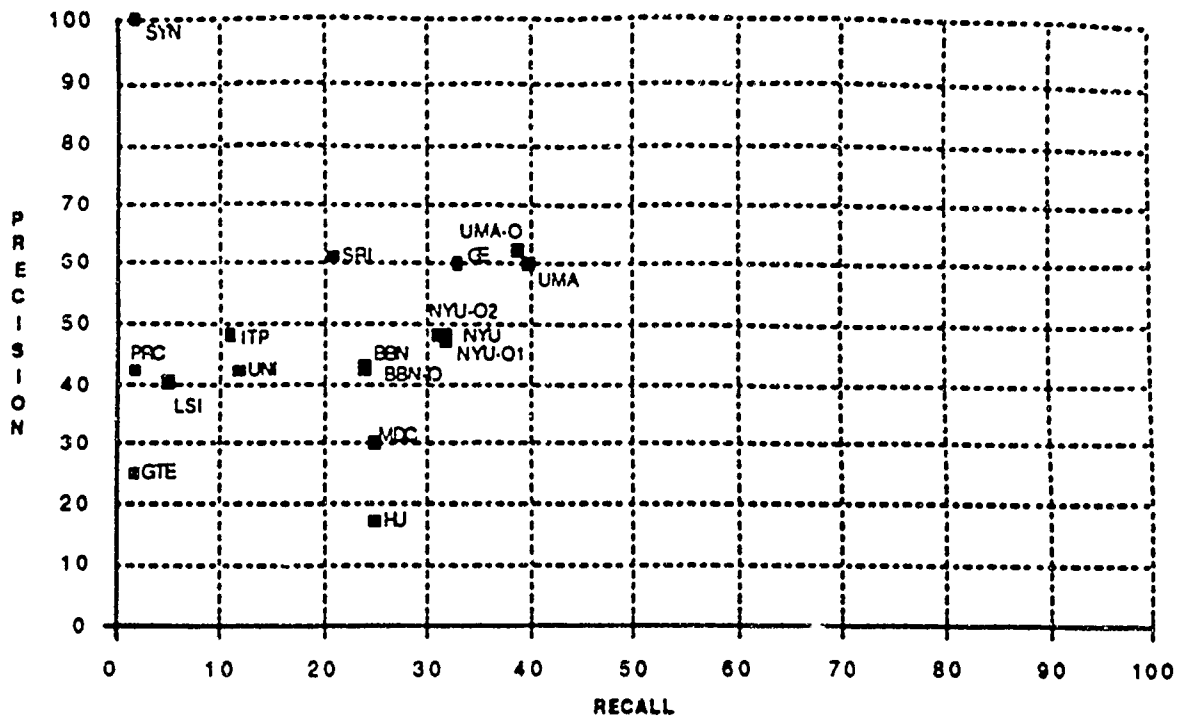


Figure A5. Recall vs Precision for Human Target ID Slot  
 (Some systems did not attempt to fill this slot and therefore do not appear in the figure.)

# MUC-3 EVALUATION METRICS AND LINGUISTIC PHENOMENA TESTS

*Nancy Chinchor, Ph.D.*  
*Science Applications International Corporation*  
*10260 Campus Point Drive, MIS 12*  
*San Diego, CA 92121*  
*(619) 458-2728*  
*chinchor@nosc.mil*

This presentation describes the development of the evaluation metrics and the linguistic phenomena tests for the DARPA-sponsored Third Message Understanding Conference (MUC-3). The systems participating in the conference were evaluated for their performance on a specific data extraction task. The systems represented a variety of approaches to the problem of data extraction many of which include natural language processing. The task for evaluation was based on news reports of potential terrorist activities and the task design is discussed in detail in a separate presentation by Beth Sundheim of the Naval Ocean Systems Center.

The message understanding systems participating in the MUC-3 evaluation produce filled database templates for test messages. The template fills were scored semi-automatically against a human-produced key by software developed especially for MUC-3. The scoring algorithm implemented in that software was designed based on initial consultations among members of the Program Committee, criteria determined during the process of fully specifying the overall evaluation metrics for a dry run of the testing procedure (February 1991), and discussions following the dry run. This presentation describes the evaluation metrics, the rationale behind them, and their utility in the final evaluation.

In addition to these official overall metrics, linguistic phenomena tests were also run for each of the systems. These tests have been designed in two phases, with an analysis of the linguistic phenomena test experiment concerning the validity of the tests completed in the second phase (May 1991). This presentation discusses the development of the linguistic phenomena tests and the results of the linguistic phenomena test experiment.

## EVALUATION METRICS

The evaluation metrics for MUC-3 were based on tallying raw scores for the template slot fills given by the system. The templates were identified by the message identifier and the template number for that message. Templates could be generated in any order for a particular message and the scoring system would map them to the key in a manner which optimized the score. Participants could remap the templates by hand and a history of that remapping would be kept for the official record. The other slots pertaining to the incident contained two types of fills, those that come from a finite specified list of fills and those that are string fills that essentially come from an infinite set or a set with an indeterminable cardinality. These two types of slots are treated differently in the scoring.

Tallies for both kinds of slots were kept as to whether the fills were correct, incorrect, noncommittal, partially correct, spurious, or missing (see Figure 1). A system response was considered correct if it exactly matched the answer key. It was considered incorrect if it did not match the answer key. Possible mismatches caused the semi-automated scoring software to prompt the user for information as to the correctness of the mismatched slot filler. In the case of set fills, the most credit the user could give was partial credit. In the case of string fills, full credit could be given. Noncommittal slot fills were those for which both the answer key and the system response was blank. A system response was spurious if the answer key was blank and the system had filled the slot. The system response was scored as missing if it was blank where the answer key had a fill.

\* \* \* TOTAL SLOT SCORES \* \* \*

SLOT	POS	ACT	COR	PAR	INC	ICR	IPA	SPU	MIS	NON	REC	PRE	OVG	FAL
template-id	118	115	114	0	0	0	0	1	4	39	97	99	1	
incident-date	114	110	90	10	10	31	10	0	4	4	83	86	0	
incident-type	118	114	112	1	1	0	1	0	4	0	95	99	0	0
category	90	109	88	0	0	0	0	21	2	7	98	81	19	14
indiv-perps	106	61	59	0	2	10	0	0	45	50	56	97	0	
org-perps	71	68	58	0	1	15	0	9	12	48	82	85	13	
perp-confidence	71	68	56	1	2	12	1	9	12	48	80	83	13	2
phys-target-ids	59	57	54	3	0	14	3	0	2	77	94	97	0	
phys-target-num	41	41	39	0	2	0	0	0	0	77	95	95	0	
phys-target-types	59	57	52	4	1	11	4	0	2	77	92	95	0	0
human-target-ids	145	131	129	2	0	33	2	2	14	23	90	99	2	
human-target-num	94	88	79	6	2	0	6	1	7	23	87	93	1	
human-target-types	145	131	126	2	3	24	2	2	14	23	88	97	2	0
target-nationality	35	19	17	2	0	3	2	5	16	103	51	95	0	0
instrument-types	25	22	16	1	0	0	0	5	8	88	66	75	23	0
incident-location	118	113	88	24	1	0	1	0	5	0	85	88	0	
phys-effects	41	44	37	3	0	8	3	4	1	89	94	88	9	0
human-effects	56	54	43	2	2	10	2	8	9	81	78	81	15	1
MATCHED ONLY	1464	1402	1257	61	27	171	37	62	119	826	88	92	4	
MATCHED/MISSING	1506	1402	1257	61	27	171	37	62	161	857	85	92	4	
ALL TEMPLATES	1506	1420	1257	61	27	171	37	80	161	861	85	91	6	
SET FILLS ONLY	640	618	547	16	9	68	15	49	68	516	87	90	8	0

Figure 1: Summary Score Report

The four evaluation metrics used in MUC-3 were recall, precision, fallout, and overgeneration. These four metrics were defined in terms of the categories mentioned above (see Figure 2). Recall was the sum of the points for actual attempts divided by the total possible points. The numerator was the sum of the number of correct answers and 0.5 times the number of partially correct answers. The denominator was the number of slot fillers in the answer key. Recall measured the completeness with which the system extracted data. It was a measure of the amount of relevant data the system put in the templates relative to the total amount of data that should have been put in the templates. Recall was calculated for each slot and over all slots. Recall was an important metric in the evaluation of the message understanding systems because of the importance of completeness in the extraction of data.

$$\text{recall} = \frac{\text{correct} + (\text{partial} \times 0.5)}{\text{possible}}$$

where possible = number of required slot fillers in key  
+ number of matched optional values

$$\text{precision} = \frac{\text{correct} + (\text{partial} \times 0.5)}{\text{actual}}$$

where actual = number of slot fillers in response

$$\text{overgeneration} = \frac{\text{spurious}}{\text{actual}}$$

$$\text{fallout} = \frac{\text{incorrect} + \text{spurious}}{\text{possible incorrect}}$$

where possible incorrect = number of possible  
incorrect answers which  
could be given in response

Figure 2: Calculation of Metrics

Precision was a measure of the accuracy of the system's answers. It was calculated by dividing the sum of points for all actual attempts by the total possible points if all actual attempts were correct. The numerator was the sum of the number of the correct answers and 0.5 times the number of partially correct answers. The denominator was the sum of the number of correct, partially correct, incorrect, and spurious answers generated by the system. The number of spurious slot fillers generated by the system affected the overall precision because it was necessary to penalize for systems that overgenerate slot fillers in an attempt to maximize their score. Precision was a measure of the amount of relevant data the system put into the templates relative to the total amount of the data the system put into the templates. It was the tendency of the system to avoid assigning bad fillers as it assigned more good fillers. If overgeneration tends to be proportional to correct generation, then precision is a measure of overgeneration. Precision was calculated for individual slots as well as for all slots. Precision, like recall, was an important metric in the evaluation of message understanding systems because it indicated how well the system performed on the fillers it actually generated.

Fallout measured the tendency of a system to assign more incorrect fillers as the number of potential incorrect fillers increased. Fallout was calculated by dividing the number of incorrectly given fillers, i.e., the number of incorrect and spurious fillers given for the slot, by the number of possible incorrect fillers. Because of its dependence on the cardinality of the set of fillers available for a slot, fallout could only be measured for those slots in the MUC-3 templates which were filled from finite sets. No global fallout score could be calculated due to the fact that not all slots were filled from finite sets. A partial global score was calculated for all slots that were filled from finite sets. Fallout was a measure of overgeneration if overgeneration was proportional to the number of opportunities to overgenerate.

Fallout was only somewhat important in the evaluation because it was an attempt to measure false positives for a task which did not lend itself well to this sort of measurement due to its open-ended nature.

The "overgeneration" measure was a measure of the amount of spurious fillers assigned in relation to the total assigned. It was calculated by dividing the number of spurious slot fillers by the number of slot fillers given. Overgeneration was calculated for individual slots as well as all slots. The overgeneration measure was an important part of the final analysis of the systems because it isolated a key aspect of precision and shed further light on the trade-off between overgeneration and recall.

In the discussions following the dry run, it was decided that four summary rows of metrics would be included in the score report. The four rows would be calculated based on different ways of tallying the raw scores that go into the calculation of the metrics. Three of the rows treated spurious and missing templates with varying levels of strictness depending on whether the spurious and missing slot fills in those templates were scored or whether just the template id slot was penalized for the error.

The first row, called "matched only," scored spurious and missing templates only in the template id slot. The second row, called "matched/missing," scored missing slot fills as missing for all the affected slots. This row was used as the official score for the final MUC-3 test run and was the only summary row in the dry run. The third row, called "all templates," scored spurious and missing slot fills in all the slot rows affected. This row represented the strictest scores.

A fourth and final summary row was also given for "set fills only." This row represented how the systems did for the slots whose fills came from finite sets. A global fallout score was calculated from the totals in this summary row in an attempt to get a "false alarm" rate for the systems. However, this global fallout score is not representative of the task because of the considerable number of important string fill slots not included. The effort to measure the false alarm rate has been hampered because the number of possible incorrect cannot be determined for slots whose fills come from a potentially infinite set.

The variety of summary scores was provided both to determine the appropriate way to score the spurious and missing templates and to provide an indication of how well systems would do for applications with varying requirements. Applications may have differing tolerances for the amount of data missing from the generated database and the amount of spurious data entered into the database. An analysis of the final results of MUC-3 was done by plotting recall versus precision, overgeneration versus recall, and overgeneration versus precision. The results indicate that the "all templates" score reflects the expected effect of overgeneration on the plots.

A more extensive report on the evaluation metrics entitled "MUC-3 Evaluation Metrics" appears in the proceedings of the Third Message Understanding Conference (MUC-3) published by DARPA.

## LINGUISTIC PHENOMENA TESTING

Linguistic phenomena testing can supplement the overall evaluation of data extraction systems because the tests measure performance on a set of characteristics of the input critical to the output. For the dry run held in February, there were three main linguistic phenomena tests devised representing increasing frequency



of the phenomena in the test messages. The first tested for processing of negatives in filling two slots in the template fill task. The results of the dry run and the information from participants strongly suggested that negation should be tested more broadly in terms of slots filled and in terms of linguistic constructions considered, if possible. The second linguistic phenomena test was for the processing of conjunctions in filling two slots. The dry run showed that this was a useful test with regards to the task and the systems. Conjunction was more frequent than negation and was frequent enough to be representative. The third linguistic phenomena test was constructed around the terrorist incident type verb forms (active versus passive) and the kind of clauses in which they appeared (main versus subordinate). The intent was to determine if the systems performed differently on slots that required processing of active versus passive verb forms and main versus subordinate clauses. The preliminary results showed that the test results for this third test coincided with overall system performance and did not necessarily reveal new information about linguistic processing capabilities. This test was also the most difficult of the three to devise because of the process required to determine which slots reflected the linguistic processing we were testing. We limited the scoring to slot fillers appearing in the same sentence as the verb and not appearing anywhere else in the message.

In general, it was decided as a result of the dry run that the linguistic phenomena tests would only begin to be meaningful once the systems were performing at a higher level. This threshold of performance may be reached in MUC-3 or may not be reached until MUC-4. However, it was also decided that the development of linguistic phenomena tests was to proceed in parallel with the development of the systems. The validity of the phenomena testing became the focus of an experiment run in the second and final phase of MUC-3 in May 1991.

The experiment was designed to determine if linguistic phenomena could be isolated. All of the phenomena tests for MUC-3 were scored using the MUC-3 scoring system. The experiment was run for the phenomenon of apposition of noun phrases, for example, "David Lecky, Director of the Columbus school." One or more appositives containing information critical to the template fill task appeared in approximately 60 sentences in the test corpus. This frequency was higher than the required frequency of 20 indicated by the dry run.

The appositives were scored in several ways to determine the validity of the testing. The scores on slots filled from phrases and sentences containing appositive constructions were different from the overall scores for the systems indicating that it was possible that the phenomenon was being isolated. The scores for slots filled from phrases and sentences were similar indicating that future phenomena tests could be designed based on information in the entire sentence containing the phenomenon. The appositives were subjectively divided into subsets based on complexity prior to testing. The systems scored consistently higher on the simpler set than they did on the more complex set. This result provided more confidence that the phenomenon was being isolated.

The appositives were also divided according to whether they were postposed or preposed. A postposed appositive from the test corpus was "Jose Parada Grandy, the Bolivian Police Chief." "Rede Golobo journalist Carlos Marcelo" is a preposed appositive. If systems scored differently on these types of appositives, we would have more confidence that we were testing appositives. Neither type would necessarily be easier because, although postposed appositives are more prototypical, preposed appositives could be processed as modifiers. The systems did score differently on the two types but did not score consistently higher on either type.

There was one part of the testing that did not follow the hypotheses put forth before the testing. That test was constructed to compare results for "minimal pairs." A set of messages was constructed to be exactly like the original test messages except that in place of the appositives, simple sentences asserting the equivalence of the appositioned noun phrases were introduced. The test was voluntary because it required an additional run of the data extraction systems as well as scoring. Two sites volunteered out of the fifteen participating. It was hypothesized that their scores would be higher on the modified messages without the appositives. However, this was not the case. Instead, the "simple" sentences introduced a complexity not considered. The use of the copula in the sentences and the reference resolution required interfered with the strategies being used by the systems to obtain the slot fills. Essentially, both systems ignored much of the information in the added sentences. The appositives were more direct conveyors of this information. However, this result also supports the ability to isolate the phenomenon because there was a difference when the appositives in the messages were taken out. The results of the entire experiment indicate that phenomena can be isolated and that linguistic phenomena tests are valid when carefully designed.

A more extensive report on the linguistic phenomena test experiment entitled "MUC-3 Linguistic Phenomena Test Experiment" appears in the proceedings of the Third Message Understanding Conference (MUC-3) published by DARPA.

#### **BRIEF DESCRIPTION OF RESEARCH ACTIVITIES:**

My research has recently focused on designing the metrics and phenomena tests for MUC-3 and facilitating the implementation of the semi-automated scoring system used in the official scoring of participating systems. The MUC-3 Program Committee and the participants have engaged in an interchange of ideas about these topics. My role has been to be a part of this interchange and resolve the issues necessary to finalize the testing. In addition to MUC-3, my other research activities include algorithm development and implementation for a variety of applications of artificial intelligence.

# A Developing Methodology for the Evaluation of Spoken Language Systems

Madeline Bates and Sean Boisen  
BBN Systems and Technologies  
10 Moulton Street  
Cambridge, MA 02138  
bates@bbn.com

## Abstract

Work in speech recognition (SR) has a history of evaluation methodologies that permit comparison among various systems, but until recently no methodology existed for either developers of natural language (NL) interfaces or researchers in speech understanding (SU) to evaluate and compare the systems they developed.

Recently considerable progress has been made by a number of groups involved in the DARPA Spoken Language Systems (SLS) program to agree on a methodology for comparative evaluation of SLS systems, and that methodology has been used in practice several times. This evaluation is probably the only NL evaluation other than MUC to have been developed and used by a group of researchers at different sites, although an excellent workshop was held to study some of these problems [Palmer, 1988].

This paper gives an overview of the process that was followed in creating a meaningful evaluation mechanism, describes the current mechanism, and presents some problems and directions for future development. The development corpora and all material related to the evaluation will be publically available from NIST.

## 1. A Brief History

The goal of the DARPA Spoken Language Systems program is to further research and demonstrate the potential utility of speech understanding. Currently, four major sites (BBN, CMU, MIT, and SRI) are developing complete SLS systems, and another site (UNISYS) is integrating its NL component with MIT's speech system. Representatives from these and other organizations meet regularly to discuss program goals and to evaluate progress.

This DARPA SLS community formed a committee on evaluation<sup>1</sup>, chaired by Dave Pallett of the National Institute of Standards and Technology. The committee was to develop a methodology for data collection, training data dissemination, and testing for SLS systems under development.

The first community-wide evaluation using the first version of methodology developed by this committee took place in June, 1990, and the second in February, 1991. They are reported in [2] and [3] respectively. Additional information about the methodology can be found in [Boisen, 1989] and [Ramshaw, 1990].

The emphasis of the committee's work has been on automatic evaluation of queries to an air travel information system (ATIS). Why ATIS? Because a database-oriented task seemed

---

<sup>1</sup> The primary members of the committee are: Lyn Bates (BBN), Debbie Dahl (UNISYS), Bill Fisher (NIST), Lynette Hirschman (MIT), Bob Moore (SRI), and Rich Stern (CMU). Many other people contributed to the work of the committee and its subcommittees.

most tractable, and air travel is an application that is easy for everyone to understand. Why "automatic" evaluation? Because it was the most objective, least expensive practical solution we could come up with.

## 2. Some Issues

Systems for NL understanding, or speech understanding are inherently much more difficult to evaluate than SR systems. This is because the output of speech recognition is easy to specify - it is a character string containing the words that were spoken as input to the system - and it is trivially easy to determine the "right" answer and to compare it to the output of a particular SR system. Each of these steps,

1. specifying the form that output should take,
2. determining the right output for particular input, and
3. comparing the right answer to the output of a particular system,

is very problematic for NL and SU systems.

## 3. The Goal

The goal of the work was to produce a well-defined, meaningful evaluation methodology (implemented using an automatic evaluation system) which will both permit meaningful comparisons between different systems and also allow us to track the improvement in a single NL or SU system over time. The systems are assumed to be front ends to an interactive application (database inquiry) in a particular domain (ATIS).

The intent is to evaluate specifically *NL understanding* capabilities, not other aspects of a system, such as the user interface, or the utility (or speed) of performing a particular task with a system that includes a NL component.

## 4. The Evaluation Framework

The methodology that was developed is very similar in style to that which has been used for speech recognition systems for several years. It is:

1. Collect a set of data as large as feasible, under conditions as realistic as possible.
2. Reserve some of that corpus as a test set, and distribute the rest as a training set.
3. Develop agreement on meanings and answers for the items in the test set, and an automatic comparison program to compare those "right" answers with the answers produced by various systems.
4. Send the test set to the sites, where they will be processed unseen and without modifications to the system. The answers are then returned and run through the evaluation procedure, and the results reported.

Figure 1 illustrates the relationship between an SLS system and the evaluation system.

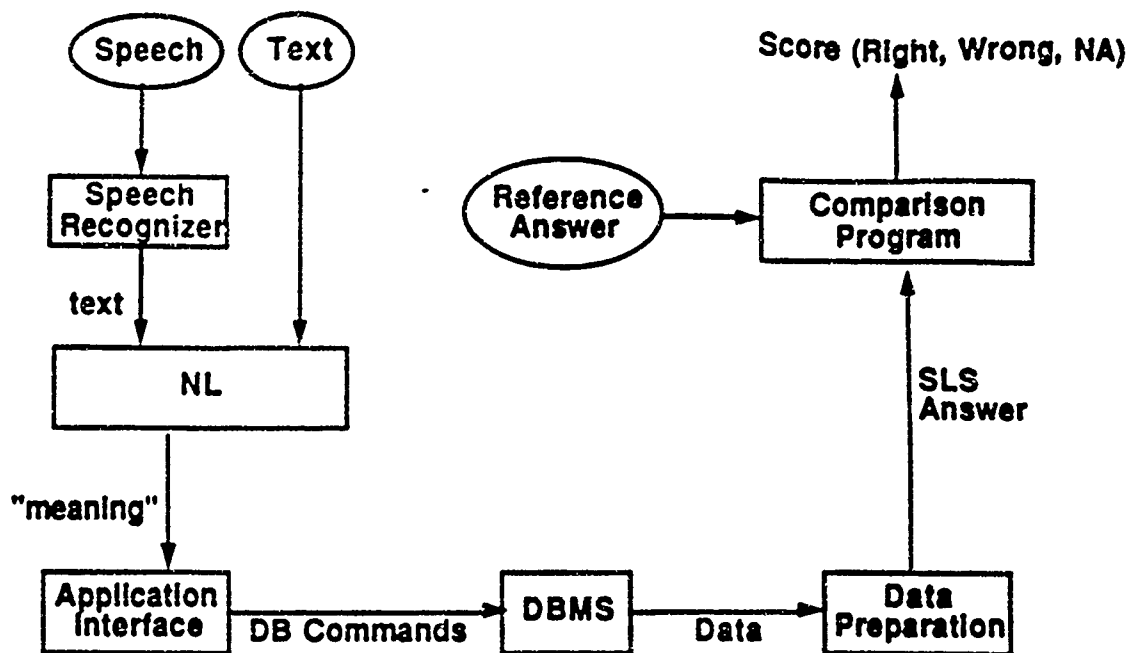


Figure 1: The Evaluation Framework

#### 4.1 Collecting Data

A method of data collection called "Wizard scenarios" was used to collect raw data (speech and transcribed text). This system is described in [Hemphill 1990]. It resulted in the collection of a number of human-machine dialogues.

It became clear that the language obtained in Wizard scenarios is very strongly influenced by the particular task, the domain and database being used, and the amount and form of data returned to the user.

#### 4.2 Classifying Data

One of the first things to become clear was that not all of the collected data was suitable as test data, because not all of the queries posed by the subjects could be answered by the wizard, and some of those that were answered were clearly beyond any reasonable goal for this generation of NL systems. Thus it was desirable that the training data be marked to indicate which queries one might reasonably expect to find in the test set.

The notion emerged of having a number of classes of data, so that we could begin with a core (Class A) which was clearly definable and possible to evaluate automatically, and, as we came to understand the evaluation process better, which could be extended to other types of queries (Classes B, C, D etc.).

Several possible classification systems were presented and discussed in great detail. Two have been agreed on at this time, Class A, which consists basically of independent utterances with agreed upon meanings, and Class D1, which consists of very short dialogue fragments.

### 4.3 Agreeing on Meaning

Agreeing on the meaning of queries has been one of the hardest tasks for the committee. The issues are often subtle, and interact with the structure and content of the database in sometimes unexpected ways.

As an example of the problem, consider a request to "list the direct flights from Boston to Dallas that serve meals". It seems straightforward, but should this include flights that might in Chicago without making a connection there? Should it include flights that serve a snack, since a snack is not considered by some people to be a full meal?

Without some common agreement, many systems would produce very different answers for the same questions, all of them equally right according to the systems' own definitions of the terms, but not amenable to automatic evaluation. It was necessary to agree on the meaning of terms such as "mid-day", "meals", "the fare of a flight", and several dozen other things, but this agreement was achieved and is documented.

### 4.4 Developing Reference Answers

It is not enough to agree on meaning of queries in the chosen domain. It is also necessary to develop a common understanding of what is to be produced as the answer, or part of the answer, to a question.

For example, if a user asks "What is the departure time of the earliest flight from San Francisco to Atlanta?", one system might reply with a single time and another might reply with that time plus additional columns containing the carrier and flight number, a third system might also include the arrival time and the origin and destination airports. None of these answers could be said to be wrong, although one might argue about the advantages and disadvantages of terseness and verbosity.

It was agreed that, for the sake of automatic evaluation, a canonical reference answer (the minimum "right" answer) should be developed for each evaluable query in the training set, and that the reference answer should be that answer retrieved by a reference SQL expression. That is, the right answer was defined by the expression which produces the answer from the database, as well as the answer retrieved. This ensures A) that it is possible to retrieve the canonical answer via SQL, B) that even if the answer is empty or otherwise limited in content, it is possible for system developers to understand what was expected by looking at the SQL, and C) the reference answer contains the least amount of information needed to determine that the system produced the right answer.

What should be produced for an answer is determined both by domain-independent linguistic principles [Bolsen, 1989] and domain-specific stipulation (Appendix A). The language used to express the answers is defined in Appendix B.

### 4.5 Developing a Comparator

A final necessary component is, of course, a program to compare the reference answers to those produced by various systems. One was written in C by NIST.; anyone interested in obtaining the code for these comparators should contact Bill Fisher at NIST.

The task of answer comparison is complicated substantially by the fact that the canonical answer is intentionally minimal, but the answer supplied by a system may contain extra

information. Some intelligence is needed to determine when two answers match (i.e. simple identity tests won't work).

#### 4.6 Presenting Results

Expressing results can be almost as complicated as obtaining them. Originally it was thought that a simple "X percent correct" measure would be sufficient, however it became clear that there was a significant difference between giving a wrong answer and giving no answer at all, so the results are now presented as: Number right, Number wrong, Number not answered, Weighted error percentage (weighted so that wrong answers are twice as bad as no answer at all), and Score (100 - weighted error).

### 5. Strengths of the Methodology

It forces advance agreement on the meaning of critical terms and on at least minimal information to be included in the answer.

It is objective, to the extent that a method for selecting testable queries can be defined, and to the extent that the agreements mentioned above can be reached.

It requires less human effort (primarily in the creating of canonical examples and answers) than non-automatic, more subjective evaluation. It is thus better suited to large test sets.

It can be easily extended.

### 6. Weaknesses of the Methodology

It does not distinguish between merely acceptable answers and very good answers.

It does not distinguish between some cases, and may thus give undue credit to a system that "over answers".

It cannot tell if a system gets the right answer for the wrong reason.

It does not adequately measure the handling of some phenomena, such as extended dialogues.

### 7. Future Issues

The hottest topic currently facing the SLS community with respect to evaluation is what to do about dialogues. Many of the natural tasks one might do with a database interface involve extended problem-solving dialogues, but no methodology exists for evaluating the capabilities of systems attempting to engage in dialogue with users. Several suggestions have been made ([Hirschman, 1990] and [Bates, 1991]) and will be discussed in depth.

## References

1. *Proceedings of the Speech and Natural Language Workshop, October 1989*, Morgan Kaufmann Publishers, Inc.
  2. *Proceedings of the Speech and Natural Language Workshop, June 1990*, Morgan Kaufmann Publishers, Inc.
  3. *Proceedings of the Speech and Natural Language Workshop, February, 1991*, Morgan Kaufmann Publishers, Inc., (to appear).
- Bates, M., Boisen, S., and Makhoul, J., *Developing an Evaluation Methodology for Spoken Language Systems*, in Proceedings of the Speech and Natural Language Workshop, June 1990, Morgan Kaufmann Publishers, Inc.
- Bates, M. and Ayuso, D., *A Proposal for Incremental Dialogue Evaluation*, in Proceedings of the Speech and Natural Language Workshop, February, 1991, Morgan Kaufmann Publishers, Inc.,
- Black, E., et al., *A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars*, in Proceedings of the Speech and Natural Language Workshop, February, 1991, Morgan Kaufmann Publishers, Inc.,
- Boisen, Sean , Lance A. Ramshaw, Damaris Ayuso, and Madeleine Bates, *A Proposal for SLS Evaluation*, in Proceedings of the DARPA Speech and Natural Language Workshop, October 1989, Morgan Kaufmann Publishers, Inc.
- Hemphill, Charles, *T1 Implementation of Corpus Collection*, in Proceedings of the Speech and Natural Language Workshop, June 1990, Morgan Kaufmann Publishers, Inc., 1990.
- Hirschman, Lynette, et al, *Beyond Class A: A Proposal for Automatic Evaluation of Discourse*, in Proceedings of the Speech and Natural Language Workshop, June 1990, Morgan Kaufmann Publishers, Inc., 1990.
- Pallet, David S., William M. Fisher, Jonathan G. Fiscus, *Tools for the Analysis of Benchmark Speech Recognition Tests*, *Proceedings of ICASSP 1990*, p. 97.
- Pallett, David S., et al, *DARPA ATIS Test Results June 1990*, in Proceedings of the Speech and Natural Language Workshop, June 1990, Morgan Kaufmann Publishers, Inc., 1990.
- Pallett, D.S., et al., *NIST DARPA SLS February 1991 Benchmark Test Results Handout*, unpublished paper presented at the Speech and Natural Language Workshop, February, 1991.
- Palmer, M., T. Finin and S. Walter, *Workshop on the Evaluation of Natural Language Processing Systems*. RADC-TR-89-302. RADC Technical Report on the Workshop held in Wayne, PA, in December 1988.
- Ramshaw, Lance A and Sean Boisen, *An SLS Answer Comparator*. SLS Note No. 7, BBN Systems and Technologies Corporation, Cambridge, MA, May 25, 1990.



## Appendix A: Interpreting ATIS queries Relative to the Database

### Basics:

A large class of tables in the database have entries that can be taken as defining things that can be asked for in a query. In the answer, each of these things will be identified by giving a value of the primary key of its table. These tables are:

Name of Table	English Term(s)	Primary Key
aircraft	aircraft, equipment	aircraft_code
airline	airline	airline_code
airport	airport	airport_code
city	city	city_code
compound_class	service classes, e.g. "coach", etc.	fare_class
day	names of the days of the week	day_code
fare	fare	fare_code
flight	flight	flight_code
food_service	meals	meal_code, meal_number, meal_class
ground_service	ground transportation	city_code, airport_code, transport_code
month	months	month_number
restriction	restrictions	restrict_code
state	names of states	state_code
time_zone	time zones	time_zone_code, time_zone_name
transport	transport code	transport_code

### Special meanings:

In this arena, certain English expressions have special meanings, particularly in terms of the database distributed by TI in the spring of 1990. Here are the ones we have agreed on: (In the following, "A.B" refers to field B of table A.)

#### 1. Flights.

A flight "between X and Y" means a flight "from X to Y".

In an expression of the form "flight number N", where N is a number, N will always be interpreted as referring to the flight number (flight.flight\_number). "Flight code N" will unambiguously refer to flight.flight\_code. "Flight N" will refer to flight.flight\_number if N is in the range  $0 \leq N \leq 9999$  but to flight.flight\_code if  $N \geq 10000$ .

A "one-way" flight is a flight with a fare whose one-way cost is non-empty.

Principle: If an attribute "X" of a fare, such as "one-way" or "coach", is used as a modifier of a flight, it will be interpreted as "a flight with an X fare".

#### 2. Fare (classes).

A "one-way" fare is one with a non-empty one-way cost.

References to the "cheapest fare" are ambiguous, and may be interpreted as meaning either the cheapest one-way fare, the cheapest round-trip fare, the cheapest of either, or the cheapest of both.

A "coach" fare is one whose `compound_class.class_type = "COACH"`. Similarly, the fare modifiers "first class", "business class", and "thrift class" refer to values of the `compound_class.class_type` field.

A reference to ranking of fares, e.g. "fares that are Y class or better", will be interpreted as a reference to the rank of the associated base fare (`class_of_service.rank`). A "discounted fare" is one whose `compound_class.discounted = "YES"`.

An "excursion fare" is one with a restriction code (`fare.restrict_code`) that contains the string "AP", "EX", or "VU".

A "family fare" is the same thing as an "excursion fare".

A "special fare" is one with a non-null restriction code (`fare.restrict_code`).

### 3. Time.

The normal answer to otherwise unmodified "when" queries will be a time of day, not a date or a duration.

The answer to queries like "On what days does flight X fly" will be a list of `day.day_code` fields, not a `flight_days` string.

Queries that refer to a time earlier than 1300 hours without specifying "a.m." or "p.m." are ambiguous and may be interpreted as either.

### 4. Units.

All units will be the same as those implicit in the database (e.g. feet for `aircraft.wing_span`, but miles for `aircraft.range_miles`, durations in minutes).

### 5. Meals.

For purposes of determining flights "with meals/meal service", snacks will count as a meal.

"list the types of meal" should produce one tuple per meal, not a single `meal_code` string.

### 6. "With" clauses.

"with"-modification clauses: "show me all the flights from X to Y with their fares" will require the identification of both flights and their fares (so if there are 2 flights, each with three fares, the answer will have 6 tuples, each with at least the `flight_code` and `fare_code`). In general, queries asking for information from two or more separate tables in the database will require the logical union of fields that would identify each table entry separately.

7. The "itinerary" of a flight refers to the set of all non-stop legs of that flight. When an "itinerary" is asked for, each leg of the flight will be identified by the origin and destination cities for that leg, e.g. (("BOS" "ATL") ("ATL" "DFW")).

8. "what kind of X is Y" queries, where Y is clearly a kind of X, will be interpreted as equivalent to "what does Y mean?", where Y is a primary key value for the table referred to by X (see 10 below).

9. "class".

References to classes of service will be taken as referring to the contents of the compound\_class table (not the class\_of\_service table).

Queries about (unmodified) "class X", e.g. "What is class X?", will be interpreted as referring to the set of compound\_class.fare\_class entries for which "X" is the fare\_class, not the base\_class, e.g. '({"X"})', not '({"XA"} {"XB"})'.

The expression "(fare) classes for flight X" refers ambiguously to either the mostly 1-character class codes that are stored in flight\_class.fare\_class (sometimes called "booking codes"), or to the largely multi-character class codes that are stored in fare.fare\_class as attributes of fares that are associated with flight X via the flight\_fare table (sometimes called "fare bases"). In either case, the answer should have the class codes in separate fields, not packed together as they are in flight.class\_string.

10. Requests for the "meaning" of something will only be interpretable if that thing is a code with a canned definition in the database. Here are the things so defined, with the fields containing their decoding:

Table	Key Field	Decoding Field
aircraft	aircraft_code	aircraft_type
airline	airline_code	airline_name
airport	airport_code	airport_name
city	city_code	city_name
code_description	code	description
column_table	heading	column_description
day	day_code	day_name
food_service	meal_code	meal_description
Interval	period	begin_time, end_time
month	month_number	month_name
state	state_code	state_name
time_zone	time_zone_code	time_zone_name
transport	transport_code	transport_description

11. A request for a flight's stops will be interpreted as asking for the final stop in addition to intermediate stops.

12. Queries that are literally yes-or-no questions may be answered by either a boolean value ("YES/TRUE/NO/FALSE") or a relation, expressed as a table. Reference (REF) answers to such questions will be recorded as either a null or a non-null table. If the reference answer is a null table, then either "NO/FALSE" or a null table will count as correct; if the reference answer is a non-null table, then either "YES/TRUE" or a table that matches the REF table will be counted as correct. In other words, if a table is given as an answer, it must match the REF table to be counted correct.

13. A city and an airport will be considered "near" (or "nearby") each other iff the city is served by the airport, and two cities will be considered "near" (or "nearby") each other iff there is an airport that serves them both.

14. When it is clear that an airline is being referred to, the term "American" by itself will be taken as unambiguously referring to American Airlines.

15. Vague queries of the form "Give me information about X" or "Describe X" or "What is X" will be interpreted as equivalent to "List X".

16. References to "the city" or "downtown" are ambiguous, and may be interpreted as referring to any city that seems reasonable.

17. When a query refers to an aircraft type with a descriptive phrase, such as "BOEING 767" or "TYPE 767 -- BY BOEING", the reference is ambiguous: it may be taken to be the set of entries in the "aircraft" table whose "aircraft\_type" field (the second in the current table) matches the descriptive phrase well, such as:

767, "BOEING 767 (ALL SERIES)", ...

763, "BOEING 767-300/300ER", ...

or it may be taken to mean just the entry with the matching "aircraft\_code" value (the first field in the current table).

18. Utterances whose answers require arithmetic computation are not now considered to be interpretable.

19. Cases like "the prices of flights, first class, from X to Y", in which the attachment of a modifier that can apply to either prices or flights is unclear, should be (ambiguously) interpreted both ways, as both "the first-class prices on flights from X to Y" and "the prices on first-class flights from X to Y". More generally, if structural ambiguities like this could result in different (SQL) interpretations, they must be treated as ambiguous.

## Appendix B: Common Answer Specification (CAS) Syntax

### BASIC SYNTAX IN BNF:

```
<answer> ::= <cas1> | (<cas1> OR <answer>)
<cas1> ::= <scalar-value> | <relation> | NO_ANSWER | no_answer
<scalar-value> ::= <boolean-value> | <number-value> | <string>
<boolean-value> ::= YES | yes | TRUE | true | NO | no | FALSE | false
<number-value> ::= <integer> | <real-number>
<integer> ::= (<sign> <digit>+)
<sign> ::= + | -
<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<real-number> ::= <sign> <digit>+ . <digit>* | <digit>+ . <digit>*
<string> ::= <char_except_whitespace>+ | "<char>*"
<relation> ::= (<tuple>*)
<tuple> ::= (<value>+)
<value> ::= <scalar-value> | NIL
```

Standard BNF notation has been extended to include two other common devices. "<A>+" means "one or more A's" and "<A>\*" means "zero or more A's".

The above formulation does not define <char\_except\_whitespace> and <char>. All of the standard ASCII characters count as members of <char>, and all but "white space" count as <char\_except\_whitespace>. Following ANSI "C", blanks, horizontal and vertical tabs, newlines, formfeeds, and comments are, collectively, "white space".

The only change in the syntax of CAS itself from the previous version is that now a string may be represented as either a sequence of characters not containing white space or as a sequence of any characters enclosed in quotation marks. Note that only non-exponential real numbers are allowed, and that empty tuples are not allowed (but empty relations are).

#### ADDITIONAL SYNTACTIC CONSTRAINTS

The syntactic classes <boolean-value>, <string>, and <number-value> define the types "boolean", "string", and "number", respectively. All the tuples in a relation must have the same number of values, and those values must be of the same respective types (boolean, string, or number).

If a token could represent either a string or a number, it will be taken to be a number; if it could represent either a string or a boolean, it will be taken to be a boolean. Interpretation as a string may be forced by enclosing a token in quotation marks.

In a tuple, NIL as the representation of missing data is allowed as a special case for any value, so a legal answer indicating the costs of ground transportation in Boston would be

```
(( "L" 5.00 ) ( "R" nil ) ( "A" nil ) ( "R" nil ))
```

#### ELEMENTARY RULES FOR CAS COMPARISONS

String comparison is case-sensitive, but the distinguished values (YES, NO, TRUE, FALSE, NO\_ANSWER, and NIL) may be written in either upper or lower case.

Each indexical position for a value in a tuple (say, the *i*th) is assumed to represent the same field or variable in all the tuples in a given relation.

Answer relations must be derived from the existing relations in the database, either by subsetting and combining relations or by operations like averaging, summation, etc.

In matching an hypothesized (HYP) CAS form with a reference (REF) one, the order of values in the tuples is not important; nor is the order of tuples in a relation, nor the order of alternatives in a CAS form using "OR". The scoring algorithm will use the re-ordering that maximizes the indicated score. Extra values in a tuple are not counted as errors, but distinct extra tuples in a relation are. A tuple is not distinct if its values for the fields specified by the REF CAS are the same as another tuple in the relation; these duplicate tuples are ignored. CAS forms that include alternate CAS's connected with "OR" are intended to allow a single HYP form to match any one of several REF CAS forms. If the HYP CAS form contains alternates, the score is undefined.

In comparing two real number values, a tolerance will be allowed, the default is plus or minus .01%. No tolerance is allowed in the comparison of integers. In comparing two strings, initial and final sub-strings of white space are ignored. In comparing boolean values, "TRUE" and "YES" are equivalent, as are "FALSE" and "NO".

# Multi-site Natural Language Processing Evaluation: MUC and ATIS\*

Deborah Dahl  
*Unisys Defense Systems, Inc.*

Doug Appelt  
*SRI International*

Carl Weir  
*Unisys Defense Systems, Inc.*

July, 1991

## 1 Introduction

Currently there are two major multi-site efforts in progress which share the goal of providing quantitative evaluations of natural language processing. The *Air Travel Information System (ATIS)* ([4]) has been developed to evaluate speech, natural language and spoken language processing. The various template generation (data extraction) tasks that have formed the basis for a series of *Message Understanding Conferences (MUCs)* ([5]) have been designed to evaluate text understanding systems. The MUC and ATIS efforts have both been designed to compare the performance of multiple systems on a common blackbox task. In both efforts the task of developing evaluation procedures has been extremely valuable, providing new insights into technical issues and revealing unanticipated merits and pitfalls in the evaluation enterprise. It is safe to say that developing evaluation procedures has required more effort than originally had been anticipated by MUC and ATIS participants, and that many unexpected issues have arisen.

Although the specifics of the MUC and ATIS efforts differ, both have required their participants to deal with similar issues in the areas of task and procedure definition, data specification, and scoring. In this paper we describe similarities and differences in the solutions that have been arrived at for such issues in the two efforts. We also describe the benefits which have come from these efforts, and point out aspects of natural language processing performance which are not yet being measured. Our goal at one level is to document the issues that arise in defining large-scale, multi-site evaluations. At a more global level we hope to provide insights into general issues of evaluation based on the experience that we have gained by participating in the MUC and ATIS tasks.

---

\*This work was supported by DARPA contract N000014-89-C0171 to Unisys Corporation, administered by the Office of Naval Research, by DARPA contract MDA-903-89-C-0041 to Unisys Corporation, administered by the Meridian Corporation, by Independent Research and Development funding from Unisys Defense Systems, and by DARPA contract N00014-90-C-0220 to SRI. We thank Dave Pallett of the National Institute of Standards and Technology for his helpful comments on a draft of this paper.

## 2 Goals of Evaluation

In a blackbox evaluation a dual perspective on the goals of the evaluation can arise. Specifically, is the intent of the evaluation to use performance on a task as a tool to evaluate natural language processing or is the intent of the evaluation to measure how well the task can be performed using any techniques at all?

Once a specific task is defined, the question naturally arises of how well that task could be performed using techniques other than natural language. No task that we can use for evaluation is likely to absolutely require natural language processing—there is always the possibility that some other techniques will do at least part of the job. If techniques other than natural language processing can be used to perform the task then the evaluation can be seen as an evaluation of natural language processing as a tool for solving a particular problem versus other possible tools. This is quite a different goal than that of evaluating natural language processing. It can be a valuable goal, but it is important to be clear what the goal is.

The goals of the evaluation also determine what it means if other techniques can solve the problem well. (This assumes that we can decide what is and is not a “natural language processing technique”, which can be controversial.) If the goal is to evaluate natural language processing, then the fact that other techniques work well might mean that the task is not a good one for evaluating natural language processing. On the other hand, if the goal is to perform the task well, the fact that other techniques work well might mean that natural language is not a good tool for this particular problem.

## 3 Tasks

The ATIS and MUC efforts both involve blackbox evaluation measures ([3]). For multi-site system comparisons, this approach to evaluation is currently the only practical choice, since it permits measurements of performance for systems with very different architectures by making it easier (although by no means trivial) to obtain the consensus of multiple sites on performance metrics, and by simplifying the development of automatic scoring software.

### 3.1 The ATIS Task

The task used in the ATIS evaluation is database queries to a relational database of information on air travel planning—for example, information about flight schedules, fares and ground transportation in ten cities. A typical query is shown in Figure 1.

A standard database is supplied to all sites participating in the evaluation. Scoring is based on the answer returned from the database. The data has been collected using a Wizard of Os paradigm, although some data being collected currently has much of the speech and language processing performed by systems, rather than wizards. The data is used for evaluation of speech recognition as well as spoken and written language understanding, although our focus in this paper is on language understanding.

User: please list all the flights between baltimore and atlanta on tuesdays between four in the afternoon and nine in the evening .

Very well. I will ask the expert to do that.

Displaying:

Flight(s):

- departing between 4 pm and 9 pm
- from Baltimore
- to Atlanta
- on Tuesday

3 records found.

FLT_CODE	FLY_DAY	FRM	TO	DEPT	ARRV	AL	FLT#	CLASSES	ZQP	MEAL	STOP	DC	DURA
107166	1234567	BWI	ATL	1600	1750	DL	1055	FYBMQ	M80	S/	0	N	110
107167	1234567	BWI	ATL	1659	1854	EA	131	FYBQK	D9S	D	0	N	115
107168	1234567	BWI	ATL	1945	2128	DL	469	FYBMQ	757	□	0	N	103

Figure 1: Example ATIS query and Unisys system response

### 3.2 The MUC Task

The MUC evaluation effort began in 1987 when DARPA sponsored a message understanding conference directed by Beth Sundheim at NOSC in San Diego. Participating research groups at this first conference were required to report on the performance of their text understanding software in processing military message traffic of a certain type. The conference was successful in bringing research groups together to work on a common domain, but it was clear that a common application and scoring methodology were needed before any such evaluation effort could produce consistent, cross-system performance measures.

Darpa consequently sponsored a second message understanding conference (MUCK-2) in June, 1989. For this conference, a blackbox template (database record) generation task was defined. A message domain similar to the one used in the first conference was used, and guidelines for scoring template generation performance were established. The results reported by participating research groups at this second conference provided a concrete view of the capabilities of current text understanding technology.

A third message understanding conference, MUC-3, took place in June, 1991. For this conference, the same type of template generation task was used that was introduced in the second conference. The message domain, however, was changed to newspaper articles and transcribed radio broadcasts and speeches. A portion of a typical MUC-3 message and its corresponding template fill are shown in Figure 2.

For MUC-3, Darpa funded the development of a scoring program so that the evaluation of template generation performance could be automated to the maximum extent possible, thereby reducing the



BOGOTA, 7 JUL 89 (EFE) -- [TEXT] COLOMBIAN OFFICIALS REPORT THAT GUERRILLAS PRESUMED TO BE MEMBERS OF THE PRO-CASTRO ARMY OF NATIONAL LIBERATION (ELN) TODAY ONCE AGAIN DYNAMITED THE CANO LIMON-COVENAS OIL PIPELINE, COLOMBIA'S MAJOR OIL PIPELINE. AN ECOPEPETROL [COLOMBIAN PETROLEUM ENTERPRISE] SPOKESMAN SAID THAT THE EXPLOSION TOOK PLACE AT KM - 102 OF THE PIPE NEAR BENADIA IN ARAUCA INTENDANCY, IN THE EASTERN PART OF THE COUNTRY.

Slot	Description	Filler
0	message id	TST2-MUC3-0099
1	template id	1
2	date of incident	07 JUL 89
3	type of incident	BOMBING
4	category of incident	TERRORIST ACT
5	perpetrator: id of indiv	"GUERRILLAS"
6	perpetrator: id of org(s)	"PRO-CASTRO ARMY OF NATIONAL LIBERATION"
7	perpetrator: confidence	SUSPECTED OR ACCUSED BY AUTHORITIES: "PRO-CASTRO ARMY OF NATIONAL LIBERATION"
8	physical target: id(s)	"OIL PIPELINE"
9	physical target: total num	1
10	physical target: type(s)	ENERGY: "OIL PIPELINE"
11	human target: id(s)	-
12	human target: total num	-
13	human target: type(s)	-
14	target: foreign nation(s)	-
15	instrument: type(s)	*
16	location of incident	COLOMBIA: ARAUCA (INTENDANCY): BENADIA (TOWN)
17	effect on physical target	SOME DAMAGE: "OIL PIPELINE"
18	effect on human target(s)	-

Figure 2: A MUC-3 message fragment and its associated template fill.

inconsistencies of human scorers. The number of messages that text understanding systems were required to process in the MUC-3 task increased by an order of magnitude over the number processed in the second conference. Darpa has already announced a fourth message understanding conference, MUC-4, to be held in June, 1992. For this conference, the same domain, task, and scoring procedures will be used that were used for the MUC-3 cycle.

### 3.3 Task Simplification

In both the MUC and ATIS efforts the tasks to be accomplished were simplified versions of *real world* applications. Simplification is necessary because handling a completely realistic task would require building a great deal of hardware and software infrastructure that is really peripheral to the goal of evaluation. On the other hand, oversimplification must be avoided or the behavior evaluated will not scale up to realistic applications.

In ATIS, the task was simplified in several respects. First, the database used was a subset of the actual air travel planning database and did not include information about all cities served. Specifically, only 11 US cities were included in the database. The fact that some information is missing may have unknown effects on the types of queries collected. Second, the wizard was uncooperative in the sense

that if the subject made a query that was outside the domain, the wizard would only issue an error message and would not attempt to help the subject obtain the information. In addition the wizard would never take the initiative and help the subject out if he or she seemed to need direction. This simplification makes the queries easier to evaluate but at the cost of creating a task that does not fully represent the complexities of human-human communication.

The MUC-3 task was simplified in that the systems being evaluated were not expected to deal with an active stream of incoming messages. On the other hand, the corpus processed was far more realistic in magnitude than in previous MUC evaluations, and the task of template filling was acknowledged by government observers to be representative of true applications.

## 4 Procedures

MUC and ATIS are large-scale, cyclic evaluation programs involving a number of independent research groups. For such large scale efforts it is necessary to have a more or less formal process for making decisions and administering the evaluation.

### 4.1 Development of the Applications and Domains

The MUC program directed by Beth Sundheim at NOSC under the sponsorship of DARPA is now planning its fourth cycle of evaluation. Evaluations have taken place in May of 1987, 1989, and 1991. These cycles vary somewhat in application, domain, and complexity. But these variations are not methodical; they have resulted from an evolving view of what an evaluation cycle for text understanding systems ought to be. The ATIS program, administered by NIST through an interagency agreement with DARPA, is a more recently established evaluation effort than MUC.<sup>1</sup> Evaluations have taken place in June of 1990 and February of 1991, and the third evaluation is planned for February of 1992. These evaluations have exhibited less variation than those of MUC. Each cycle has focused on the same application and domain, and the same basic scoring procedure has been used.

The tendency to use the same application and domain that is evolving in the MUC effort and that has been present from the outset of the ATIS effort has the advantage of allowing more informative intercycle performance measures. However, this advantage has a price. By not varying the application and domain between cycles, it is difficult to evaluate portability, at least in the sense of porting speed.

### 4.2 Coordination

Although the MUC and ATIS programs share the tendency to use the same application and domain across evaluation cycles, they differ in the nature of the roles played by the program coordinators and by the participants in the evaluation. In MUC, Beth Sundheim has been the dominant force in shaping the direction taken in the evaluation effort. Although she has created committees to help her make decisions, she has been the driving force in setting the rules for participation, defining the evaluation tasks, locating data, creating scoring guidelines, and so forth. In the ATIS effort, on the

---

<sup>1</sup>Although benchmark testing for speech evaluation began in March of 1987, evaluation of spoken language understanding (using the ATIS domain) began in 1990.

other hand, although NIST coordinates the overall evaluation, there are also several working groups which make recommendations on data collection, evaluation and the relational database.

### 4.3 Documentation and Scoring Software

In both the MUC and ATIS evaluation efforts automatic scoring software has been used. Although this has probably been an important component in keeping the evaluations objective and relatively inexpensive, it introduces another factor into the evaluation process that is peripheral to the research. The selection of scoring software may also affect the amount of time any given research group has to prepare for an evaluation—research groups that differ in their familiarity with the scoring software will differ in the amount of time they spend reading scoring documentation and learning how to use scoring software. This situation arose in the MUC effort when a scoring program built on top of GNU emacs was adopted. Research groups that were not familiar with this text editor spent significantly more time installing the software and learning how to use it.

In defining the evaluation procedures in both the MUC and ATIS efforts, many decisions about various details were made. It is very important to document these decisions in order to allow new groups to participate in the evaluation. Familiarity with scoring techniques and familiarity with the current application and domain of a large-scale effort will have a significant effect on the entry cost for new research groups that have decided to participate in an evaluation program. The entry cost will vary from program to program, and the desire to minimize the cost may vary as well.

## 5 Data

One of the primary areas in which the evaluation trend has benefited the progress of research in natural-language processing systems is where it has pointed out the clear necessity for developing methods to deal with naturally occurring text, with complicated structure, ungrammaticalities, and long sentences. Whether one is developing a "hand-crafted" system, or applying statistical methods to adapt a system to the types of texts typical of a certain domain, it is necessary to have a great deal of data to provide exemplars of the phenomena that arise and to provide a statistically adequate sample of texts in the domain.

For the MUC evaluation, obtaining data is a relatively simple task, since newspaper articles on any topic are readily obtainable in large quantities from any wire service. Although naturally occurring raw data is abundant and easy to obtain, this data is not necessarily useable for system development without considerable processing. First, accurate answer templates for all of the texts must be obtained. If systems wish to use the collected data for training statistical models, it is also desirable to have lexical category tags and structural bracketing accurately assigned—a very time-consuming task.

The problems of gathering data for the ATIS corpus were somewhat different. Because there are no existing systems that perform as the envisioned ATIS systems do, it is necessary to set up artificial "Wizard of Oz" scenarios in which a subject interacts with a person pretending to be a computer. This technique can yield much data; however, the exact characteristics of the data are very sensitive to the precise protocols under which the data is collected. Very small differences in the mental state of the subject (e.g. whether or not the subject is aware that the experiment is a simulation, or believes

he or she is actually talking to a machine) has a significant effect on the linguistic phenomena that are evident in the data. The same observation holds true for the task that is assigned to the user in the data collection experiment. Because the experiment requires presenting an answer interactively to the user, the correct ATIS answers are collected as a byproduct of running the scenario.

The ATIS data was originally collected by several sites under contract to NIST, which compiled the data and distributed it to the participating system developers. However, sites working on ATIS are now moving to a multi-site data collection paradigm, where data is collected by the participating sites themselves and contributed to a common pool. This common data collection paradigm serves both to diversify the training data that sites will see and to reduce the possibility of gaps in the data flow because of problems at one or two sites. However, data collected at multiple sites requires additional efforts in standardization and coordination to make sure that the data are consistent.

The raw MUC data (the original texts) was collected by NOSC, but the generation of key templates was done by the participants themselves, which proved to be time-consuming and distracting for all involved. Part of speech tagging and bracketing of a subset of the data was undertaken by the TREEBANK project ([1]).

Because of the labor required to run the data collection experiments, the ATIS evaluation is relatively data poor, and many sites have supplemented the official data with data collected on their own. MUC participants, on the other hand, found themselves with too much data in one sense and not enough in another. Because raw data is easily obtainable in large quantities, there was no shortage. However, some sites found that a shortage of processed data with accurate answer keys, part of speech tags and/or bracketing hindered their development efforts.

One interesting difference between the MUC and ATIS data collection efforts was that because the MUC data was provided all at once, many economies of scale in processing the data were possible which weren't cost effective in handling the ATIS data, which was provided in smaller increments. For example, in MUC it was cost effective to use batch lexical entry tools for entering hundreds of new words at a time while in ATIS, entering a few new words at a time as the data arrived didn't justify the start up overhead of using batch tools. In addition, with a large amount of data it is possible to more reliably prioritize development efforts on the basis of frequency of occurrence of linguistic phenomena. A large amount of data is helpful in determining to what extent observed phenomena are sporadic or are really representative of the domain.

## 6 Scoring

The intended application of a system determines what evaluation metrics are most appropriate. Since the ATIS application (interactive database information retrieval) and the MUC application (off-line text information extraction) are considerably different, it is not surprising that the evaluation metrics chosen for them are quite different.

Since the intended ATIS application is interactive question answering, the ability of the system to produce correct answers to queries is the relevant capability to be measured. Correct answers are assumed to reflect the correct processing of all relevant aspects of the input, and incorrect answers reflect the failure to handle some relevant aspect of the query, whether in speech recognition or subsequent NL processing. For this application, wrong answers were considered twice as bad as no

answer at all, hence the overall scoring metric:  $Score = Right - Wrong$ .

Because the ATIS systems are intended to function eventually in an interactive setting, it may be considered helpful to a user to provide information that is not explicitly requested. Therefore the scoring program has been quite tolerant of the inclusion of additional database fields not explicitly requested in the answer, even though this tolerance could conceal a system's inability to determine exactly what it was that the user requested. Recently this tolerance has come into question and a stricter scoring methodology is being developed which will penalize a system for exceeding the maximum answer for a query.

The MUC task, on the other hand, involves not question answering, but data extraction. The systems were evaluated on their ability to correctly recover 18 different types of information from each text, and therefore the score involves measures of recall, the overall percentage of correct fills, and precision, the percentage of correct fills greater than those offered. The measurement of precision was further decomposed into measurements of overgeneration, which is the percentage of spurious (false positive) slot fills of all fills offered, and fallout, which pertains only to those slots with a finite number of possible values, and is a measure of the tendency of a system to make errors as the number of possible options increases.

In addition to scoring performance for each of the 18 template slots in the MUC-3 task, the collective performance over all the slots was calculated. However, a known flaw with the MUC-3 evaluation metrics is that there are both logical and statistical dependencies among fillers for various slots, and this confounds methods that tend to treat the filling of any particular slot as an independent event.

Like the ATIS task, the MUC task scoring procedures allow some flexibility in accepting "additional" information in that answers may specify "optional" fills that do not contribute either to recall or overgeneration scores. Because some template slots are filled with strings from the text, and the criteria for determining the correct string fill are incomplete, the participants in MUC-3 were allowed to score inexact matches as partially correct, which counted as half of a correct answer.

The end-user application of the MUC task is less clear than that for the ATIS task, and so it is much less clear how to accurately characterize a system's performance with a single number. Different tradeoffs between recall, precision, and computation may be appropriate for different applications, and therefore there is no clear criteria for the comparison of systems that choose a different tradeoff point between recall and precision. Although most developers have tended to prefer a strategy that emphasizes precision at the expense of recall, the stipulated goal of MUC-3 was a balanced maximization of both parameters.

Neither MUC nor ATIS has yet used statistical significance tests on scores. Statistical significance tests would be helpful because they would tell us both when apparently small differences in fact do reflect reliable differences between system performance, and when apparently large differences do not reflect reliable differences between system performance.

## 7 Benefits

With these formal evaluation programs, we are beginning to be able to compare natural language processing techniques in an objective, quantitative way. However, because spoken language systems and text processing systems are enormously complex, the single number or small set of numbers which

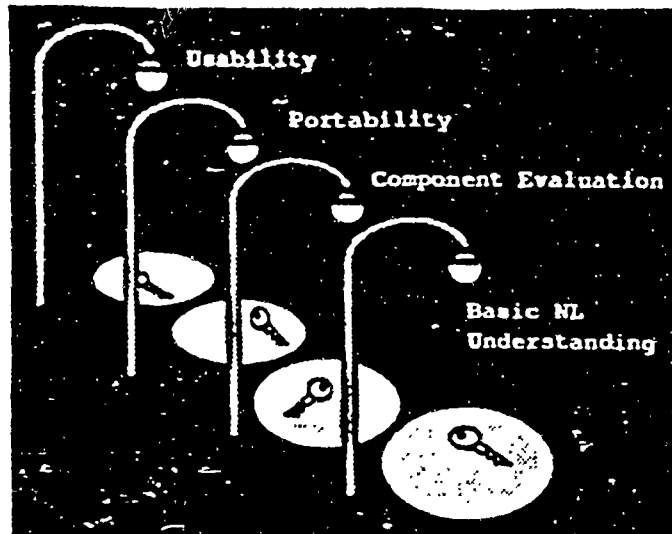


Figure 3: Desirable properties of natural language systems and evaluations

represent a system's performance is a very coarse measure of the value of the various algorithms which make up the system. For this reason we believe it is important to perform parametric intra-system comparisons. By this we mean that scores are presented from a single system using different components. For example, at the most recent ATIS evaluation, Unisys performed three tests, using the same natural language system with three different speech recognizers ([2]). This provided a clean, controlled comparison of the performance of the speech recognizers. Similarly, CMU presented scores of the performance of their system with and without a knowledge-based module ([6]).

## 8 Other Evaluations

Both MUC and ATIS are aimed at comparing basic language understanding capabilities across systems. Other properties of natural language understanding systems are important as well, and it is important not to lose site of these as we become more successful in executing formal evaluations like MUC and ATIS. For example, we want to know how systems compare on usability and portability, as well as how their different components compare, in addition to the basic natural language understanding. Evaluating just what we know how to evaluate is like looking for the keys under the lampost. We don't want to look for the keys just under the lamp post—we want to turn on more lights. As Figure 3 suggests, there are several lights that we may want to turn on.

## 9 Conclusions

By participating in these evaluations, we have learned that:

- The evaluation infrastructure is complex and building it requires active participation from the sites being evaluated.

- The cost of evaluation is high in terms of both amount of time spent on the evaluation and diversion of researchers' energies from other activities. As natural language systems become more portable, these costs may be reduced, since less time will be spent on routine porting.
- The scientific value of these evaluations is tremendous, but we need to continue exploring other forms of evaluation in order to get an evaluation of other desirable properties of natural language systems.

Multi-site evaluation of natural language processing systems as in the ATIS and MUC efforts has been extremely stimulating to the field of natural language processing in the four years since the first MUC evaluation. At the same time an enormous amount of effort has gone into defining the requirements and procedures for carrying out the evaluations. In this paper we have tried to document the requirements of successful multi-site black box evaluations from our perspective as participants in both the MUC and ATIS evaluations so that future evaluations can build on these experiences.

## References

- [1] Mitch Marcus. Very large annotated database of American English. In *Proceedings of the DARPA Speech and Language Workshop*, Hidden Valley, PA, June 1990. Morgan Kaufmann.
- [2] Lewis M. Norton, Marcia C. Linebarger, Deborah A. Dahl, and Nghi Nguyen. Augmented role filling capabilities for semantic interpretation of natural language, February 1991. DARPA Speech and Language Processing Workshop.
- [3] Martha Palmer, Tim Finin, and Sharon Walters. Workshop on evaluation of natural language processing systems, 1990. To appear in *Finite String Newsletter*, AJCL.
- [4] P.J. Price. Evaluation of spoken language systems: the ATIS domain. In *Proceedings of the Speech and Natural Language Workshop*, pages 91-95. Morgan Kaufmann, 1990.
- [5] B. M. Sundheim. Third message understanding conference (muc-3): Phase 1 status report. In *Proceedings of the DARPA Speech and Natural Language Workshop*. Morgan Kaufman Publishers, Inc., Asilomar, CA, 1991.
- [6] Sheryl Young. Using semantics to correct parser output for ATIS utterances. In *Proceedings of the DARPA Speech and Natural Language Workshop*, Asilomar, CA, February 1991.

# Benchmark Investigation/Identification Project <sup>1</sup> Phase I

Jeannette G. Neal  
Calspan Corporation  
4455 Genesee Street  
Buffalo, NY 14225  
neal@cs.buffalo.edu

Elissa L. Feit  
Dept. of Computer Science  
State University of NY at Buffalo  
Buffalo, NY 14260  
feit@cs.buffalo.edu

Christine A. Montgomery  
Language Systems, Inc.  
6269 Variel Avenue, Suite F  
Woodland Hills, CA 91367  
chris@priam.usc.edu

## 1. Introduction

The evaluation of natural language processing (NLP) systems has become an issue of increasing concern within the computational linguistics community and the producers and consumers of NLP products. Evaluation of NLP systems is essential in order to measure capabilities and track improvement in individual systems, compare different systems, measure technical progress and growth in the field, and provide a basis for selecting NLP systems to best fit the communication requirements of applications and applications systems.

The objectives of the Benchmark Investigation/Identification (Benchmark I/I) Project are designed to support these evaluation activities. As part of the Benchmark I/I Program, we are developing a method and procedure for evaluating NLP systems that:

- produces profiles of NLP systems that are:
  - descriptive: the profiles provide descriptive information with regard to the types of linguistic phenomena on which the NLP succeeded or failed, not just one or two numerical scores (e.g., recall and precision) that provide no detailed analysis.
  - hierarchically organized: the capabilities of NLP systems are described by individual capability as well as by class of capability, at the various levels of granularity provided by the hierarchical structure of the profile.
  - quantitative: scores assigned by evaluators to individual test items are aggregated by class, and weighted averages are used to calculate a numerical performance score for each class in the hierarchy.

---

<sup>1</sup>This research is supported by Rome Laboratory under Contract No. F30602-90-C-0034.



- objective: test items are defined in a detailed manner to remove evaluator subjectivity.
- is usable across application domains.
- is applicable across different types of NLP systems such as database query NL front-end systems, text/message processing systems, and interactive NL dialogue interfaces.
- does not require an NLP system to be modified, re-engineered, or re-implemented to adapt it to a particular text corpus or domain. This unique feature sets the Benchmark I/I approach apart from others. Other evaluation efforts (e.g., the MUC evaluations) provide a domain or text corpus to which NLP systems must be adapted or ported even though this porting may be very costly.
- is repeatable; the Procedure produces consistent results, independent of the evaluator.
- does not require that the evaluator be a trained linguist.
- is unbiased with respect to linguistic theories, system-internal processing methods, and knowledge representation techniques.

The Benchmark I/I Evaluation Procedure is being designed to produce comprehensive descriptive evaluation profiles for NLP systems. Such a comprehensive profile should be interpreted in terms of the application requirements for which the NLP system will be used. Since the Benchmark I/I Procedure is being designed to be comprehensive and to be applicable across different application domains and different NLP system types, one would not necessarily expect a particular type of NLP system to excel in all areas. For example, a text processing system that performs well at information extraction to update a database may not process NL queries or commands. On the other hand, a database query NL front-end system may perform extremely well at processing NL queries and commands, but may not process declarative sentences.

This paper discusses the content and structure of the Benchmark I/I Evaluation Procedure and the results of assessing the Procedure in the context of applying it to each of three NLP systems by each of three Interface Technologists at the end of the first six-month phase of the project. Section 2 discusses background and scope. Section 3 briefly presents an overview of the Benchmark I/I Project. Section 4 presents the design principle underlying the Evaluation Procedure and its organization. Section 5 describes the Procedure's content and scoring method and Section 6 discusses the profiles produced by the Procedure. Section 7 reports on the assessment of the Evaluation Procedure. Section 8 presents the current status of the project and future directions. Section 9 summarizes the important issues discussed in this paper and Section 10 provides references. The appendix includes a profile of an NLP system produced by the Benchmark I/I Procedure.

As discussed in Section 3, this paper reports on the results of only the first six-month phase of an 18-month project. So the reader should keep in mind that although the Procedure

is described and results are reported from the application of the Procedure for assessment purposes, this work is not complete.

## 2. Background

There are many different areas and issues for which NLP systems need to be evaluated. Table 1 categorizes and lists many of these issues. The problems in evaluating NLP systems are difficult and many approaches to these different issues have been discussed [Bates90], [BBN88], [Biermann83], [Flickinger87], [Guida86], [Hayes-Roth89], [Hendrix76], [Hershman79], [Hix91], [Kohoutek84], [Lazzara90], [Malhotra75], [Mitta91], [Ogden88], [Palmer89], [Read88], [Sundheim91], [Tennant79], [Weischedel86]. The Benchmark Evaluation Procedure focuses on the *linguistic issues* listed in the first column of the table. The following paragraphs briefly review some of the related evaluation efforts and approaches.

Table 1: Categories of Evaluation Issues

Linguistic Issues	Intelligent Behavior & Reasoning Issues	End User Issues	System Development Issues
lexicon	inference	habitability	quality of tools
syntax	learning	reliability	cost
semantics	cooperative dialogue	likability	ease of development
discourse	speaker/hearer modeling	efficiency	maintainability
pragmatics	real world knowledge	extensibility	portability
			integrability

Several studies have focused on the issue of habitability. In laboratory evaluations, Hershman, Kelly, and Miller [Hershman79] studied ten Navy officers using LADDER, a natural language query system designed to provide easy access to a naval database. The study simulated the actual operational environment in which LADDER would be used and the subjects were trained to the database and LADDER interface. The results of the study indicated that the extensive training given to the subjects was adequate for training the functional and conceptual coverage of the system, but not for training the syntactic and lexical coverage.

Focusing on habitability and efficiency, Biermann, Ballard, and Sigmon [Biermann83] designed an experiment that was concerned with the usefulness of English as a programming language. Their experiment used a natural language programming system, called NLC, that allows a user to display and manipulate tables and matrices while at a display terminal. All user inputs were expressed in English. The results of the study indicated that, with relatively little training on NLC, subjects were able to type system-acceptable syntax with a high enough success rate to obtain correct answers in a reasonable amount of time.

The Performance Evaluation Plan (PEP) [Lazzara90] addresses the evaluation of NLP tools or shells for developing specific NLP applications from a user-oriented perspective, where

three classes of users are identified: systems developers, end users, and systems maintainers. As a result, the PEP provides a methodology for evaluating issues such as integrity, maintainability, extendability, portability, user productivity and likability.

Jayes-Roth [Hayes-Roth89] and Mitta [Mitta91] are concerned with evaluation of knowledge systems and expert systems. Hayes-Roth [Hayes-Roth89] is primarily concerned with extrinsic issues such as advice quality, reasoning correctness, robustness, and solution efficiency; and intrinsic issues such as elegance of knowledge base design, modularity, and architecture. Mitta [Mitta91] discusses a methodology for evaluating an expert system's usability, based on the following six variables or measures: user confidence that the solution is correct, user perception of difficulty, correctness of solution, number of responses required of users, inability of expert system to provide a solution, and rate of help requests. Although focusing on knowledge systems or expert systems, these discussions and methodologies are applicable to NLP systems because NLP systems are special types of knowledge systems.

Several approaches and studies focus on linguistic and NL understanding capabilities. Guida and Mauri [Guida86] have developed a formal and detailed method for evaluating NLP systems. They treat a NLP system as a function from a set of input expressions to one or more sets of outputs. Their method requires a measure of error, defined to compare the closeness of the output with the correct output, and a measure of the importance of each input. Their method of evaluation computes the sum of the errors weighted by the importance of the input.

Several approaches that focus on linguistic capabilities have entailed the development of test corpora for evaluating NL database query interfaces [BBN88], [Hendrix76], [Malhotra75], and [Flickinger87]. Flickinger, Nerbonne, Sag, and Wasow [Flickinger87] developed a test suite of English sentences, annotated by construction type, that covers a wide variety of syntactic and semantic phenomena. The test suite reflects grammatical issues with which linguists have been concerned for a considerable length of time. Anomalous strings are included as well as well-formed sentences.

As part of the Artificial Intelligence Measurement System (AIMS) project [Read88], evaluation criteria and methods for describing linguistic coverage are being developed for NLP systems. As a result, a *Sourcebook* [Read88] is being developed that consists of a database of "exemplars" of representative problems in NL processing. Each exemplar includes a piece of illustrative text, description of the linguistic/conceptual issue at stake, discussion of the problems in understanding the text, and references to more extensive discussion in the literature.

The Naval Ocean Systems Center (NOSC) completed the third evaluation of English text processing systems in May, 1991, with the Third Message Understanding Conference (MUC-3) [Sundheim91]. These evaluations focused on the performance of text analysis systems on an information extraction task. The training corpus consisted of 1300 texts with an overall size of over 2.5 megabytes. The task was to extract information on terrorist incidents from relevant text. At the end of each phase of MUC-3, participating systems were required to

extract information from a test corpus of 100 previously unseen texts. Scoring results for both phases, examples of the development and test corpora, and descriptions of the participating systems are given in the proceedings [Sundheim91].

Finally, an important issue is the reliability of evaluation methods. That is, different evaluators must produce consistent results when applying the same evaluation method to the same target system. Although not directly concerned with NL processing, the approach of Hix and Schulman [Hix91] for testing the reliability of their methodology for evaluating human-computer interface development tools is relevant. To empirically test their methodology, Hix and Schulman had six evaluators each apply the method to two (out of a total three) application tools, so that each tool was evaluated by four different participants. To produce statistical tests of reliability, the researchers computed the probability that responses from the four evaluators for each tool would match by chance. The observed proportion of matches for each category of items was compared with the chance probability using a binomial test.

The Benchmark I/I evaluation method focuses on the linguistic capabilities of NLP systems. Important components of the Benchmark I/I Procedure are the hierarchically structured classification scheme for linguistic phenomena, emphasis on descriptions of the linguistic phenomena covered in the Procedure, and examples illustrating linguistic phenomena. In these aspects, the Benchmark I/I method has some similarity to the Sourcebook approach of Read et al. [Read88]. The Benchmark I/I method also provides a Procedure for testing whether NLP systems are capable of handling the described linguistic phenomena. The Procedure includes patterns, instructions, and illustrative examples for composing NL text for testing purposes and a Profile generator that produces descriptive profiles of NLP systems organized according to the hierarchically structured classification scheme for linguistic phenomena. We are also designing a reliability test for the Benchmark I/I Evaluation Procedure that is somewhat similar to that of Hix and Schulman [Hix91]. In contrast to most of the approaches discussed above, with the exception of the Sourcebook [Read88], the Benchmark I/I evaluation tool is being designed to be applicable to different types of NLP systems and across application domains.

The following section provides an overview of the Benchmark Project. Subsequent sections discuss the design and content of the Benchmark I/I Evaluation Procedure, evaluation Profiles, experience in applying the Procedure, and assessment of the Procedure.

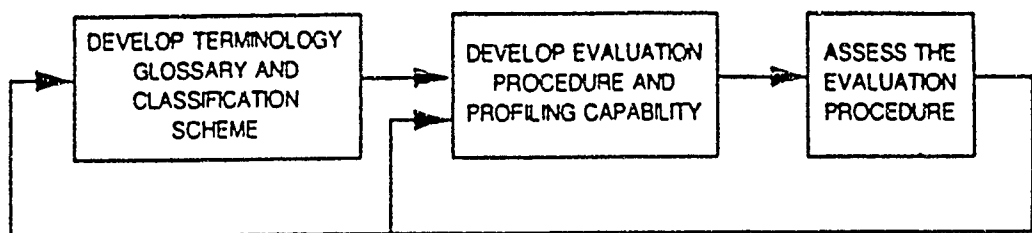
### 3. The Benchmark Investigation/Identification Project

The Benchmark Project is an eighteen-month project that includes the following key tasks:

- development of an Evaluation Procedure that produces profiles of NLP systems consisting of hierarchically organized, quantitative, objective descriptions of the systems' capabilities. Supporting this task is the development of:
  - a database of non-subjective Descriptive Terminology for describing NLP capabilities outside the context of their application to target software.

- a Classification Scheme for NLP capabilities and issues that provides the hierarchical organization.
  - a Procedure to guide the evaluator through the evaluation process. This Procedure assists the evaluator in developing test sentences and provides for the recording of results/scores.
- assessment of the Evaluation Procedure at the end of each of the three six-month development phases. This assessment activity consists of having Interface Technologists, who have had no involvement with the development of the Evaluation Procedure, apply the Procedure to several actual NLP systems. More specifically, each of three Interface Technologists apply the Procedure to each of three NLP systems at each of the three milestones shown in Figure 1. The results of the assessment by the Interface Technologists provide feedback to the developers of the Procedure during its incremental development.

MAJOR TASK FLOW:



TIMELINE:

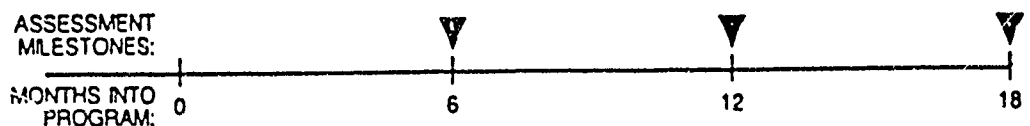


Figure 1: Assessment Milestones

These issues are discussed in more detail in the following sections.

#### 4. Design Principle and Procedure Structure

##### Design Principle

The design of the Evaluation Procedure is based on the principle of having each NLP capability, *C*, tested in at least one Procedure item that includes no other "intruding" NLP capability that might obscure the system's performance on the focal capability *C* for the particular test item. To accomplish this type of design for the Procedure, the Procedure is

being developed so that, for untested individual NLP capabilities, each Procedure item tests just one untested NLP capability at a time, to the extent possible; combinations are tested after individual capabilities are tested. Thus, the Procedure is being designed to progress from very elementary sentence types containing simple constituents to more complex sentence (group) types. The idea is that each time a test sentence (group) is presented to the NLP system being evaluated, the sentence (group) should contain only one new (untested) linguistic capability or one new untested combination of tested capabilities. The other capabilities required for processing the input should already have been tested and the NLP system should already have succeeded on these other issues. The Procedure must avoid the situation in which tests for several capabilities are always combined in the same test sentences, since the Procedure would then be insensitive to the individual capabilities. For example, a test of ellipsis only in the context of question-answering dialogue would not be usable with a system that is not designed to handle questions (e.g., a text understanding system designed for an information extraction task, which typically processes declarative sentences, but not interrogatives or imperatives).

### Descriptive Terminology

In any evaluation effort, it is important to identify and define the evaluation criteria. The Benchmark project focuses on linguistic capabilities; that is, the ability of NLP systems to process the various constructs and phenomena of natural language. As part of this project, we are developing a database of Descriptive Terminology to describe the language constructs and features for which NLP systems are tested in the Benchmark Evaluation Procedure. This Terminology is being developed from the literature on linguistics and computational linguistics. Definitions are based on, or selected from, well-respected literature sources. This terminology is used throughout the Procedure both to identify what is being evaluated in each item and for the system Profiles produced by the Procedure.

### Classification Scheme

The Benchmark Evaluation Procedure is being developed to produce descriptive profiles of NLP system capabilities, displayable at several levels of granularity or detail. A classification scheme is being developed in the form of a hierarchical structure to provide various levels of granularity. Each class of the scheme is representative of a subset of NLP issues or capabilities. The current top level of the hierarchy includes the following classes or types of linguistic phenomena: basic sentence types, simple verb phrases, noun phrases, quantifiers, simple adverbials, comparatives, connectives, and embedded sentences. This classification scheme is not complete nor fixed, of course, since this paper reports on the status after only the first six-month phase. Each of the classes mentioned comprises sub-classes or sub-types of linguistic phenomena. The bottom level of the classification scheme consists of the individual capabilities. As the project continues, issues that are being added to the classification scheme and evaluation procedure include: different verb types, tense and aspect, verb phrases, reference (including anaphoric and cataphoric), ellipsis, and semantics of

events. The classification scheme is not being limited by the current state of the art in NLP capabilities, but is being developed to include generic capabilities of human-to-human communication in natural language that could be applicable to human-machine communication in the future.

## 5. The Evaluation Procedure

The Procedure is being designed to be *domain independent*. Therefore, the Procedure does not include nor rely on a particular corpus of natural language text or sentences. Instead, the test sentences or paragraphs to be processed by the NLP system are composed by the evaluator either during, or prior to, the administration of the Evaluation Procedure. The Procedure is designed to assist the evaluator with the creation, modification, or tailoring of test sentences.

Since the Procedure is being designed for use by people who are not well versed in linguistics, each Procedure test item includes explanatory material that is intended to provide sufficient instruction to enable the evaluator to compose legal test sentences, and to score the performance of NLP systems on these test items.

As stated previously, the Procedure is being designed to progress from very elementary sentence types containing simple constituents to more complex sentence (group) types. The Procedure is being developed so that, for untested individual NLP capabilities, each Procedure item tests just one NLP capability at a time, to the extent possible, and combinations are tested after the individual capabilities are tested. Each Procedure item consists of the following components:

- A brief explanation and definition of the linguistic capability being tested, along with any special instructions for testing. This is particularly important for evaluators with no linguistic background.
- Patterns that define the structure and features of the test sentences to be composed and input to the NLP system under evaluation. The patterns may include non-terminal words from closed classes (e.g., prepositions, connectives). The domain-specific words of the test sentences are supplied by the evaluator, appropriate to the particular application for which the NLP system has been installed and with which it executes.
- Example sentences to aid the evaluator in composing test sentences.
- A box for the evaluator's test sentences.
- A box for the evaluator's score.

Appendix A includes an abbreviated excerpt from the Procedure section on relative clauses. Rows of asterisks mark the places where material has been omitted. However abbreviated,

the example does provide a sample of the type of explanatory material, instructions, examples, and recording provisions that are included in the Procedure.

For each test item in the Procedure, the evaluator submits a NL input to the NLP system being evaluated and determines whether or not the response indicates that the system processed the input correctly and understood the input. The evaluator has four choices of scores to award to the system for each test item, as listed below. The Procedure allows for a score to be split between the score types enabling the evaluator to indicate confidence in the system's correct processing of the input. Total score for one test item should be 1.0.

- Success: The system successfully processed the NL input and indicated by its response that it understood the input.
- Failure: The system failed to understand the NL input.
- Indeterminate: The evaluator was unable to determine whether the system understood the NL input, even after trying more than once to test the particular NL construct or capability.
- Unable to compose NL input: The evaluator was unable to compose a NL test input for the Procedure item. This problem can arise if the language handled by the NLP system being evaluated is so restricted that the words or phrase types necessary to compose test sentences for the Procedure item are lacking. An example would be the attempt to evaluate a system's ability to handle quantification without having any of the quantifiers in the lexicon of the system being evaluated.

## 6. Evaluation Profiles

The Evaluation Procedure is designed to produce descriptive profiles of NLP systems. The profiles are hierarchically organized according to the classification scheme discussed in Section 4. The profiles can be viewed or examined at any level of granularity (levels of granularity corresponding to the hierarchy levels). At the bottom level of the hierarchy are the individual NLP capabilities. At any level other than the bottom level, the scores of the lower level items or classes are combined in a weighted average to produce the score for the parent class or category. The weights are not fixed, but may be specified by the evaluator. They should remain constant when using the Procedure to compare different systems. Figure 2 shows a sample system profile consisting of only the top level of the hierarchy. Appendix B includes a NLP system profile that shows the top three levels of results.



Evaluation Profile : Top Level

	Person 3.		System 3		Unable to		Total			
	Successes.		Failures:		Compose	Indeterminate:	Error:			
	#	%	#	%	A Sentence:		Time			
I. BASIC SENTENCES	20	97.14	1	2.86	0	0.00	0	0.00	0	0:42
II. SIMPLE VERB PHRASES	8	100.00	0	0.00	0	0.00	0	0.00	0	0:18
III. NOUN PHRASES	57.5	76.54	17	14.62	1	1.14	5.5	7.70	2	4:12
IV. QUANTIFIERS	38	47.47	27	33.51	2	2.00	12	17.02	0	2:05
V. SIMPLE ADVERBS	2	100.00	0	0.00	0	0.00	0	0.00	0	0:05
VI. COMPARATIVES	32	48.53	31	49.80	0.5	1.00	0.5	0.67	0	1:52
VII. CONNECTIVES	30	83.33	5	13.89	0	0.00	1	2.78	0	0:44
VIII. EMBEDDED SENTENCES	2	40.00	2	40.00	0	0.00	1	20.00	0	0:12

Figure 2: A System Profile: Top Level Only

## 7. Assessment of the Evaluation Procedure

As stated in Section 1, the objectives of the Benchmark Project specify that the Evaluation Procedure incorporate the following important features: domain independence, consistency across evaluators, applicability across NLP system types, usability without requiring modification or re-engineering of the NLP system being evaluated, and usability by non-linguists.

The assessment task is designed to determine whether these objectives are being met and to provide feedback to the developers to improve and refine the Procedure during its development. As part of the assessment of the Evaluation Procedure, the Procedure is scheduled to be applied to three different NLP systems by each of three evaluators, called Interface Technologists, at the end of each of the three six-month phases of the project. The following are some of the important features of the design of the assessment task:

- To achieve an impartial assessment of the Procedure, the Interface Technologists have had no involvement in the development of the Procedure.
- To ensure that the Procedure is not biased with regard to a particular type of NLP system, a variety of NLP system types are scheduled to be used in the assessment task. As a minimum, database front-end NL query systems and text understanding systems are scheduled for use. The NLP systems selected are a mix of commercially available systems and advanced research products. Additional types of systems will be used during assessment of the Procedure, depending on the availability and cost of using other systems.
- To minimize bias with regard to particular NLP systems, at least one of the NLP

systems used at each assessment milestone is scheduled to be a system that has not previously been used in the assessment activities.

- To ensure that the Procedure can be used with different application domains, NLP systems with at least two different domains are scheduled for use in the assessment activities.
- To minimize bias caused by order of Procedure application by the Interface Technologists, a Latin square design is being used for the Procedure applications. This Latin square design is illustrated in Table 2.

Table 2: Latin Square Design for the Assessment Task

Phase	Evaluator	NLP System		
I	IT #1	SYS#1	SYS#2	SYS#3
I	IT #2	SYS#2	SYS#3	SYS#1
I	IT #3	SYS#3	SYS#1	SYS#2
II	IT #1	SYS#1	SYS#2	SYS#4
II	IT #2	SYS#2	SYS#4	SYS#1
II	IT #4	SYS#4	SYS#1	SYS#2
III	IT #1	SYS#1	SYS#2	SYS#5
III	IT #2	SYS#2	SYS#5	SYS#1
III	IT #5	SYS#5	SYS#1	SYS#2

As part of the assessment task at each of the three six-month milestones, the Evaluation Procedure is being evaluated using several techniques:

- Statistically: Data generated by the nine applications of the Procedure are analyzed statistically, even though the number of subjects is small.
- Critique during use: The Interface Technologists record problems, criticisms, and suggestions regarding individual Procedure items during the use of the Procedure.
- Error Analysis: The developers of the Evaluation Procedure examine the Procedure books completed by each Interface Technologist for each NLP system and examine any errors made by the Technologists, particularly in composing test sentences during their use of the Procedure. An item analysis is performed to identify which items caused problems across Interface Technologists. The errors are investigated to determine the nature and cause of the errors.
- Questionnaire: The Interface Technologists complete an Assessment Questionnaire developed to evaluate the Procedure.

Because of limited space in this paper, we have not included detailed results of all the assessment techniques. They are, however, available from the authors. In assessing the consistency of the Evaluation Procedure across Interface Technologists, the scores for each NLP system were compared across Interface Technologists. The graph of Figure 3 shows the score for each major category for each Interface Technologist. As the graph shows, the

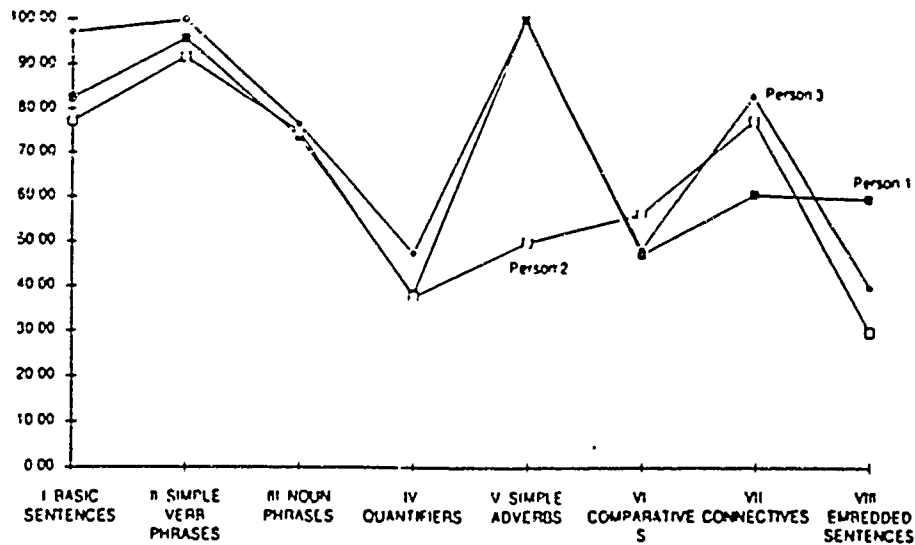


Figure 3: Major Category Scores Across Interface Technologists for System #3

scores were very consistent across Interface Technologists except for the section on Adverbs. This aberration was due to the small section on adverbs, just three items. Furthermore, adverbs were not common in the vocabulary of the NLP systems used. One of the Interface Technologists was able to use some adverb(s) that the system understood, while the other Technologists were not able to do so.

## 8. Current Status and Future Directions

This paper reports on the status of the Benchmark project as of the end of its first six-month phase. Development of the Procedure has continued so that other major categories of linguistic phenomena have been included in the Procedure and Classification Scheme. We will soon be at another assessment milestone, assessing the extended and revised Procedure with two out of three systems being new to the project, and a new additional application domain.

Possible future directions for this effort include continuing to extend the coverage of the Evaluation Procedure, since it will be impossible to cover all relevant linguistic phenomena in this Project. Other areas to which the Procedure could be extended include knowledge acquisition and the handling of ill-formed input. The Procedure could also be improved by

providing the evaluators with automated support for some of the activities performed during the application of the Procedure and the generation of systems' Profiles.

## 9. Summary

This paper has discussed the Benchmark Investigation/Identification Project, the Evaluation Procedure being developed as part of the project, and the results of the first six-month phase of the project. Important features of the Benchmark Procedure include the fact that it is being developed to produce comprehensive profiles of NLP systems that:

- provide descriptive information regarding the linguistic phenomena being tested,
- are hierarchically organized according to a classification scheme that provides various levels of granularity for profile display,
- provide quantitative information based on the scores that are assigned by the evaluators to individual test items; the scores are aggregated by class and weighted averages calculated for each class in the hierarchy, and
- are objective in that test items are defined in a detailed manner so as to remove the subjectivity of the evaluator.

In response to difficulties that have been recognized in other evaluation methods, the Benchmark Procedure is being designed with certain important attributes:

- The Procedure is being developed to be usable with different application domains. The great value in this feature is that the system developers are saved the (possibly considerable) expense of re-engineering or porting the system to a new domain in order to be tested. This feature is unique to the Benchmark Evaluation approach.
- The Procedure is being developed to be usable with different types of NLP systems such as database query NL front-end systems, text/message processing systems, interactive NL dialogue interfaces, etc. Since these classes of NLP systems are not mutually exclusive and some of the different classes have linguistic capabilities in common, the idea is to have a test that can provide comprehensive profiles of these different types of systems. This is another unique feature of the Benchmark Evaluation approach.
- The Procedure is repeatable and produces consistent results, independent of evaluator.
- The Procedure does not require that the evaluator be a trained linguist. The Procedure items include instructions, explanatory material, and examples that enable the evaluator to create, modify, or tailor test sentences for a particular NLP system.

The scoring method provides the evaluator with four choices when scoring a system's performance: *Success*, *Failure*, *Indeterminate*, and *Unable to compose input*. *Indeterminate* means that the evaluator was unable to determine whether the system understood the NL input, even after several different attempts to test for the particular capability. An evaluator may be *unable to compose input* if the language of the NLP system is so restricted that the needed words or phrase types are not available.

As part of the Benchmark Project, the Evaluation Procedure is being assessed to determine whether the Procedure meets the objectives of the project. The assessment task provides for the Procedure to be applied to three different NLP systems by each of three evaluators at the end of each of the three six-month phases of the project. The results of the first phase assessment seemed to indicate that the Procedure, as developed so far, does indeed meet the objectives listed above.

The Benchmark Evaluation Procedure should prove to be a comprehensive evaluation method and tool that is useful for assessing the development of individual systems, for comparing different systems, and for matching NLP systems with the requirements of application tasks and systems. We welcome feedback from members of the computational linguistics community. As the project moves from Phase I into Phase II, the Procedure continues to be extended and improved.

## 10. References

- [Bates90] Bates, M., Boisen, S., and Makhoul, J. 1990. Developing an Evaluation Methodology for Spoken Language Systems, In *Proceedings of the Speech and Natural Language Workshop, June, 1990*, Morgan Kaufmann Publishers, Inc.
- [BBN88] BBN Systems and Technologies Corp. Dec., 1988. Draft Corpus for Testing NL Data Base Query Interfaces, NL Evaluation Workshop, Wayne, PA.
- [Biermann83] Biermann, A.W., Ballard, B.W., and Sigmon, A.H. 1983. An Experimental Study of Natural Language Programming. *International Journal of Man-Machine Studies*, 18, pp. 71-87.
- [Cohen88] Cohen, P.R. and How, A.E. 1988. Towards AI Research and Methodology: Three Case Studies in Evaluation. COINS Technical Report 88-31, University of Massachusetts, Amherst, MA.
- [Flickinger87] Flickinger, D., Nerbonne, J., Sag, I., and Wasow, T. 1987. Toward Evaluation of Natural Language Processing Systems. Technical Report, Hewlett-Packard Laboratories.
- [Guida86] Guida, G. and Mauri, G. 1986. Evaluation of Natural Language Processing Systems: Issues and Approaches. *Proceedings of the IEEE*, 74(7), pp. 1026-1035.

- [Hayes-Roth89] Hayes-Roth, F. 1989. Towards Benchmarks for Knowledge Systems and Their Implementations for Data Engineering. *Transactions on Knowledge and Data Engineering*, Vol. 1, No. 1, March, 1989, pp. 101-109.
- [Hendrix76] Hendrix, G.G., Sacerdoti, E.D. and Slocum, J. 1976. Developing a Natural Language Interface to Complex Data. Technical Report. Artificial Intelligence Center, SRI International.
- [Hermansen87] Hermansen, J. 1987. Message Processing Systems: Evaluation Factors. Technical TM-RD-87-1, Planning Research Corp.
- [Hershman79] Hershman, R.L., Kelly, R.T., and Miller, H.G. 1979. User Performance with a Natural Language Query System for Command Control. Technical Report NPRDC-TR-79-7. San Diego, CA: Navy Personnel Research and Development Center.
- [Hix91] Hix, D. and Schulman, R.S. 1991. Human-Computer Interface Development Tools: A Methodology for their Evaluation. *Communications of the ACM*, Vol. 34. No. 3, pp. 75-87.
- [Kohoutek84] Kohoutek, H. J. 1984. Quality Issues in New Generation Computing. *Proceedings of the International Conference on Fifth Generation Computer Systems*, ICOT, pp. 695-700.
- [Lazzara90] Lazzara, A.V., Tepfenhart, W.T., and Thomas, M. 1990. Language Analysis Domain Independent Exploitation Shell (LADIES): Performance Evaluation Plan (PEP). Interim Technical Report. Knowledge Systems Concepts, Inc.
- [Malhotra75] Malhotra, A. 1975. Design Criteria for a Knowledge-Based Language System for Management: An Experimental Analysis. MIT/LCS/TR-146.
- [Ogden88] Ogden, W. C. 1988. Using Natural Language Interfaces. *Handbook of Human-Computer Interaction*. pp. 281-299.
- [Palmer89] Palmer, M., Finin, T., and Walter, S.M. 1989. Workshop on the Evaluation of Natural Language Processing Systems. RADC-TR-89-302. RADC Technical Report on the Workshop held in Wayne, PA, in December 1988.
- [Read88] Read, W., Quilici, A., Reeves, J., Dyer, J., and Baker, E. 1988. Evaluating Natural Language Systems: A Sourcebook Approach. *COLING-88*, pp. 530-534.
- [Sundheim91] Sundheim, B. (ed.) 1991. *Proceedings of the Third Message Understanding Conference*, Morgan Kaufmann Publishers (forthcoming).
- [Tennant79] Tennant, H. 1979. Experience with the Evaluation of Natural Language Question Answerers. *Proceedings of the Sixth IJCAI*, pp. 874-876.

[Weischedel86] Weischedel, R.M. 1986. Evaluating Natural Language Interfaces to Expert Systems, Proceedings of 1986 IEEE International Conference on Systems, Man, and Cybernetics.

## APPENDIX A. Excerpts from the Benchmark I/I Procedure

NOTE: Rows of asterisks indicate where Procedure material has been omitted.

### 5. Noun Phrase Postmodification

#### 5.1 Relative Clauses

A relative clause is a sentence that is embedded in the postmodification position of a noun phrase. A full relative clause consists of a relative pronoun followed by a sentence or verb phrase with some omitted constituent(s). For example, the sentence

*"The plane [that we saw] was a DC-10"*

includes the relative clause "that we saw" in brackets. This relative clause consists of the relative pronoun "that" followed by the sentence "we saw (the plane)" where "the plane" has been omitted.

Relative pronouns have the double role of referring to the antecedent (the head of the noun phrase being modified) and of functioning as a constituent in the relative clause (e.g., the omitted object *the plane* in the above relative clause).

\*\*\*\*\*  
\*\*\*\*\*

##### 5.1.1 Relative Pronoun as Subject

The structure of this type of relative clause is:

	[Rel-Pronoun]	[VP]
Eg,	[that]	[hired Mary Smith]
Eg,	[who]	[joined the C.S. Department]

In the next three test items, you will use this type of relative clause in the postmodification position of a noun phrase, first with the pronoun *THAT*, then with a personal pronoun, then with a non-personal pronoun.

##### 5.1.1.1 The Relative Pronoun *THAT* - Restrictive Only

Eg, Is John Smith the person [that is V.P. of Finance] ?  
Eg, Who is the person [that is V.P. of Finance] ?  
Eg, List the person [that is V.P. of Finance].

Score: \_\_\_\_\_

\*\*\*\*\*  
\*\*\*\*\*



### 5.1.2 Relative Pronoun as Object

The structure of this type of relative clause is:

[Rel-Pronoun]	[NP]	[Verb]
Eg, [that]	[James Harris]	[hired]
Eg, [who]	[the Chairman]	[promoted]

In the next two test items, you will use this type of relative clause in the postmodification position of a noun phrase.

#### 5.1.2.1 The Relative Pronoun *THAT* as Object - Restrictive Only

Eg, Is John Smith the person [that the sales department promoted] ?  
Eg, Who is the person [that Mark Watson hired] ?

\_\_\_\_\_ Score: \_\_\_\_\_

\*\*\*\*\*  
\*\*\*\*\*

### 5.1.4 Relative Pronoun Prepositional Object

#### 5.1.4.1 Preposition First in the Relative Clause

The structure of this type of relative clause is:

[Preposition]	[Rel-Pronoun]	[NP]	[VP]
Eg, [for]	[whom]	[Jim Davis]	[works]

#### 5.1.4.1.1 Preposition First with Personal Pronoun

Eg, Is John Smith a person [for whom you have an address] ?  
Eg, List the employees [for whom John Smith is supervisor].

\_\_\_\_\_ Score: \_\_\_\_\_

\*\*\*\*\*  
\*\*\*\*\*

## APPENDIX B. Example NLP System Profile

Evaluation Profile

		Person 3,		System 3											
		Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:		Total Time		Average Time Per Item:	
		#	%	#	%	#	%	#	%	#	%				
<b>I. BASIC SENTENCES</b>		20	97.14	1	2.86	0	0.00	0	0.00	0	0.00	0	0.42	0:02:00	
1 Imperative Sentences		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2 Interrogative Sentences		19	96.43	1	3.57	0	0.00	0	0.00	0	0.00	0			
2.1 What-questions		5	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.1.1 WHAT as a pronoun.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.1.2 WHAT as a determiner.		3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.2 Who-questions.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.3 Where-questions.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.4 When-questions.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.5 Which-questions. (WHICH as Det.)		3	75.00	1	25.00	0	0.00	0	0.00	0	0.00	0			
2.6 How-questions.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.6.1 How [Adjective] [BE-Verb] [NP] ?		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.6.2 How [Adverb] [BE-Verb] [NP] ?		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.6. Yes/No-Questions.		3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
<b>II. SIMPLE VERB PHRASES</b>		8	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0:18	0:02:15	
1 Copular Verb Phrases		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2 Verb Phrases with Primary Verb DO.		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.a Yes/No-Questions.		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.b Wh-Questions.		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
3 Transitivity		3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
3.a Simple transitive verb phrase.		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
3.b Simple intransitive verb phrase.		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
3.c Simple ditransitive verb phrase.		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
4 Voice		2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
<b>III. NOUN PHRASES</b>		57.5	76.54	17	14.62	1	1.14	5.5	7.70	2	4:12	0:03:07			
1 PP as Postmodifier in a NP		3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2 The Noun Head		2.5	62.50	1.5	37.50	0	0.00	0	0.00	0	0.00	0			
2.1 Nouns		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.1.1 Mass Nouns		1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0			
2.2 Nominals		1.5	50.00	1.5	50.00	0	0.00	0	0.00	0	0.00	0			

Evaluation Profile

Person 3, System 3		Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:		Total Time:		Average Time Per Item:	
	#	%	#	%	#	%	#	%	#	%	#	%			
2.2.1 Adjective Nominal	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
2.2.2 Passive Participle as Nominal	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00			
2.2.3 Progressive Participle as Nominal	0.5	50.00	0.5	50.00	0	0.00	0	0.00	0	0.00	0	0.00			
<b>3 Determinatives in More Detail</b>	<b>15.5</b>	<b>73.81</b>	<b>4</b>	<b>19.05</b>	<b>1</b>	<b>4.76</b>	<b>0.5</b>	<b>2.38</b>	<b>0</b>	<b>0.00</b>	<b>0</b>	<b>0.00</b>			
3.1 Predeterminers	2.5	83.33	0.5	16.67	0	0.00	0	0.00	0	0.00	0	0.00			
3.1.1 Quantifier Predeterminers	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.1.2 Multipliers, Fractions	0.5	50.00	0.5	50.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2 Central Determiners	9	69.23	3.5	26.92	0	0.00	0	0.00	0.5	3.65	0	0.00			
3.2.5 Possessive Determiners	5	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.5 Third-Person Singular	3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.5.1 Feminine	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.5.2 Masculine	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.5.3 Neuter	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.6 Third-Person Plural	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.6.1 Personal	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.5.6.2 Impersonal	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.6 Relative Determiners	1	50.00	1	50.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.6.1 Personal (WHOSE)	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.6.2 Impersonal (WHICH)	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.7 Interrogative Determiners	3	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.7.1 Indefinite Reference (WHAT)	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.7.2 Definite Reference (WHICH)	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.7.3 Personal Reference (WHOSE)	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.8 Wh-Determiners	0	0.00	2.5	83.33	0	0.00	0	0.00	0.5	16.67	0	0.00			
3.2.8.1 Indefinite Reference (WHATEVER)	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.2.8.2 Definite Reference (WHICHEVER)	0	0.00	0.5	50.00	0	0.00	0	0.00	0.5	50.00	0	0.00			
3.2.8.3 Personal Reference (WHOSEVER)	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.3 Postdeterminers	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.3.1 Cardinals	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.3.2 Ordinals	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.3.2.1 Ordinal Numbers	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
3.4 Determinative Combinations	2	66.67	0	0.00	1	33.33	0	0.00	0	0.00	0	0.00			
4 Premodification	22	69.90	9	13.90	0	0.00	0	0.00	5	16.19	0	0.00			
4.2 Central	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			

Evaluation Profile

	Person 3, System 3										Total Time	Average Time Per Item
	Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:			
	#	%	#	%	#	%	#	%				
4.2.1 Simple Adjectives	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.2.2 Superlative Adjectives	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.3 Precentral	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.4 Particles	1	50.00	0	0.00	0	0.00	1	50.00	0	0		
4.4.1 Passive Participle	0	0.00	0	0.00	0	0.00	1	100.00	0	0		
4.4.2 Progressive Participle	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.5 Noun-Noun Phrases	4	26.57	7	50.00	0	0.00	3	21.43	0	0		
4.5.1 HAS-PART Nominal Compound Type	0	0.00	0	0.00	0	0.00	1	100.00	0	0		
4.5.2 PART-OF Nominal Compound Type	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.5.3 MADE-OF Nominal Compound Type	0	0.00	0	0.00	0	0.00	0	0.00	0	0		
4.5.4 AT-TIME Nominal Compound Type	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.5.5 AT-LOCATION Nominal Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.6 PRODUCED-from Nom. Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.7 PRODUCED-BY Nom. Compound Type	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.5.8 PURPOSE-BENEFITS Nom. Compound Type	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.5.9 PURPOSE-IS Nominal Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.10 PURPOSE-APPLIES-TO Nom. Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.11 PURPOSE-CONCERNS Nom. Compound Type	0	0.00	0	0.00	0	0.00	1	100.00	0	0		
4.5.12 PURPOSE-PRODUCES Nom. Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.13 USES Nominal Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.5.14 HAS-TYPE Nominal Compound Type	0	0.00	1	100.00	0	0.00	0	0.00	0	0		
4.6 Genitives and Possessives	9	90.00	1	10.00	0	0.00	0	0.00	0	0		
4.6.1 Genitive as Possessive	3	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.1.1 Genitive as Possessive with Simple	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.1.2 Gen. as Possessive with Complex	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.1.3 Genitive as Poss. with Multiple Genitives	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.2 Possessive Expressed Using a Preposition	2	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.2.1 Possessive Expressed Using OF	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.2.2 Possessive Expressed Using WITH	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.3 Verbs Indicating Possession	2	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.3.1 Possessive Expressed with HAVE	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.3.2 Possessive Expressed with OWN	1	100.00	0	0.00	0	0.00	0	0.00	0	0		
4.6.4 Combinations of the Above Types	2	66.67	1	33.33	0	0.00	0	0.00	0	0		
4.6.4.1 Prepositions OF and WITH	1	100.00	0	0.00	0	0.00	0	0.00	0	0		

Evaluation Profile

	Person 3, System 3									
	Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:	Total Time:
	#	%	#	%	#	%	#	%		
4.6.4.2 Verb HAVE and Preposition OF	1	100.00	0	0.00	0	0.00	0	0.00	1	
4.6.4.3 Genitive and Preposition OF	0	0.00	1	100.00	0	0.00	0	0.00	0	
4.7 Premodifier Combinations	5	71.43	1	14.29	0	0.00	1	14.29	0	
5 Postmodification	14.5	87.86	2.5	12.14	0	0.00	0	0.00	1	
5.1 Relative Clauses	14.5	87.86	2.5	12.14	0	0.00	0	0.00	1	
5.1.1 Relative Pronoun as Subject	3	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.1.1 The Relative Pronoun THAT - Re	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.1.2 Personal Relative Pronoun WHO	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.1.3 Non-Personal Rel. Pronoun WH	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.2 Relative Pronoun as Object	2.5	83.33	0.5	16.67	0	0.00	0	0.00	0	
5.1.2.1 Restricted Rel. Pronoun THAT	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.2.2 Personal Relative Pronoun WHO	0.5	50.00	0.5	50.00	0	0.00	0	0.00	0	
5.1.2.3 Non-Personal Rel. Pronoun WH	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.3 Deletion of Relative Pronoun Object	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4 Relative Pronoun Prepositional Object	4	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.1 Preposition First in the Relative C	2	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.1.1 Prep. First with Personal Pr	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.1.2 Prep. First with Non-Pers. P	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.2 Prep. Following the Verb in Rel.	2	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.2.1 With Personal Pronoun	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.4.2.2 With Non-Personal Pronoun	1	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.5 Relative Pronoun as Adverbial	3	60.00	2	40.00	0	0.00	0	0.00	0	
5.1.5.1 Rel. Pronoun as Adverbial for PL	2	100.00	0	0.00	0	0.00	0	0.00	0	
5.1.5.2 Rel. Pronoun as Adverbial for TM	1	50.00	1	50.00	0	0.00	0	0.00	0	
5.1.5.3 Omitted Rel. Pronoun - Adverbia	0	0.00	1	100.00	0	0.00	0	0.00	0	
5.1.6 Multiple Relative Clauses w/ the Sam	1	100.00	0	0.00	0	0.00	0	0.00	0	
IV. QUANTIFIERS	38	47.47	27	33.51	2	2.00	12	17.02	0	2:05
1 Determiners	19	54.40	10	28.96	0	0.00	6	16.65	0	
1.1 Predeterminer Quantifiers	4.5	64.29	2	28.57	0	0.00	0.5	7.14	0	
1.1.1 Predeterminer Quantifiers with Count	3.5	56.33	2	33.33	0	0.00	0.5	8.33	0	
1.1.1.1 Universals	2	50.00	2	50.00	0	0.00	0	0.00	0	
1.1.1.2 Multipliers and Fractions	1.5	75.00	0	0.00	0	0.00	0.5	25.00	0	

Person 3, System 3

	Successes:		Failures:		Unable to Compose		Indeterminate:		Error:	Total Time:	Average Time Per Item:
	#	%	#	%	A Sentence:	%	#	%			
1.1.2 Predeterminer Quantifiers with Mass	1	100.00	0	0.00	0	0.00	0	0.00	0		
1.2 Post-determiner Quantifiers	3.5	50.00	1	14.29	0	0.00	2.5	35.71	0		
1.2.0 Non-Numerical Quantification	1	33.33	1	33.33	0	0.00	1	33.33	0		
1.2.1 Numerical Quantification	2.5	62.50	0	0.00	0	0.00	1.5	37.50	0		
1.2.1.1 Cardinal Numbers	2	100.00	0	0.00	0	0.00	0	0.00	0		
1.2.1.2 Ordinal Numbers	0.5	25.00	0	0.00	0	0.00	1.5	75.00	0		
1.3 Pre- and Post-Determiner Combinations	4	50.00	2	25.00	0	0.00	2	25.00	0		
1.4 Quantitative Determiners	7	53.85	5	38.46	0	0.00	1	7.69	0		
1.4.1 Universal	2	100.00	0	0.00	0	0.00	0	0.00	0		
1.4.2 Existential (assertive)	0	0.00	3	100.00	0	0.00	0	0.00	0		
1.4.2.1 Count	0	0.00	2	100.00	0	0.00	0	0.00	0		
1.4.2.1.1 Singular	0	0.00	1	100.00	0	0.00	0	0.00	0		
1.4.2.1.2 Plural	0	0.00	1	100.00	0	0.00	0	0.00	0		
1.4.2.2 Mass	0	0.00	1	100.00	0	0.00	0	0.00	0		
1.4.3 Existential (non-assertive)	3	60.00	1	20.00	0	0.00	1	20.00	0		
1.4.3.1 Either	1	50.00	1	50.00	0	0.00	0	0.00	0		
1.4.3.1.1 Singular Sense	0	0.00	1	100.00	0	0.00	0	0.00	0		
1.4.3.1.2 Plural Sense	1	100.00	0	0.00	0	0.00	0	0.00	0		
1.4.3.2 Any	2	66.67	0	0.00	0	0.00	0	0.00	0		
1.4.3.2.1 Count	1	50.00	0	0.00	0	0.00	1	33.33	0		
1.4.3.2.1.1 Singular	0	0.00	0	0.00	0	0.00	1	50.00	0		
1.4.3.2.1.2 Plural	1	100.00	0	0.00	0	0.00	0	0.00	0		
1.4.3.2.2 Mass	1	100.00	0	0.00	0	0.00	0	0.00	0		
1.4.4 Negative Quantification	2	66.67	1	33.33	0	0.00	0	0.00	0		
2 Quantifier Pronouns	15	44.41	17	47.20	0	0.00	3	8.39	0		
2.1 Indefinite Quantifier Pronouns	8	66.67	4	33.33	0	0.00	0	0.00	0		
2.1.1 Universal	2	100.00	0	0.00	0	0.00	0	0.00	0		
2.1.1.1 Personal	1	100.00	0	0.00	0	0.00	0	0.00	0		
2.1.1.2 Nonpersonal	1	33.33	2	66.67	0	0.00	0	0.00	0		
2.1.2 Assertive	0	0.00	2	100.00	0	0.00	0	0.00	0		
2.1.2.1 Personal	1	100.00	0	0.00	0	0.00	0	0.00	0		
2.1.2.2 Nonpersonal	1	100.00	0	0.00	0	0.00	0	0.00	0		
2.1.3 Nonassertive	3	100.00	0	0.00	0	0.00	0	0.00	0		
2.1.3.1 Personal	2	100.00	0	0.00	0	0.00	0	0.00	0		

Evaluation Profile

Person 3, System 3		Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Total Time:	Average Time Per Item:
		#	%	#	%	#	%	#	%		
2.1.3.2 Nonpersonal		1	100.00	0	0.00	0	0.00	0	0.00	0	
2.1.4 Negative		1	33.33	2	66.67	0	0.00	0	0.00	0	
2.1.4.1 Personal		1	50.00	1	50.00	0	0.00	0	0.00	0	
2.1.4.2 Nonpersonal		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.2 Numerical Quantifier Pronoun		2	50.00	1	25.00	0	0.00	1	25.00	0	
2.2.1 Cardinals		2	100.00	0	0.00	0	0.00	0	0.00	0	
2.2.2 Ordinals		0	0.00	1	50.00	0	0.00	1	50.00	0	
2.3 Indef Quant. Pronouns w/ Partitive OF		5	26.32	12	63.16	0	0.00	2	10.53	0	
2.3.1 Universal Pronouns		3	100.00	0	0.00	0	0.00	0	0.00	0	
2.3.2 Existential Assertive		1.5	12.50	9.5	79.17	0	0.00	1	8.33	0	
2.3.2.0 Standard Existential		0	0.00	2	100.00	0	0.00	0	0.00	0	
2.3.2.0.1 Mass		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.0.2 Count		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.1 Multial		0.5	12.50	3.5	87.50	0	0.00	0	0.00	0	
2.3.2.1.1 Multial (Mass)		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.1.2 Multial (Count)		0.5	16.67	2.5	83.33	0	0.00	0	0.00	0	
2.3.2.1.2.1 Absolute:		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.1.2.2 Comparative:		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.1.2.3 Superlative:		0.5	50.00	0.5	50.00	0	0.00	0	0.00	0	
2.3.2.2 Paucal		1	16.67	4	66.67	0	0.00	1	16.67	0	
2.3.2.2.1 Paucal (Mass)		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.2.2 Paucal (Count)		1	20.00	3	60.00	0	0.00	1	20.00	0	
2.3.2.2.2.1 Absolute:		0	0.00	1	100.00	0	0.00	0	0.00	0	
2.3.2.2.2.2 Comparative:		0	0.00	2	100.00	0	0.00	0	0.00	0	
2.3.2.2.2.3 Superlative:		0.5	25.00	0.5	25.00	0	0.00	1	50.00	0	
2.3.3 Existential Non-Assertive		0	0.00	2	100.00	0	0.00	0	0.00	0	
2.3.4 Negative		4	50.00	0	0.00	0	0.00	2	25.00	0	
3 Existential THERE		0	0.00	0	0.00	0	0.00	0	0.00	0	
3.a Imperatives		2	100.00	0	0.00	0	0.00	0	0.00	0	
3.b WHAT Questions		2	100.00	0	0.00	0	0.00	0	0.00	0	
3.c Yes-No Questions		0	0.00	0	0.00	0	0.00	0	0.00	0	
3.d Tag Questions		0	0.00	0	0.00	0	0.00	0	0.00	0	
4 Open Class Quantifiers		0	0.00	0	0.00	0	0.00	0	0.00	0	
4.1 Universal Adjectives		0	0.00	0	0.00	0	0.00	1	100.00	0	



Evaluation Profile

Person 3, System 3		Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:	Total Time:	Average Time Per Item:
	#	%	#	%	#	%	#	%	#	%		
<b>V. SIMPLE ADVERBS</b>												
1 Adverb Following a Verb	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0:02:30
2 Adverb Following a Verb Phrase	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
<b>VI. COMPARATIVES</b>												
1 Comparative Adjectives	32	48.53	31	49.80	0.5	1.00	0.67	0	0.00	0	1:52	0:01:45
1.1 Comparison to a Higher Degree	10	57.81	8	42.19	0	0.00	0.00	0	0.00	0		
1.1.0 Without Gapping	3	75.00	1	25.00	0	0.00	0.00	0	0.00	0		
1.1.1 With Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.1.1.1 Adjective Gapping	3	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.1.1.2 Copula and Adjective Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.1.1.3 NP and Copula Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.2 Comparison to a Lower Degree	3	75.00	1	25.00	0	0.00	0.00	0	0.00	0		
1.2.0 Without Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.2.1 With Gapping	3	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.2.1.1 Adjective Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.2.1.2 Copula and Adjective Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.2.1.3 NP and Copula Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.3 Comparisons To the Same Degree	3	37.50	5	62.50	0	0.00	0.00	0	0.00	0		
1.3.1 in Assertive Contexts	3	75.00	1	25.00	0	0.00	0.00	0	0.00	0		
1.3.1.0 Without Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.3.1.1 With Gapping	3	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.3.1.1.1 Adjective Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.3.1.1.2 Copula and Adjective Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.3.1.1.3 NP and Copula Gapping	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		
1.3.2 in Non-Assertive Contexts	0	0.00	4	100.00	0	0.00	0.00	0	0.00	0		
1.3.2.0 Without Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.3.2.1 With Gapping	0	0.00	3	100.00	0	0.00	0.00	0	0.00	0		
1.3.2.1.1 Adjective Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.3.2.1.2 Copula and Adjective Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.3.2.1.3 NP and Copula Gapping	0	0.00	1	100.00	0	0.00	0.00	0	0.00	0		
1.4 Comparison to a Constant	1	100.00	0	0.00	0	0.00	0.00	0	0.00	0		

Evaluation Profile  
Person 3, System 3

	Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:	Total Time:	Average Time Per Item:
	#	%	#	%	#	%	#	%			
	1.5 Comparative Adjs as Pre-Mods in NP	0	0.00	1	100.00	0	0.00	0			
2 Superlatives	3	100.00	0	0.00	0	0.00	0	0.00	0		
2.1 Superlatives to a Higher Degree	2	100.00	0	0.00	0	0.00	0	0.00	0		
2.2 Superlatives to a Lower Degree	1	100.00	0	0.00	0	0.00	0	0.00	0		
3 Comparative Adverbial Phrase	9	71.11	4	28.89	0	0.00	0	0.00	0		
3.1 Comparison to a Higher Degree	3	100.00	0	0.00	0	0.00	0	0.00	0		
3.1.0 Without Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.1.1 With Gapping	2	100.00	0	0.00	0	0.00	0	0.00	0		
3.1.1.1 Verb phrase gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.1.1.2 Noun phrase gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.2 Comparison to a Lower Degree	2	66.67	1	33.33	0	0.00	0	0.00	0		
3.2.0 Without Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.2.1 With Gapping	1	50.00	1	50.00	0	0.00	0	0.00	0		
3.2.1.1 Verb Phrase Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.2.1.2 Noun Phrase Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0		
3.3 Comparisons to the Same Degree	3	50.00	3	50.00	0	0.00	0	0.00	0		
3.3.1 In Assertive Contexts	3	100.00	0	0.00	0	0.00	0	0.00	0		
3.3.1.0 Without Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.3.1.1 With Gapping	2	100.00	0	0.00	0	0.00	0	0.00	0		
3.3.1.1.1 Verb Phrase Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.3.1.1.2 Noun Phrase Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0		
3.3.2 in Non-Assertive Contexts	0	0.00	3	100.00	0	0.00	0	0.00	0		
3.3.2.0 Without Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0		
3.3.2.1 With Gapping	0	0.00	2	100.00	0	0.00	0	0.00	0		
3.3.2.1.1 Verb Phrase Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0		
3.3.2.1.2 Noun Phrase Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0		
3.5 Comparisons to a Constant	1	100.00	0	0.00	0	0.00	0	0.00	0		
4 Superlative Adverbs	2	66.67	0.5	16.67	0	0.00	0	0.00	0		
4.1 Superlative Advs to a Higher Degree	1	50.00	0.5	25.00	0	0.00	0	0.00	0		
4.2 Superlative Advs to a Lower Degree	1	100.00	0	0.00	0	0.00	0	0.00	0		
5 THAN as a Connective	0	0.00	1	100.00	0	0.00	0	0.00	0		
6 Comparative Noun Phrases	2	25.00	6	75.00	0	0.00	0	0.00	0		
6.1 Quantitative	2	50.00	2	50.00	0	0.00	0	0.00	0		

Evaluation Profile

Person 3, System 3		Successes:		Failures:		Unable to Compose		Indeterminate:		Total Time:	Average Time Per Item:	
	#	%	#	%	#	%	#	%	#	%		
6 1 0 Without Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
6 1 1 With Gapping	2	66.67	1	33.33	0	0.00	0	0.00	0	0.00	0	
6 1 1.1 Object Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
6 1 1.2 Verb and Object Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
6 1 1.3 Noun Phrase and Verb Gapping	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
6 2 Qualitative	0	0.00	4	100.00	0	0.00	0	0.00	0	0.00	0	
6 2 0 Without Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
6 2 1 With Gapping	0	0.00	3	100.00	0	0.00	0	0.00	0	0.00	0	
6 2 1.1 Object Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
6 2 1.2 Verb and Object Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
6 2 1.3 Noun Phrase and Verb Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
7 Comparatives with Adjuncts	6	40.00	9	60.00	0	0.00	0	0.00	0	0.00	0	
7 1 Quantitative Comparatives	6	40.00	9	60.00	0	0.00	0	0.00	0	0.00	0	
7 1 0 Without Gapping	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
7 1 1 Quant Comparative - Single Gapping	1	25.00	3	75.00	0	0.00	0	0.00	0	0.00	0	
7 1 2 Quant Comparative - Double Gapping	2	33.33	4	66.67	0	0.00	0	0.00	0	0.00	0	
7 1 3 Quant Comparative - Triple Gapping	3	75.00	1	25.00	0	0.00	0	0.00	0	0.00	0	
7 2 Qualitative Comparatives	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
7 2 0 Without Gapping	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
7 2 1 Qual Comparative - Single Gapping	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
7 2 2 Qual Comparative - Double Gapping	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
7 2 3 Qual Comparative - Triple Gapping	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
8 Implicit comparatives	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
9 Multipliers/Fractions In Comparison	0	0.00	2	100.00	0	0.00	0	0.00	0	0.00	0	
9 1 Multipliers	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
9 2 Fractions	0	0.00	1	100.00	0	0.00	0	0.00	0	0.00	0	
VII. CONNECTIVES	30	83.33	5	13.89	0	0.00	1	2.78	0	0.00	0	0:44
1 Coordinators (Single Pair Coord.)	30	83.33	5	13.89	0	0.00	1	2.78	0	0.00	0	
1.1 Conjunction with AND	30	83.33	5	13.89	0	0.00	1	2.78	0	0.00	0	
1 1 1 Sentential Connectives	2	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
1 1 1.1 Imperatives Conjoined by AND	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	
1 1 1.2 Interrogatives Conjoined by AND	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	

Evaluation Profile

Person 3, System 3		Successes:		Failures:		Unable to Compose A Sentence:		Indeterminate:		Error:		Total Time:		Average Time Per Item:	
	#	%	#	%	#	%	#	%	#	%	#	%			
1.1.2 Relative Clauses Conjoined by AND	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.3 Noun Phrase Constructions with AND	4	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.3.1 Conjunction of Simple Noun	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.3.2 Conjunction of [Adjective][Noun]	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.3.3 Conj. of [Article][Adjective][Noun]	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.3.4 Conj. of [Article][Adj][Noun][Postp]	1	100.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00			
1.1.5 Verb Phrases with the Conjunction AND	18	90.00	2	10.00											
1.1.5.1 [BE-Verb] Conjoined by AND	5.5	91.67	0.5	8.33											
1.1.5.1.1 [BE-Verb] with Complement	3	100.00	0	0.00											
1.1.5.1.2 Passive Constructions Conj.	2.5	83.33	0.5	16.67											
1.1.5.2 [DO-Verb] Constructs with Conjunction	0.5	50.00	0.5	50.00											
1.1.5.3 Full Verb Constructs with Conjunction	2	100.00	0	0.00											
1.1.5.3.1 Conjunction of Intransitive Verb	1	100.00	0	0.00											
1.1.5.3.2 Conjunction of [Verb + IRAN]	1	100.00	0	0.00											
1.1.5.4 Mixed VPs in Constructions with Conjunction	0	100.00	0	0.00											
1.1.5.5 Scope in Verb Phrases	1	50.00	1	50.00											
1.1.6 Conjunction AND in Premodification	3	60.00	1	20.00											
1.1.6.1 Conjunctions of Adjectives	1	100.00	0	0.00											
1.1.6.2 Conjunctions of Article with Adjective	0	0.00	1	100.00											
1.1.6.3 Conjunction of Genitives	1	100.00	0	0.00											
1.1.6.4 Scope of Adjectives over Nouns	1	50.00	0	0.00											
1.1.7 Adverbs Conjoined by AND	0	0.00	1	100.00											
1.1.8 Prepositions and PPs with the Conjunction AND	2	66.67	1	33.33											
VIII. EMBEDDED SENTENCES	2	40.00	2	40.00											
1 THAT Clauses	0	0.00	1	100.00											
2 Wh-Interrogative clauses	1	100.00	0	0.00											
3 Yes-No Interrogative clauses	1	100.00	0	0.00											
4 Infinitive clauses Introduced by TO	0	0.00	0	0.00											
5 -ING Clauses	0	0.00	1	100.00											
													0	0:12	0:02:24
													1	20.00	
													0	0.00	
													0	0.00	
													0	0.00	
													0	0.00	
													0	0.00	
													1	100.00	
													0	0.00	
													0	0.00	

## Evaluating Syntax Performance of Parser/Grammars of English

Philip Harrison, Boeing Computer Services  
Steven Abney, Bellcore  
Ezra Black, IBM  
Dan Flickinger, Hewlett Packard  
Claudia Gdaniec, Logos, Inc.  
Ralph Grishman, NYU  
Donald Hindle, AT&T  
Robert Ingria, BBN  
Mitch Marcus, U. of Pennsylvania  
Beatrice Santorini, U. of Pennsylvania  
Tomek Strzalkowski, NYU

We report on an ongoing collaborative effort to develop criteria, methods, measures and procedures for evaluating the syntax performance of different broad-coverage parser/grammars of English. The project was motivated by the apparent difficulty of comparing different grammars because of divergences in the way they handle various syntactic phenomena. The availability of a means for useful comparison would allow hand-bracketed corpora, such as the University of Pennsylvania Treebank, to serve as a source of data for evaluation of many grammars. The project has progressed to the point where the first version of an automated syntax evaluation program has been completed and is available for testing. The methodology continues to undergo refinement as more data is examined.

The project began with a comparison of hand syntactic analyses of 50 Brown Corpus sentences by grammarians from nine organizations: Steve Abney (Bellcore), Ezra Black (IBM), Dan Flickinger (Hewlett Packard), Claudia Gdaniec (Logos), Ralph Grishman and Tomek Strzalkowski (NYU), Philip Harrison (Boeing), Donald Hindle (AT&T), Robert Ingria (BBN), and Mitch Marcus and Beatrice Santorini (U. of Pennsylvania). The purpose of the bracketing exercise was to provide a focus for the discussion of syntactic differences and a source of data to test proposals for evaluation techniques. The participating grammarians produced labelled bracketings representing what they ideally want their grammars to specify. After the exercise was completed, a small workshop was held at the University of Pennsylvania to discuss the results and examine proposals for evaluation methodologies.

The results of the hand-bracketing exercise revealed that very little structure was common to all the parses. For example, an analysis revealed that the following three Brown Corpus sentences (taken from what we call the "consensus" parses) display only the indicated phrases in common to all of the bracketings:

The famed Yankee Clipper, now retired, has been assisting (as (a batting coach)).

One of those capital-gains ventures, in fact, has saddled him (with (Gore Court)).

He said this constituted a (very serious) misuse (of the (Criminal court) processes).

A rather more encouraging result was obtained when phrases were selected which appeared bracketed in a majority of the analyses (the "majority" parses):

(((((The famed (Yankee Clipper)) , (now retired) ,)  
(has been (assisting (as (a batting coach)))))) .)

((((One (of (those capital-gains ventures))) , (in fact) ,  
(has (saddled him (with (Gore Court)))))) .)

((He (said (this (constituted (a (very serious) misuse  
(of (the (Criminal court) processes)))))) .)

The lack of structure for the consensus parses is a reflection of the diversity of approaches to such phenomena as punctuation, the employment of null nodes by the grammar, and the attachment of auxiliaries, negation, pre-infinitival 'to', adverbs, and other types of constituents. But the results for the majority parses indicated that a good foundation of agreement exists among the several grammars.

The challenge was to find an evaluation method that would not penalize even those analyses that diverged from the majority in ways that would be considered generally acceptable. The proposed solution, explored in depth by hand analysis at the workshop, involves 1) the systematic elimination of certain problematical constructions from the parse tree (resulting in trees that show a much higher degree of structural agreement) and 2) systematic restructuring of constituents to a minor degree for particular constructions if the grammar being evaluated differs from the evaluation standard for these constructions. The evaluation program itself carries out the elimination of constituents for both the standard parse and the parse being tested (hereafter the test or candidate parse, provided by the client grammarian for evaluation). The client is responsible for restructuring the special constructions in the test parse. These restructurings will be discussed after the evaluation procedure itself.

The proposed evaluation procedure has been implemented and is still undergoing analysis and modification, but generally, it has these characteristics: it judges a parse based only on the constituent boundaries it stipulates (and not the categories or features that may be assigned to these constituents); it compares the parse to a hand-parse of the same sentence from the University of Pennsylvania Treebank, (the standard parse); and it yields two principal measures for each parse submitted: Crossing Parentheses and Recall.

The procedure has three steps. For each parse to be evaluated:

- (1) erase all word-external punctuation and null categories from both the standard tree and the test tree; use the standard tree to identify and erase from both trees all instances of: auxiliaries, "not", pre-infinitival "to", and possessive endings ('s and ').
- (2) recursively eliminate from both trees all parenthesis pairs

- enclosing either a single constituent or word, or nothing at all;
- (3) using the nodes that remain, compute goodness scores (Crossing Parentheses, and Recall) for the input parse, by comparing its nodes to a similarly-reduced node set for the standard parse.

For example, for the Brown Corpus sentence:

*Miss Xydis was best when she did not need to be too probing.*

consider the candidate parse:

(S (NP-s (PNP (PNP Miss) (PNP Xydis)))  
 (VP (VPAST was) (ADJP (ADJ best)))  
 (S (COMP (WHADVVP (WHADV when)))  
 (NP-s (PRO she))  
 (VP (X (VPAST did) (NEG not) (V need))  
 (VP (X (X to) (V be)) (ADJP (ADV too) (ADJ probing))))))  
 (? (FIN .)))

After step-one erasures, this becomes:

(S (NP-s (PNP (PNP Miss) (PNP Xydis)))  
 (VP (VPAST was) (ADJP (ADJ best)))  
 (S (COMP (WHADVVP (WHADV when)))  
 (NP-s (PRO she))  
 (VP (X (VPAST ) (NEG ) (V need))  
 (VP (X (X ) (V be)) (ADJP (ADV too) (ADJ probing))))))  
 (? (FIN .)))

And after step-two erasures:

(S (NP-s Miss Xydis) (VP was best)  
 (S when she (VP need (V be (ADJP too probing))))))

The University of Pennsylvania Treebank output for this sentence, after steps one and two have been applied to it, is:

(S (S (NP Miss Xydis) (VP was best))  
 (SBAR when (S she (VP need (VP be (ADJP too probing))))))

Step three consists of comparing the candidate parse to the Treebank parse and deriving two scores: (1) The Crossing Parentheses score is the number of times the candidate parse has a structure such as ((A B) C) and the standard parse has one or more structures such as (A (B C)) which "cross" with the test parse structure. (2) The Recall score is the number of parenthesis pairs in the intersection of the candidate and treebank parses (T intersection C) divided by the number of parenthesis pairs in the treebank parse T, viz. (T intersection C) / T. This score provides an additional meas-

ure of the degree of fit between the standard and the candidate parses; in theory a Recall of 1 certifies a candidate parse as including all constituent boundaries that are considered essential to the analysis of the input sentence by the Treebank. (Treebank parses are in general underspecified because certain structures, such as compound nouns, are not bracketed.) For the above example sentence, there are no crossings and the recall is 7/9.

The last element of the proposed evaluation method involves the restructuring of trees by the client, which is necessary only if the parse submitted treats any of certain constructions in a manner different from the standard. At the workshop, three constructions were identified: extraposition, modification of noun phrases by post-head phrases such as PP, and sequences of prepositions which occur constituent-initially and/or particles which occur constituent-finally. Briefly, for extraposition sentences like *It is necessary for us to leave* the extaposed phrase *for us to leave* should be attached at the S level and not, for example as a sister of *necessary*. For NP modification, post-head modifiers should be attached to the NP and not at the N-BAR level. Finally, for sequences of prepositions/particles we attach to the top node of the constituent. Thus if the initial client analysis is

(We (were ((out of) (oatmeal cookies))))

then the restructured analysis should be

(We (were (out of (oatmeal cookies)))).

These three constructions were identified from a hand analysis of a limited amount of data and we are currently examining more data to see whether the list should be extended.

Generally, there are two strategies that can be followed in cases where a client's analysis differs systematically from the standard: modify the evaluation program so that it deletes certain nodes, or specify a procedure that can be adopted by clients to bring their trees into conformity with the standard. However, we have seen that there are instances where reconciliation is very difficult or impossible and are working to assess the expected frequency of such cases.

Before the evaluation software was available, we applied the method by hand, using the UPenn Treebank as a standard, to 14 of the above-mentioned 50 Brown Corpus sentences which were given their "ideal" analyses by the grammarians. (Canonical modifications as specified above were required.) The sentences were selected because they had been successfully run by one of our automated systems (NYU's) and were expected to give some hint of the method's reliability for sentences that are easy for automated systems. The Crossing score was zero in every case and the corresponding Recall average score was 94%. We were encouraged by this initial result to pursue the development of software to carry out the scoring.



After the evaluation program became available, we ran it on the entire 50 sentence corpus and obtained the following results:

	crossings		recall
AT&T	3	(1)	.88
BBN	4	(1)	.86
Bellcore	10	(5)	.87
Boeing	4	(1)	.97
HP	4	(0)	.97
IBM	4	(2)	.96
Logos	3	(0)	.86
NYU	10	(10)	.79

The first number in the crossings column is the total number of sentences that contained a crossing while the second number in parentheses is the number of sentences with crossings that remain after certain policy changes are implemented in the standard parse and the node deletion protocol of the evaluation procedure.

There are several points to be made about the above data: We feel that the number of crossings initially obtained is unacceptably high and that changes in the standard bracketing procedures or changes in the deletion protocols need to be adopted. Second, the number of crossings obtained after a few suggested changes are implemented (the number in parentheses) is an acceptable level of crossings for a 50 sentence corpus for all but two of the grammars. However, until more data are examined, we will not know whether this level of crossings can be maintained with a fixed evaluation method. We are still in a "training phase" as far as the bracketing and deletion policies go and the actual level that will be attained may turn out to be less than is acceptable. The policy changes themselves are still being debated by the group. Finally, we note that two of the grammars (Bellcore's and NYU's) differ significantly from the others with respect to crossings. The Bellcore grammar is based on a new grammar methodology called "chunking" which results in non-standard phrasal groupings in some instances while the NYU grammar has significantly different in that it does not use any category corresponding to verb phrase, which results in non-standard attachments. It is unclear at this time whether convenient transformations can be found to allow these grammars to be compared to the standard so as to reduce their crossings scores.

There are four proposed changes to the evaluation method and the standard that are being debated at this time by our group. If the four policies below are adopted, then the crossing scores obtained are the ones in parentheses in the above table. The four policies are:

1) Delete left-recursive subnodes of type S from the standard. The Treebank uses recursive attachment at the S level for adverbial attachment in sentences like

*Miss Xydis was best when she did not need to be too probing*  
 which results in a structure of the form (S (S (A ..)(B...))(C ...)). Several of us preferred to attach the rightmost constituent (the 'when' phrase) at a lower level. With a

structure of the form (S (A ...)(B ...)(C ...)) all of the crossings are eliminated from our data. This policy can be implemented by the evaluation program.

2) Flatten structures in the standard containing the collocations *less than*, *more than*, *greater than*, etc. when they precede a number or adjective. Some of us take these collocations as constituents (under a certain reading of the sentence) while others always build phrases with *than* and phrases to its right before combining with *less*, *more*, etc. The lack of agreement among practitioners can be accommodated only if the standard is neutral. So the phrase *more than 4000,000,000 inhabitants* would need to be bracketed something like

(NP (ADVP more than 400,000,000) inhabitants),

The same requirement would also be imposed for phrases such as *more than likely*.

3) Flatten certain common sequences involving preposition, noun, preposition such as *in light of* and *in violation of*. Here again, there is a diversity of practice in our group as to whether the preposition, noun, preposition sequence is treated as a multi-word preposition or has NP and PP structures built between the words, as exemplified by:

(PP in (NP light (PP of (NP his success))))

A neutral bracketing of this phrase is

(PP in light of (NP his success))

4) Delete copular *be* when it precedes an adjective. A phrase such as *is happy to leave* would receive both of the following bracketings in our data: ((is happy)(to leave)) and (is (happy (to leave))). The deletion policy will eliminate any crossings for this type of phrase.

Even with these additional policies, there is still a residual set of eight sentences with crossings for some of the grammars (excluding, for the sake of brevity, some sentences for which the NYU grammar has crossings). We present here the eight sentences along with a discussion of the differences in analysis that led to the crossings:

1. *The petition listed the mayor's occupation as attorney and his age as 71.*

The standard analyzes this by coordinating *listed ... as attorney* with *his age as 71*. (The second coordinate is taken to be a verb phrase with an ellipted verb.) One of us prefers an analysis in which *the mayor's occupation as attorney* and *his age as 71* are treated as the coordinated constituents, creating a phrase crossing with the first coordinate phrase of the standard.

2. *His political career goes back to his election to city council in 1923.*

The standard analysis makes a constituent out of *back to ... 1923* while one of our analyses postulates *goes back* as a constituent.

3. *All Dallas members voted with Roberts, except Rep. Bill Jones, who was absent.*

The standard attaches non-restrictive relative clauses to NP. In this case *who was*

*absent* is attached to *Rep. Bill Jones*. Two of us attach non-restrictive relative clauses at the sentential level.

4. *The odds favor a special session, more than likely early in the year.*

The standard attaches *more than likely early in the year* to the NP associated with *session* while some of us attach it higher.

5. *The year will probably start out with segregation still the most troublesome issue.*

One of our grammars attaches the adverb *probably* at a low level to the verb *start* while that standard associates it with the S and specifies a verb phrase from *start* to the end of the sentence, which produces a crossing.

6. *The dinner is sponsored by organized labor and is scheduled for 7 p.m.*

The standard coordinates *is sponsored by organized labor* and *is scheduled for 7 p.m.* while another analysis coordinates *the dinner is sponsored by organized labor* with *is scheduled for 7 p.m.*

7. *He is willing to sell it just to get it off his hands.*

There is significant disagreement in our group over how to attach the phrase *just to get it off his hands*. The standard attaches it under the root S, while others attach it variously to phrases beginning with *is willing*, *willing*, and *sell*. (A recursive attachment to *is willing to sell it* would not produce a crossing with the standard.)

8. *Mr. Reama, far from really being retired, is engaged in industrial relations counseling.*

The standard takes *far* as an adverb that subcategorizes a PP, while one of our grammars treats *far from* as a multi-word lexical item.

In conclusion, we believe that the degree of disagreement that remains after the application of our deletion and restructuring method does not pose a significant barrier to the use of hand bracketed corpora for evaluation purposes for most of our grammars. However, the amount of data that we have been able to examine so far is limited and our judgements about the success of the method are still tentative. We will continue with our hand analyses, but also start to use the evaluation program with the real output of our parsers in a realistic test of the complete evaluation methodology. We invite other groups to participate and will make our evaluation software (which runs in Common Lisp) available.

# A Diagnostic Tool for German Syntax

John Nerbonne, Klaus Netter,  
Abdel Kader Diagne, Ludwig Dickmann and Judith Klein \*

Deutsches Forschungszentrum für künstliche Intelligenz  
and Institut für Computerlinguistik  
Universität des Saarlandes  
Stuhlsatzenhausweg 3  
D-6600 Saarbrücken 11, Germany  
nerbonne@dfki.uni-sb.de

## 1 Introduction

This paper describes an effort to construct a catalogue of syntactic data which is intended eventually to exemplify the major syntactic patterns of the German language. Our purpose in developing the catalogue and related facilities is to obtain an empirical basis for diagnosing errors in natural language processing systems analyzing German syntax, but the catalogue may also be of interest to theoretical syntacticians and to researchers in speech and related areas. The data collection differs from most related enterprises in two respects: (i) the material consists of systematically and artificially constructed sentences rather than naturally occurring text, and (ii) the material is annotated with information about the syntactic phenomena illustrated, which goes beyond tagging parts of speech. The catalogue currently treats verb government, (including reflexive verbs and verbal prefixation) and coordination.

The data consists of linguistic expressions (mostly short sentences designed to exemplify one syntactic phenomenon) together with annotations describing selected syntactic properties of the expression. The annotations of the linguistic material serve (i) to classify construction types in order to allow selected systematic testing

---

\*This work was undertaken with financial support from the German Ministry for Research and Technology (BMFT) and the LLOG project of the IBM Corporation

of specific areas of syntax, e.g., coordination; and (ii) to provide a linguistic knowledge base supporting the research and development of natural language processing (NLP) systems. Besides classificatory information, the annotations contain information about the precise structure of the sentence such as the position of the finite verb and the positions of other phrases.

In order to probe the accuracy of NLP systems, especially the detection of unwanted overgeneration, the test material includes not only genuine sentences, but also some syntactically ill-formed strings.

The syntactic material, together with its annotations is being organized into a relational database in order to ease access, maintain consistency, and allow variable logical views of the data. The database system is in the public domain and is (mostly) independently supported.

Our intent is to make public this work—both the test material and the database of annotations. We plan to share this work first with selected contributing partners, and later with the general research and development community.

## 2 Goals of a Diagnostics Tool

Our goal in collecting and annotating syntactic material is to develop a diagnostic tool for natural language processing systems, but we believe the material may be of interest to other researchers in natural language, particularly syntactic theoreticians. Finally, although this is not an evaluation tool by itself, our work points to possibilities for evaluating systems of syntactic analysis by allowing the systematic verification of claims about, and investigation of, the coverage and precision of systems.

### 2.1 Natural Language Processing

There is general consensus, both in theoretical computational linguistics and in practical, industrially sponsored research in natural language processing, that systems for syntactic analysis (parsing, recognition and classification) are possible and valuable. The applications of syntactic analysis currently under investigation include grammar and style checking; machine translation; natural language understanding (particularly interfaces to databases, expert systems, and other software systems); information retrieval; speech synthesis; and speech recognition. The potential impact of syntactic analysis technology is technically and financially profound.

But if we are to realize the full benefits of syntactic analysis, then we must ensure that correct analyses are provided. The development of a diagnostic tool serves just this purpose—pointing out where analyses are correct, and where incorrect. There are, of course, other measures of quality which apply to natural language software, e.g. general software standards. Systems which perform syntactic analysis are naturally subject to the same general standards of software quality that are imposed throughout the software engineering field, e.g., efficiency, modularity, modifiability, compatibility, and ease of installation and maintenance. Special-purpose systems may be subject to further standards; e.g., interface software is generally required to have clear and intuitive boundaries (transparency). Compared to such general software standards, correctness of syntactic analysis is an orthogonal criterion, though for many applications, an overriding one. Attending exclusively to general software standards means risking incorrectness—whether this be incorrectness of matrix multiplication in a linear algebra package or misanalyses in a natural language parser. The ultimate costs of such misanalysis depend, of course, on the particular application, but these costs may easily outweigh the benefits of the system deployed.

The importance of precision in syntactic analysis is occasionally disputed. It is pointed out, for example, that humans make speech errors (and typos), and that natural language understanding systems will have to be sufficiently robust to deal with these. Here, it is claimed, less precise systems may even have an advantage over more exact, and hence “brittle” competitors. What is correct about this point is that systems should be able to deal with ill-formed input. What is questionable is the suggestion that one deal with it by relaxing syntactic or other constraints *generally* (although it might be quite reasonable to use constraint relaxation where no exact analysis may be found—as a processing strategy).

The problem with general constraint relaxation is that it inevitably involves not only providing analyses for ill-formed input (as intended), but also providing additional incorrect analyses for well-formed input—“spurious ambiguity”. To see this, consider agreement, probably a good candidate for a less important “detail” of syntax which might safely be ignored. For example, it might be argued that sentence (1) below ought to be regarded as syntactically acceptable, since it’s clear enough what’s intended:

- (1) *Liste alle Sekretärinnen, die einen PC benutzt*  
List all secretaries who uses a PC

Syntactically tolerant systems would accept this sentence, but they would then have no way of distinguishing correct and incorrect parses of sentences such as (2), which are distinguished only by agreement:

- (2) *Liste jede Sekretärin in Finanzabteilungen, die einen PC benutzt*  
List every secretary in finance departments who uses a PC

The relative clause *die einen PC benutzt* can of course only be understood as modifying *jede Sekretärin* (the only NP with which it agrees), but a system which ignored agreement information would have no way of eliminating the parse in which the relative clause is construed as modifying *Finanzabteilungen*.

Furthermore, even if we accepted the argument that some applications may ignore syntactic accuracy, we are still faced with the applications at the other end of the spectrum of syntactic sensitivity, i.e., applications where syntactic accuracy is essential. Applications of this sort are found where the microstructure of the text plays an important role, e.g., grammar or style checking, and generally the entire area of NL generation: clearly, nobody wants a system which over-generates in synthesis. Similarly it is hard to find any advantage for underconstrained systems in applications such as speech understanding, where the whole point of using syntactic information is to reduce the number of hypotheses—a goal served only by maximally constrained systems.

We therefore believe that syntactic precision is indispensable for some applications and valuable even in applications in which ill-formed input may be expected.

The diagnostic tool assesses correctness of syntactic analysis—it supports the recognition of bugs in the linguistic analysis. This in turn provides both a means of assessing the effects of proposed changes in syntactic analysis as well as a means of tracking progress in system coverage over time. Neither of these derivative tasks is realistically feasible without the aid of an automated tool. Humans may spot individual errors when attending propitiously, but we're poor at systematic checks and comparisons, especially in large systems created by groups over relatively long periods of time.

## 2.2 Linguistic Research

This is an appropriate point at which to acknowledge our own debt to descriptive and theoretical linguistics, from which our primary data—the German sentences themselves—have been gathered. We expect to reciprocate, i.e., we expect that descriptive linguistics and even linguistic theory may benefit from the data collection effort we have undertaken. These benefits may take different forms: first, we have begun gathering the data in a single place; second, we are organizing it into a database in a fairly general way, i.e. with relatively little theoretical prejudice, so that variable perspectives on the data are enabled; third, in addition to relatively crude data analysis routinely provided in linguistic data collections—which seldom extends beyond marking ill-formedness/well-formedness, we have provided further fundamental data annotations. Fourth, and most intriguingly, the time may not be distant when linguistic hypotheses may be tested directly on

the computer. Many contemporary computational systems for natural language syntax are based on ideas of current interest in theoretical linguistics as well, and there is interest in general machinery for implementing syntactic analysis for wide varieties of linguistic theories. At that point, the use of diagnostic tools will be of immediate interest in linguistic research as well.

In sketching these potential benefits of the general data collection and analysis effort we have begun, it should be clear that we don't intend to speak only to linguists exploring "corpus-based" methodologies: our information includes facts about the ill-formedness of strings as well as rudimentary data analysis. This will become clearer below.

### 2.3 Toward Evaluation

The catalogue of syntactic material we have collated is intended for deployment in diagnosis—the recognition and characterization of problems in syntactic analysis. This is a task different from general system evaluation, which in most cases will judge the performance of a system relative to the achievement of a goal which is set by an application. Even if we limit evaluation to the performance of the syntactic component of a system, there are still some differences which have to be kept in mind.

The contrast between diagnosis and evaluation can be appreciated if one considers the case of applying our diagnostic tool to two different systems. In virtually every case, the result we obtain will show that neither system is perfect (nor perfectly incorrect), and that neither one analyzes exactly a subset of the constructions of the other. Suppose, for the sake of illustration, that one system is superior in treating long-distance (multi-clausal) dependencies, while the other is better at simple clause structure, but that the performance of the two systems is otherwise the same. The diagnosis is complete, but the evaluation still needs to determine the relative importance of the areas in which coverage diverged.<sup>1</sup> If matters were always as simple as in this illustration, we might appeal to a consensus of informed opinion, which would in this case certainly regard the treatment of simple clause structure as more important than that of long-distance dependencies—and would therefore evaluate the systems accordingly. But matters need not and normally are not so simple at all. There simply is not a consensus of informed opinion about the relative importance of various areas of grammatical coverage.

---

<sup>1</sup>Strictly speaking, this is not necessary; we could evaluate all such cases as equally proficient, but (i) the results of such "evaluation" would be too coarse to be of much use; and (ii) this simply goes against good sense. Some areas of grammatical coverage simply are more important than others. See the example in text, where simple clause structure is certainly more important than long-distance (multi-clausal) dependency.



Some crucial information that is lacking from our catalogue of syntactic material is information about relative frequency of occurrence. If this information could be obtained and added to the database, then it should be possible to develop an evaluation system of sorts from our diagnosis system.<sup>2</sup>

### 3 The Diagnosis Facility

We include here a brief description of the diagnostic facility; more detailed documentation, especially for the various areas of coverage of the syntactic catalogue, is currently under preparation.

#### 3.1 Sentence Suite

As noted in the introduction, our material consists of sentences we have carefully constructed to illustrate syntactic phenomena; we have not attempted to collect examples from naturally occurring text. Several considerations weighed in favor of using the the artificially constructed data:

- since the aims are error detection, support of system development, and evaluation of systematic coverage, we need optimal control over the test data. Clearly, it is easier to construct data than to collect it naturally when we have to examine (i) a systematic range of phenomena or (ii) very specific combinations of phenomena.
- we wished to include negative (ill-formedness) data in order to test more precisely (cf. discussion in Section 2.1 on "spurious ambiguity" and also on the needs of generation). Negative data is not available naturally.
- we wished to keep the diagnostic facility small in vocabulary. This is desirable if we are to diagnose errors in a range of systems. The vocabulary used in the diagnostic tool must either (i) be found in the system already, or (ii) be added to it easily. But then the vocabulary must be limited.
- we wished to exploit existing collections of data in descriptive and theoretical linguistics. These are virtually all constructed examples, not naturally occurring text.
- data construction in linguistics is analogous to the control in experimental fields—it allows the testing of maximally precise hypotheses.

---

<sup>2</sup>But it is not clear that this is the best way to go about developing an evaluation system. For example, we are not making any effort to keep some of the material secret, as speech evaluation systems routinely do in order to prevent a bias toward test material.

We have no objection to including naturally occurring data in the catalogue, subject to the restrictions above (especially constraining the size of the facility).

The vocabulary for the test suite has been taken from the domain of personnel management wherever possible. We chose this domain because it is popular in natural language processing, both as a textbook example and as an industrial test case. The domain of personnel management would also be useful in case we are to diagnose errors in semantics as well as syntax (which we are not attempting to do at present, but which is an interesting prospect for the future). It presents a reasonably constrained and accessible semantic domain. Where no suitable vocabulary from the domain of personnel management presented itself, we have extended the vocabulary in *ad hoc* ways.

The suite of test sentences is being collated by various contributors, each specializing in a single area of coverage, e.g. verb government, coordination, or NP constructions. Because of the range of syntactic material which is eventually to be included, it is difficult to draw precise guidelines about the sentences.

Still, several factors have been borne in mind while constructing the syntactic examples.

- lexicon size (cf. above)
- adherence to the following standards: (somewhat) formal, conversational High German; i.e., we have avoided colloquialisms, literary peculiarities, and regional dialects.
- selected testing of negative examples. We have tried to keep the catalogue small, but not at the cost of using great ingenuity to create minimal sets of testing data, nor at the cost of introducing very unnatural examples into the test catalogue. We have not rigorously purged superfluous examples.
- minimization of irrelevant ambiguity (bearing in mind that it cannot be fully eliminated).
- attention to analytical problems. We have attempted to catalogue not only the constructions, but also the problems known to be difficult in their analysis.

We do not deceive ourselves about our chances for success with respect to the last point: our catalogue is doubtlessly incomplete in many respects, but most sorely in this one. We invite comment and contribution everywhere, but most especially in further cataloguing the known analytical problems in German syntax.

In stressing our intention to catalogue analytical problems as well as the basic range of syntactic construction types, we do not intend to suggest that we intend to gather a collection of "cute examples". We will gather cute examples,

but these are relatively few in the general catalogue. Our primary goal will be a coverage of phenomena which is as comprehensive as feasible, even if this involves the rather tedious compilation of theoretically relatively well-explored and scientifically "uninteresting" constructions, such as the full paradigms illustrating determiner-adjective-noun agreement in German or the different types of verbal subcategorization. From our experience, it is above all the absence of systematic and comprehensive test-beds which hampers system development, rather than the lack of ingenious examples (which frustrate all systems in some way or other). Our goal is thus not primarily to show what systems cannot do, but to support the extension of what they can do.

### 3.2 Syntactic Annotations

In choosing which annotations about the sentences might be sensible, we have been guided by two considerations. First, the catalogue will be much more useful if examples from *selected* areas can be provided on demand. For example, it would be useful to be able to ask for examples of coordination involving ditransitive verbs—as opposed to simply coordination (an area of coverage). This means that we need to provide annotations about which area of coverage a given sentence (or ill-formed string) is intended to illustrate. With regard to these annotations, we have merely attempted to use standard (traditional) linguistic terminology.

Second, we can exploit some annotations to check further on precision of analysis. This is the purpose of annotations such as:

- well-formed vs. ill-formed
- position of finite matrix verb
- position of NP's
- position of PP's

So, in a sentence such as (3), the following database values are encoded:

- (3) *Der Student bittet den Manager um den Vertrag.*  
 the student asks the manager for the contract

*/OK	OK
finite matrix verb	3
position of NP's	1-2, 4-5, 7-8
position of PP's	6-8

In selecting these properties as worthy of annotation, we were motivated primarily by a wish to focus on properties about which there would be little theoretical dispute, which would be relatively easy to test, and which would still provide a reasonable reflection of a system's accuracy.

### 3.3 An Example: Verbal Government

One of the phenomena which the data collection already covers is the area of verbal government, i.e., verbal subcategorization frames. The aim was to compile a comprehensive list of combinations of obligatory arguments of verbs, forming the basis of different sentence patterns in German. We ignore both adjuncts and optional arguments in restricting ourselves to obligatory arguments, which can be tested by an operationalizable criterion, a specific sort of right extraposition:

- (4) *Er hat gegessen, und zwar Bohnen.*  
he has eaten, namely beans.
- (5) *\*Er hat verzehrt, und zwar Bohnen.*  
he has consumed, namely beans
- (6) *\*Er hat das Buch gelegt, und zwar auf den Tisch.*  
he has put the book, namely on the table
- (7) *Er hat Maria geküßt, und zwar auf die Wange.*  
he has kissed Mary, namely on the cheek

We attempted to find instances of all possible combinations of nominal, prepositional, sentential, but also adjectival complements.<sup>3</sup> Clearly, we could not immediately cover the entire field in full depth, so that we decided to adopt a breadth first strategy, e.g., we ignored the more finegrained distinctions to be made in the area of infinitival complementation or expletive complements. The description in these areas will be elaborated at later stages.

The result of the collection is a list of about 90 combinations which are exemplified in about 300 sample sentences.

The sentences illustrate

- combinations of nominal, prepositional and adjectival arguments, viz.,
  - nominal arguments only:

---

<sup>3</sup>At the basis of our list were collections to be found in the literature, such as [2], [5], [6], [7], [9] and [12]. We are also grateful to Stefanie Schachtl, Siemens Munich, who provided us with some of her material.

(8) *Der Manager gibt dem Studenten den Computer.*  
the manager gives the student the computer

- nominal and prepositional arguments with semantically empty (9) or non-empty prepositions (10):

(9) *Der Vorschlag bringt den Studenten auf den Lösungsweg.*  
the suggestion takes the student to the solution

(10) *Der Manager vermutet den Studenten in dem Saal.*  
the manager assumes the student in the hall

- nominal and adjectival (or predicative) complements

(11) *Der Manager wird krank.*  
the manager becomes ill

→ nominal arguments combined with finite (subordinate) clauses, introduced by the complementizers *daß* (12), *ob* (13) or some wh-element (14):

(12) *Daß der Student kommt, stimmt.*  
that the student comes, is-correct

(13) *Dem Manager entfällt, ob der Student kommt.*  
it escapes the manager, whether the student comes

(14) *Der Manager fragt, wer kommt.*  
the manager asks who comes

• nominal arguments in combination with infinitival complements, illustrating bare infinitives (15) and *zu*-infinitives (16):

(15) *Der Manager hört den Studenten kommen.*  
the manager hears the student come

(16) *Der Manager behauptet, den Studenten zu kennen.*  
the manager claims to know the student

• examples involving some of the combination above in connection with expletive or correlative prepositional pronouns or expletive *es*:

(17) *Der Vorschlag dient dazu, den Plan zu erklären.*  
the proposal serves (to-it) to explain the plan

(18) *Der Manager achtet darauf, ob der Student kommt.*  
the manager checks (on-it) whether the student comes

(19) *Es gelingt dem Studenten, zu kommen.*  
it succeeds to the student, to escape  
"The student succeeds in escaping"

(20) *Der Manager hält es für notwendig, zu kommen.*  
the manager considers it (for) necessary to come

Since we are interested only in verbal government here, we tried to keep as many other parameters as possible carefully under control: as already mentioned, the noun phrases in the sample sentences are built from a limited vocabulary. All noun phrase and prepositional complements have a definite determiner. In the case of prepositional phrases the fusion of preposition and determiner (*in dem* → *im*) is avoided. Since German has relatively free word order, the different complements have to be identified by their case marking in most cases—as a consequence, morphological ambiguities of case (e.g. between feminine or neuter nominative and accusative) were excluded. The matrix and subordinate clauses all have only one verbal head (i.e., they do not have any auxiliary or modal verbs), whose morphological form is the third person, singular, present, indicative form if possible. The sentences do not contain any additional irrelevant modifiers, adjuncts or particles. The word order of the sample sentences is meant to illustrate the “un-marked” order, although this should not play an important role, since the complements are uniquely case marked, as mentioned.

Every combination of complements is illustrated by at least one example. In addition, each sentence is paired with a set of ill-formed sentences, which illustrate three types of errors relevant for verbal government:

- an obligatory argument is missing;
- there is one argument too many;
- one of the arguments has the wrong form.

The material is organized in a relational database, such that queries can ask either for a description or classification of a sentence or for sentences matching combinations of descriptive parameters.

In describing the argument structure of the sentences we chose a vocabulary which is of course not theory neutral, but which at least can be expected to meet common agreement. We tried to avoid theory-specific notions such as *subject* or *direct object*, and identified the complements on the basis of morphological case marking, prepositions, complementizers and/or the morphology of the verb. Obviously, this vocabulary cannot exhaustively characterize the properties of individual complements. For example, with those few verbs which subcategorizes for two accusative NPs it is quite unlikely that they both NPs behave in the same way with respect to passivization. Similarly, a nominative complement (“subject”) may have different properties depending on the verb being un-accusative or un-ergative. However, we think that distinctions of this kind should be dealt with separately in data sets on e.g. passivization, ergativity, etc.

## 3.4 Database

### 3.4.1 Abstract Data Model

In addition to the relatively straightforward properties of sentences noted above (Section 3.2), we also model the more complex classificatory information in the catalogue.

According to the Entity-Relationship (ER) terminology (cf. [3]), we can identify two entity types and one relationship type which are specified as follows:

1. **SENTENCE**, an entity type, the major concept of the data model. An entity of this type includes a description of the main verb's valency (i.e., the number of arguments the main matrix verb governs and their description), a sentence which exemplifies the given properties, and information on its wellformedness. Each entity has a unique identifier, a key attribute which facilitates queries for description or classification of a sentence. (Given the present limited range of data and the underlying area (verb government), the attributes argument-description and fin-matrix-verb could almost be used to identify a sentence entity uniquely, because there is only one representative from most valency types in SENTENCE. But some types are represented more than once.)
2. **CATEGORY**, an entity type. Each entity of this type (e.g., NP, finite\_matrix\_verb) represents a category which appears in a related sentence.
3. **APPEARS\_IN**, a M:N relationship type<sup>4</sup> between **CATEGORY** and **SENTENCE**. Both **CATEGORY** and **SENTENCE** participations in the relation are total. **APPEARS\_IN** has additional attributes specifying the position of a given category in a related sentence and its lexical form.

The following figure illustrates the conceptual model of the database described above. It covers the area of verbal government and can be easily extended.

---

<sup>4</sup>M:N relation (many to many relation): a sentence entity may be related to (i.e. may include) numerous category entities, and a category entity may appear in numerous sentence entities.

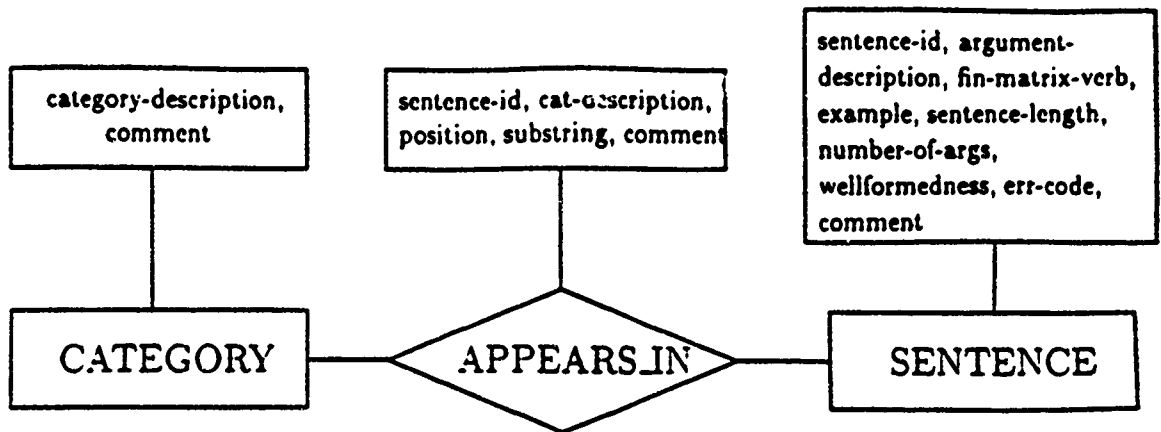


Figure 1: The ER schema diagram for the database described above.

The following example shows database entries for a given sentence.

- (21) *Der Vorschlag bringt den Studenten darauf, daß der Plan falsch ist.*  
 the suggestion takes the student to-it, that the plan is wrong

**SENTENCE**

s-id	arg-description	m-m-v	ex	sl	na	wf	err	com
1012	nom.acc.cor.sc.dass	bringen	(sl)	11	4	1	0	.

(s-id = sentence-id, m-m-v = fin-matrix-verb, ex = example, sl = sentence length, na = number of arguments, wf = wellformedness, err = error code, com = comment)

**CATEGORY**

category-description	comment
cor	correlate
fin-matrix-verb	
NP	
sc-comp	subordinate clause

**APPEARS\_IN**

sentence-id	category-description	pos-from	pos-to	substring
1012	cor	6	6	darauf
1012	fin-matrix-verb	3	3	bringt
1012	NP	1	2	der Vorschlag
1012	NP	4	5	den Studenten
1012	sc-comp	7	11	daß der Plan falsch ist.

A new database entry for a given sentence must include values for the attributes arg-description, fin-matrix-verb, example, wellformedness, category-description, pos-from, and pos-to. For ill-formed sentences the error code and additional comments should be given. All other attributes can be inserted through some triggers



including consistency checks. Splitting the position attribute into pos-from and pos-to makes the generation of the corresponding substrings possible and facilitates a consistency check (e.g., pos-from must be a positive integer number less than pos-to, pos-to must be greater than pos-from and equal or less than the sentence length.).

### 3.4.2 Database System

The database is administered in the programming language awk (cf. [1]). Some of the reasons which speak in favor of awk are:

- awk is in the public domain running under UNIX and should run in other environments; in particular, it runs on MS-DOS.
- Its ability to handle strings of characters as conveniently as most languages handle numbers makes it for our purposes more suitable than standard relational database systems; i.e., it allows more powerful data validation, increasing the availability of information with a minimal number of relations and attributes.

Compared to standard databases awk has a restricted area of application and does not provide fast access methods to information, but it is a good language for a developing a simple relational database in a number of cases. Additional resources and tools such as a report generator and a routine for consistency checking can be easily implemented.

The database includes a reduced sql-like query language. We use the database entries of the example given above to ask the following queries:

- (i) retrieve all sentences which include a correlate and a subordinate clause beginning with *daß*.

query: retrieve sentence-id, example  
from sentence  
where match(arg-description, "cor") and match(arg-description, "sc-dass")

result: 1012 der Vorschlag . . . falsch ist.

- (ii) retrieve the position and the lexical form of all NP's of sentence 1012.

query: retrieve cat-description, position, substring  
from sentence, appears.in  
where sentence-id = 1012 and category-description = "NP"

result: NP 1 2 der Vorschlag  
NP 4 5 den Studenten

The query language has been developed under SunOS using the utilities `lex` and `yacc`. `lex` is a lexical analyzer generator designed for processing of character input streams. `yacc`, a LALR(1) parser generator, is an acronym for Yet Another Compiler Compiler. It provides a general tool for describing an input language to a computer program.

### 3.5 Auxiliary Materials

The database of syntactic material is to be accompanied by a few auxiliary development tools. First, in order to support further development of the catalogue and database, it must be possible to obtain a list of words used (so that we minimize vocabulary size), and a list of differentiating concepts (so that categorization names may be accessed easily). Second, documentation must be available on each of the areas of syntactic coverage included. This is to cover (minimally) the delimitation of the area of coverage, the scheme of categorization, and the sources used to compile the catalogue.

Third, a small amount of auxiliary code may be supplied to support development of interfaces to parsers. This need not do more than dispatch sentences to the parser, and check for the correctness of results.

## 4 Comparison to Other Work

This appears to be the first attempt to construct a general diagnostic facility for German syntax, even if virtually every natural language processing group working on German has a small suite of sentences used for internal monitoring and debugging.

There have been several related efforts concerned with English syntax. Guida and Mauri [8] report on attempts to evaluate *system* performance for natural language processing systems (*n.b.*, not merely syntax) in which they attempt to finesse the issue of correctness (which we argue to be central) by measuring user satisfaction. We have attempted to address the issue of syntactic correctness head-on.

Hewlett-Packard Laboratories compiled a test suite of approximately 1,500 sentences which it distributed at the *Public Forum on Evaluating Natural Language Systems* at the 1987 Meeting of the Association for Computational Linguistics [4]. That effort differed from the present one in that it tried to evaluate semantics and pragmatics, as well as syntax, and in that it consisted essentially of sentences without annotated properties. The sentences were not organized into a database.

Read et al. [11] advocate a "sourcebook" approach, in which fewer examples are submitted to much closer scrutiny. The closer scrutiny doesn't seem subject to automation, at least at present. Furthermore, their emphasis is on evaluating *systems* for natural language understanding, and the primary focus seems to be on domain modeling, conceptual analysis and inferential capabilities, not syntax. It is similar to the HP approach (and to ours) in employing primarily constructed examples, rather than naturally occurring ones.

The Natural Language group at Bolt, Beranek, and Newman Systems and Technologies Corporation circulated a corpus of approximately 3,200 sample database queries formulated in English at the 1989 DARPA Workshop on Evaluating Natural Language [10]. The emphasis here, too, was on system (natural language understanding) performance, rather than specializations, but most of their examples seem to come from actual trial use of a natural language interface program, which gives their work added value.

The University of Pennsylvania's "Treebank" project (similar to a project of the same name at the University of Lancaster sponsored by IBM) has begun an effort to annotate naturally occurring text and speech, and to organize the annotations into a "Treebank". The annotations are phonetic, syntactic, semantic and pragmatic, and the intended scope is monumental. Since they wish to gather representative and varied data, they hope to collect and annotate approximately  $10^8$  words.

Finally, the Text-Encoding Initiative of the Association for Computational Linguistics is a loosely organized confederation of efforts concerned with the classification and annotation of various sorts of texts. Our work will be made available to this group.

## 5 Current State, Future Plans

### 5.1 Collaborations

We have contacted some research groups in the area of NLP and machine translation, which have shown interest in cooperating on the effort by submitting data

sets in exchange for the use of the database. Among these are the Institut für angewandte Informationswissenschaft (IAI), Saarbrücken and a research and development group at Siemens, Munich.

## 5.2 Eventual Range of Syntax Catalogue

As mentioned, we regard our work only as a starting point which has to be complemented by contributions from other groups and individual experts. As to extensions of the database, we can only provide the roughest of lists here. We intend the list to be suggestive rather than definitive:

Syntax of the simple clause, including verbal government and *genera verbi* (passive, etc.), negation, word order, and adverbial modification, including temporal adverbials (duratives, frequentatives, and "frame" adverbials), locative, manner, and measure adverbials. Verb phrase complementation including argument sharing or inheritance (*auf Hans ist er stolz*), clause union, extraposition, modal and auxiliary verbs. Verbal complex, fixed verbal structures (*Funktionsverbgefüge*), separable prefix verbs, idioms and special constructions.

Noun phrase syntax, including determiner and numeral (and measure) system, relative clauses of various sorts (including preposed participial phrases), pre- and postnominal adjectival modification, noun phrase coordination, and plurals. Pronominal system and anaphora.

Prepositions and postpositions, cliticization, particles (e.g., *als, ja, je, denn*).

Questions, including long-distance (multi-clause) dependence. Imperative and subjunctive moods. Adjectival and nominal government, modification, and specification. Equative, comparative, and superlative constructions. Coordination and ellipsis.

## References

- [1] A.V. Aho, B.W. Kernighan and P.J. Weinberger: *The awk programming language*. Wokingham et al., Addison Wesley, 1988
- [2] Peter Colliander: *Das Korrelat und die obligatorische Extraposition*. Kopenhagener Beiträge zur Germanistischen Linguistik. Sonderband 2. Kopenhagen, 1983.
- [3] P. Chen: The entity-relationship model. Toward a unified view of data. *ACM Transactions on Database Systems*. No. 1, 1976.

- [4] Daniel Flickinger, John Nerbonne, Ivan Sag, and Thomas Wasow: Towards evaluation of natural language processing systems. Technical report, Hewlett-Packard Laboratories, 1987.
- [5] Ulrich Engel: Die deutschen Satzbaupläne. In *Wirkendes Wort* 20, pages 361-392, 1970.
- [6] Bernhard Engelen: *Untersuchungen zu Satzbauplan und Wortfeld in der geschriebenen deutschen Sprache der Gegenwart*. Reihe I 3.3 Verblisten. München, 1975.
- [7] Lutz Götze: *Valenzstrukturen deutscher Verben und Adjektive*. München, 1979.
- [8] Giovanni Guida and Giancarlo Mauri: Evaluation of natural language processing systems: Issues and approaches. *Proceedings of the IEEE*, 74(7):1026-1035, 1986.
- [9] Gerhard Helbig: *Wörterbuch zur Valenz und Distribution deutscher Verben*. Leipzig, 5th ed., 1980.
- [10] Martha Palmer, Tim Finin, and Sharon M. Walter: Workshop on the evaluation of natural language processing systems. Technical Report RADC-TR-89-302, Rome Air Development Center, Air Force Systems Command, Griffiss Air Force Base, 1989.
- [11] Walter Read, Alex Quilici, John Reeves, Michael Dyer, and Eva Baker: Evaluating natural language systems: A sourcebook approach. In *COLING '88*, pages 530-534, 1988.
- [12] Monika Weisgerber: *Valenz und Kongruenzbeziehungen*. Frankfurt a. M., 1983.

## PRELIMINARIES TO THE DEVELOPMENT OF EVALUATION METRICS FOR NATURAL LANGUAGE SEMANTIC AND PRAGMATIC ANALYSIS SYSTEMS

James E. Hoard, MS 7-64  
Research and Technology  
Boeing Computer Services  
P.O. Box 24346  
Seattle, WA 98124

e-mail: [jhoard@atc.boeing.com](mailto:jhoard@atc.boeing.com)  
telephone: 206 865-3262

Categorizing the underlying capabilities of natural language processing (NLP) systems is a required preliminary step in the development of test suites for semantic and pragmatic phenomena and of evaluation metrics that rate systems over the results achieved against the benchmark test suites.

To get beyond black-box evaluation of NLP systems and make some progress toward glass-box evaluation, especially in the semantics and pragmatics areas, does not depend, it seems to me, on an overarching agreement on exactly how to characterize semantic and pragmatic phenomena or on being able to map from one representation to another or into a presumed superset "neutral" representation. Rather, if we acknowledge that a fundamental task for any NLP system is to resolve ambiguity, then we can direct much of our attention there, acknowledging, of course, the need to assess additional inferencing requirements as well as performance characteristics.

For instance, suppose we agree that "the girl saw the boy on the hill with a telescope" is five ways ambiguous (has five distinct meanings). Then, there ought to be five distinct semantic representations provided by any NLP system that can handle the phenomenon. Note that, for semantic evaluation, it is not necessary to require that such an NLP system also provide five distinct syntactic representations (presumably revealing different prepositional phrase attachments). Whatever syntactic analyses the system provides are simply irrelevant to our purpose. Additionally, in a context that pragmatically forces one of the semantic interpretations, an NLP system that is pragmatically adequate in this area ought to make the right choice. To evaluate a system's capability to account for this sort of ambiguity, we do not need to evaluate precisely how the semantic representations are achieved or what their precise formulation happens to be. Nor do we need to know exactly how a system selects the correct interpretation in an appropriate context. It is enough to know ("check off") that all five interpretations are available and that the correct choice is made.

Test suites will, of course, necessarily exemplify a wide range of phenomena at both the sentence and discourse levels. Those involving ambiguity resolution per se include word ambiguity, structural ambiguity, anaphora and definite reference resolution, and quantifier and negation ambiguity (scoping). Additional evaluation criteria, which

largely go beyond ambiguity resolution as normally defined, include, for example, how well a system establishes discourse relations (causal and temporal, among others), how well a system draws inferences (lexical and structural, both guaranteed and invited), and how fast and how well a system arrives at its results (both time and space complexity are relevant). I would maintain that developing evaluation metrics, and the test suites themselves, depends on establishing a valid categorization of NLP systems that is independent of the tests themselves.

The categorization of NLP systems according to their capabilities, both actual and projected, will afford insight in several ways. First, it will instruct us primarily on what sorts of test suites to construct and on what evaluation measures to use. Second, it will tell us a great deal about the purposes of the various systems that are being developed by the computational linguistics community. Some are much more limited in their intended uses than others; some are, however, very ambitious in their aims. Third, it will give us not only a means to evaluate systems comparatively, but to assess with some confidence any given system's inherent coverage and extensibility at any stage in its development. In short, we need to have benchmarks that are valid not just for fully developed systems, but a reliable means of gauging the progress that is being made by the NLP community and by particular groups within it along the development path toward robust NLP systems.

The fundamental considerations by which a system can be categorized comprise at least an understanding of the following differentiators.

1. **The intended meaning of the semantic representations.** What is the cognitive and/or real-world validity that the representations are intended to model? Is the intended meaning implicit or explicit? For example, for the sentence *John loves Mary* is the representation rather like 'loves(John, Mary)', where the interpretation is external to the representation, or is it more like 'loves(cognizer:John, range: Mary)', where it is supposed that the interpretation of the roles that John and Mary play in the sentence is fixed ("cognizer" and "range", respectively)? The latter sort of semantic representation seems to carry with it assumptions about cognitive and/or real world validity. Moreover, for representations that attempt to make the intended interpretations explicit, we can entertain making up tests that assess the "self-awareness" of the system in the sense that we might suppose that such a system could answer questions about "how it knows what it knows".

2. **The closed- or open-world assumptions that are made and the values that the logic is capable of supporting.** (I construe the word logic here in its most general sense so as to include whatever reasoning principles a system supports.) Now, I take it, rightly or wrongly, that a system that has only a two-valued logic is a system that makes a closed-world assumption. For instance, in a system that has a Prolog-style logic, 'yes' means "demonstrable from the information at hand" while 'no' means "not demonstrably 'yes'". A system with a three-valued (or higher) logic will support an open-world assumption. Thus, given information that: *All men are mortal, Socrates is a man, and Jill is not a man*, a two-valued system will respond to the query *Is Jill mortal?* with 'no'; but it will also report 'no' to the query *Is Sam mortal?*, about

which we have, by hypothesis, no information. A three-valued system ought to respond with 'don't know', or the equivalent, to the last inquiry. Systems that support fuzzy or probabilistic logics might or might not make open-world assumptions. If they do, then it is fair to test them over such a suite of inquiries.

3. What kinds of valid conclusions can be drawn using the logic and reasoning system. It seems to me that we do not very much care from an evaluation standpoint whether a system uses deductive, inductive, or abductive inferencing strategies. What we do care about is the sorts of conclusions that are supported. Among the possibilities are: a) 'X means Y', b) 'X doesn't mean Y', c) 'X includes Y', and d) 'X is similar to Y'. Plausible examples of each of these putative inferencing axioms are a) *kill* 'means' *cause to die*, b) *kill* 'doesn't mean' *injure*, c) *kill* 'includes' *murder*, d) *convince* 'is similar to' *persuade*.

4. What finitary assumptions are made. The stability of a system under different test situations may cause us to exclude certain kinds of sentences or dialogues if queries are not guaranteed to terminate in finite time or with available storage (workspace).

5. What semantic primitives are assumed. Systems may or may not invest in such notions as "locative, temporal, purposive, agent, goal," and the like. For those that do, we might expect to test quite straightforwardly for such distinctions as the differing "roles" that *Bill* and *algebra* play in *John taught Bill* and *John taught algebra*, as well as, of course, *John taught Bill algebra*. For those systems that do not have "functional" primitives, we would ask about the existence of equivalent notions and/or mechanisms.

6. Whether the semantic representations are intended to be partial or complete descriptions of sentences (propositions, discourses, etc.) and whether the representations support both full and elliptical constructions. The questions at issue here involve the representation of sentences like *John is reading* versus *John is sleeping*. For the first, it is "understood" that John is reading something, while for the second it is equally well "understood" that there is no something that John is sleeping. The adequacy of semantic representations is clearly important to assessing the capabilities of an NLP system, although it is not obvious how this should be gotten at in a test suite that one would like to think is fairminded.

7. What kind of well-formedness conditions apply to the semantic representations. While double agents may be found in the spy business, we are reasonably confident that they do not occur as independent complements of a verb. Moreover, we are also reasonably sure that the limit on direct complements is three (syntactically, the subject, object, and indirect object). We should, therefore, test an NLP system to see whether the semantic representations exclude impossible semantic structures.

8. Whether an arbitrary or non-arbitrary mapping is assumed between syntactic and semantic representations. The issues here involve the granularity and completeness of the representations. Do the representations account for every morpheme in the input sentence? For every meaningful difference in word order? I.e., do the represen-



tations purport to be complete? If no current system does, it may be futile to develop test suites to evaluate such niceties as the difference in meaning of *They stopped to search for survivors* and *They stopped searching for survivors* let alone the subtle differences between *Reagan sent Iran a message* and *Reagan sent a message to Iran*. (This is not to say that it wouldn't be quite appropriate to test a system's ability to parse such sentences.) From this perspective it is clear that current state-of-the-art systems capabilities will both determine and limit the test suites.

**9. The kind(s) of knowledge on which pragmatic interpretations are based.** Is the system knowledge in lexicons and hierarchies, or is general encyclopedic knowledge available? The kinds of test suites one will develop depends very much on the scope and the extent of the knowledge bases available for reasoning. Then, too, it is pointless to evaluate system performance over metaphors and idioms if no system has any general way of dealing with these phenomena.

**10. The types of inferencing that are supported.** Especially of interest is whether a system supports the Gricean maxims that involve providing information that is close (or relevant) to a query. For instance, a system's knowledge bases (given initially or acquired on the fly via new text or message input) may contain the information that Max died. To the query *Did John kill Max?* can the system conclude that the information it has is likely to be relevant to the query? If a system "knows" that John convinced Bill of something or other, is it capable of noting the relevance of the query *Did John persuade Bill?*

**11. Whether reasoning about classes (i.e. higher-order rules) is available.** The issues here involve the abilities of a system to recognize and exploit equivalent and related phenomena by virtue of having mechanisms to relate semantic representations. For example, assume that all cleft sentences have unclefted versions. Then, a system might reasonably be expected to recognize that *It was John who closed the door* is a "good" fit to the query *Did John close the door?* and might, then, be expected to respond with 'yes' if the first sentence is known to be true.

**12. Whether the full range of natural language quantification is handled.** If no NLP system can deal with the full range of natural language quantification, then it is clearly pointless to include test suites that contain such sentences as *few tigers are tame* and *John frequently walks the dog*, expecting one or more systems to reason over such sentences given the information that twelve tigers are known to be tame or that John walks his dog three times a week except when he's out of town. Clearly, it would be better to stick with *All men are mortal* and the like if general quantification strategies are not included in the capabilities of an NLP system.

The above listing is not intended to be anything even remotely construable as a definitive statement about the range of things to be tested and evaluated in the semantics and pragmatics areas. Instead, it is intended to foster a discussion of the preliminaries that must be dealt with in order to construct glass-box test suites and metrics suitable for ranking a system's semantics and pragmatics analysis capabilities at any given stage of its development and to rank the performance of one system against

another. I do want to emphasize that it seems to me quite hopeless to entertain any notion of developing a universal representation scheme into which the representations of particular NLP systems might be translated. The aims, scope, and capabilities of NLP systems are just too varied for that. What does seem possible, however, is to develop test suites and evaluation metrics that follow more or less directly from such considerations as those given above. By keeping the evaluation at the higher, functional level, it seems to me that reliable and defensible semantic and pragmatic evaluation metrics can be fashioned. Furthermore, test suites and metrics will not be tied to specific domains or to specific application areas, but will measure general system capabilities, as desired.

# Corpus-Based Evaluation of the Sundial System

Norman M. Fraser

Social & Computer Sciences Research Group

University of Surrey

Guildford

Surrey GU2 5XH

United Kingdom

Tel: +44 483 509292

Fax: +44 483 300803

norman@soc.surrey.ac.uk

## 0. Introduction

The Sundial (Speech UNDERstanding and DIALogue) project is presently one of the largest speech and language technology projects in Europe<sup>1</sup>. It involves researchers from five countries in 170 person years of effort spread across five years. The aim of the project is to develop prototype spoken dialogue systems for limited domains (~2000 words) for each of English, French, German and Italian (Peckham, 1990).

As far as possible, common approaches to design and implementation have been adopted, with some software modules being used in all of the prototype systems. For example, an identical dialogue manager module is used in each of the prototypes (although naturally each prototype uses its own language-particular declarative knowledge base) (Bilange 1991; McGlashan *et al.* 1991). Figure 1 shows the overall architecture of the four Sundial systems.

The task of evaluating a system of this complexity is very difficult indeed, and it is a task about which very little is currently known or understood. In Sundial, we have tried to find a balance between principle and practicality. We believe that there is a close connection between the processes of system design and system evaluation. In Section 1, I outline the Sundial approach to design. In Section 2, I go on to show how data collected in the design phase can be used to good effect in objective system evaluation. Conclusions are drawn in Section 3, where I offer a key metric for dialogue systems. The main concern of the paper is with the evaluation of the *dialogue management* performance of the system, although this is only one of several foci for evaluation in the Sundial project.

---

<sup>1</sup>The work reported here was supported in part by the Commission of the European Communities as part of ESPRIT project P2218, Sundial. Partners in the project are Cap Gemini Innovation, CNET, IRISA (France), Daimler-Benz, Siemens, University of Erlangen (Germany), CSELT, Sarnel (Italy), Logica Cambridge, University of Surrey (U.K.). In addition, Politecnico di Torino (Italy) is an associate partner and Infovox (Sweden) is a subcontractor.

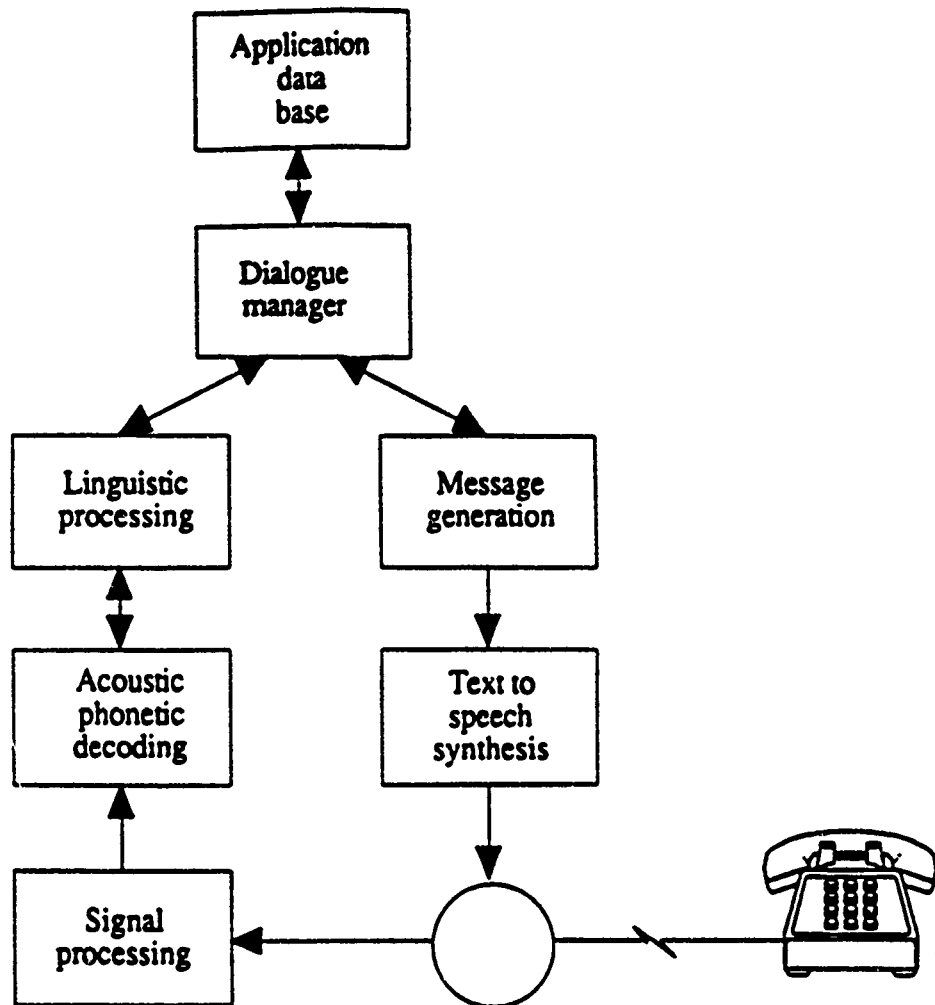


Figure 1: architecture of the Sundial system

### 1. Corpus-based design

Design of the Sundial system has been driven to a significant extent by the results of corpus analysis. Two different tasks are being implemented: flight enquiries and reservations (English and French); and train timetable enquiries (German and Italian). Initial corpora of human-human dialogues were collected using existing telephone information services. A representative sample of dialogues was used to create scenarios for Wizard of Oz (WOZ) simulation experiments in which subjects believed they were talking with an operational system when, in fact, they were really conversing with a human experimenter whose voice had been made to sound synthetic by filtering through a vocoder (Ponamale et al. 1990, Fraser and Gilbert 1991a). The English Sundial team collected a corpus of 100 human-human dialogues (the H-H corpus) and another corpus of 100 simulated human-computer dialogues (the H-C corpus). Since the dialogues in the H-C corpus were driven by scenarios derived from the H-H corpus, the two corpora were readily comparable.

The WOZ simulations revealed a number of important differences between human-

human dialogues and human-computer dialogues. For example:

- fewer words (tokens) are uttered in the H-C corpus;
- fewer distinct word forms (types) are used in the H-C corpus;
- ellipsis is virtually non-existent in the H-C corpus, although it is fairly common in the H-H corpus;
- there is a tendency to avoid the use of some syntactic constructions in the H-C corpus (e.g. there are 3-4 times as many relative clauses in the H-H corpus as there are in the H-C corpus);
- there are roughly the same number of instances of overlapping talk (when both speakers are talking simultaneously) in the whole H-C corpus as there are in one average dialogue in the H-H corpus (for further details see Fraser and Gilbert 1991b).

On the whole, the findings of the simulations are very encouraging, since they indicate that when speaking to a computer, speakers adapt their linguistic behaviour in ways which simplify the task of utterance interpretation. Some of the simulation findings are precisely those which might have been predicted, but others are much less likely to have been anticipated. An important principle underlying the Sundial project is that system design should be firmly rooted in the empirical evidence of the simulation corpora. How best to make the transition from data to design is an open research question, and one which deserves much more attention than it is presently receiving in the NLP community (Fraser *et al.* forthcoming). Clearly, a speech understanding system which modelled the *exact* behaviour found in the corpus and nothing else, would not be useful.

The approach favoured in the Sundial project is to make abstract descriptions of the corpus, and it is these descriptions which are modeled. Descriptions are made at a number of different levels. For example, at the level of word recognition the corpus is not nearly large enough to train the necessary speaker-independent word recognition subsystem. However, it does provide an abstract description of the words which have to be recognized. These words can then be collected elsewhere and used in training. At a higher level, the corpus is tagged for word classes, syntax and semantics. Moving up a level, the largest phrases (usually sentences, since ellipsis is virtually non-existent) are assigned a *dialogue act* label, indicating their function in the dialogue (e.g. as statements, questions, answers, etc). A *dialogue grammar* can then be built. This differs from a sentence grammar in having to take account of the turn structure of the dialogue (i.e. an important fact about question-answer exchanges is that answers are uttered by the speaker who did not utter the question). At another level, the unfolding goal structure of each dialogue is described. Dependencies between descriptions at different levels result in the production of equivalence classes. For example, a particular dialogue act may be realized by a number of different types of sentence. In this way generalizations over the corpus are produced.

The Sundial system has been designed to model the behaviour found at each level of analysis and to respect dependencies between levels. Each of the modules shown in Figure 1 has been implemented (except that the application database is currently being simulated within the Dialogue Manager). The first fully-integrated version of the Sundial system will be delivered in July 1991.

## **2. Corpus-based evaluation**

The simulated H-C corpus also plays a central role in evaluation of the Sundial system. A first approach is to use the utterances of subjects in simulation experiments as a script for black box evaluations of the system as a whole. Evaluation at this level is unlikely to be very informative, since there may be many different 'correct' ways of responding to an utterance at a given point in a dialogue, only some of which will be represented in the simulation corpus. However, since the corpus is tagged with labels representing relatively 'deep' things such as dialogue acts and task goals (as well as the more 'surface' oriented lexical and syntactic tags), the dialogue management performance of the system can be evaluated separately from other functions of the system such as linguistic processing or message generation.

In general, the boundaries between software modules also correspond to boundaries between levels of corpus description. The Acoustic-Phonetic Decoding module passes the Linguistic Processing module a word lattice or graph which ideally includes the words which were actually uttered. The Linguistic Processing module selects the 'best' string (on the basis of recognition scores and grammatical well-formedness) and annotates it with lexical, syntactic, and semantic markers. This serves as input to the Dialogue Manager module which further annotates the string with dialogue act and task goal labels. At this point, the annotation process ceases for the input string. Instead, the Dialogue Manager introduces a new dialogue act/goal object corresponding to the system's next utterance. This is passed to the Message Generation module which maps the deep representation onto a surface representation realizing semantic, syntactic, and lexical features.

A glass box evaluation can be used to monitor the internal interfaces of the Sundial system. Each time a module outputs a representation, the new annotations added by that module can be checked automatically against those in the simulation corpus. Since no annotations of a given type are ever added to a string by more than one module, each module has absolute responsibility for those levels at which it adds annotations. The approach of monitoring inputs and outputs to system modules is particularly applicable to the Linguistic Processing and Message Generation modules. The Dialogue Manager is rather different since its output is not a modified version of its input. Rather, its output is a response to its input. To evaluate the Dialogue Module it is necessary to look inside it to see whether the dialogue act and task goal labels match those found in the corpus. It is also necessary to ensure that user utterance-system response dialogue act/goal label pairings for user input and system response belong to the set of legal pairings derived from the corpus.

### 3. Conclusion

Dialogue understanding systems are difficult to evaluate because it is hard to define what constitutes a well-formed dialogue. (*Spoken* dialogue systems add an extra layer of indeterminacy which further complicates them). In this paper I have suggested that dialogue understanding systems should be evaluated in respect of a corpus which includes a range of dialogue phenomena. The corpus should be annotated at a number of different levels and the capacity of the system to generate annotations which conform to a generalization of those found in the corpus should be measured. This process should be repeated with corpora other than the one used during the design phase.

How is the performance of the system at different levels to be harmonized for the purposes of overall system evaluation? I suggest that the highest levels - those involving dialogue acts and task goals - are all-important, and provide the principal index of the value and effectiveness of the system.

A word recognition system is fairly simple to evaluate directly in terms of its ability to recognize and interpret its input (i.e. to map it into a different form); so is a parser and a message generator. The main task of a dialogue manager, however, is not so much to *recognize* utterances as to *respond* intelligently to them. There should be a close *match* between the input and the output of a word recognition system, but here should be an unfolding *progression* from a dialogue manager's input to its output. By developing an abstract multi-level description of a corpus of simulation dialogues it is possible to define (i) a set of correspondences between different levels of analysis for a given string; and (ii) a set of well-formed *progression paths* through dialogues, expressed at the levels of dialogue acts and task goals, by which a dialogue manager can be evaluated.

Failure at the lower levels, such as word recognition, parsing, and text generation, is ultimately not serious so long as a well-formed progression path leads from dialogue opening and goal formation all the way through to goal satisfaction and dialogue closing. Along the way, there may be any number of *insertion sequences* for purposes of confirmation, clarification, and the repair of dialogue failures (spoken dialogues typically include many more of these than written dialogues), but the dialogue system can be said to have succeeded in its primary task if the end of a progression path is eventually reached. Up to a point, the number of insertion sequences is another metric which can be used in the evaluation of dialogue systems: the smaller the number of insertion sequences, the better the dialogue system. However, the ability to repair dialogue failures and the ability to recognize that confirmation or clarification may be necessary are vital skills in dialogue. Circumstances in which there is need for confirmation, clarification, or repair crop up routinely in ordinary dialogues.

In summary, the primary criterion for the evaluation of a dialogue system should be its ability to reach the end of progression paths consistently. A secondary criterion is that there should be no more insertion sequences than are genuinely required. Unpacking this criterion leads to more specific direct evaluation criteria for the lower level processes, as outlined above. An effective dialogue understanding system is not

one which never makes any mistakes, but rather it is one which always manages to recover from its mistakes.

### **References**

- Bilange, E. (1991) A task-independent oral dialogue model. *Proceedings of the 5th Annual Meeting of the European Chapter of the ACL*, Berlin. ???-???
- Fraser, N.M. and G.N. Gilbert (1991a) Simulating speech systems. *Computer Speech and Language* 5, 81-99.
- Fraser, N.M. and G.N. Gilbert (1991b) Effects of system voice quality on user utterances in speech dialogue systems. *Eurospeech'91*, Genoa, September.
- Fraser, N.M., G.N. Gilbert, S. McGlashan and R.C. Wooffitt (forthcoming) *Analyzing Information Exchange*. London: Routledge.
- McGlashan, S., E. Bilange, N. Fraser, N. Gilbert, P. Heisterkamp and N. Youd (1991) Managing oral dialogues. Sundial Project Working Paper.
- Peckham, J. (1990) Speech understanding and dialogue over the telephone: an overview of the Sundial project. *Acoustic Bulletin*.
- Ponamale, M., E. Bilange, K. Choukri and S.A. Soudoplatoff (1990) Computer-aided approach to the design of an oral dialogue system. *Proceedings of Eastern Multiconference 90: AI and Simulation*, 229-232, Nashville.



## Natural Language Processing Systems Evaluation Workshop

### Module-Level Testing: Lessons Learned

Sharon Flank, Aaron Temin, Hatte Blejer, Andrew Kehler, and Sherman Greenstein  
Systems Research and Applications Corp. (SRA)  
2000 15th St. North, Arlington, VA 22201  
703/558-4700  
flanks@sra.com

## 1. INTRODUCTION

This paper examines the evaluation of natural language understanding within a data extraction system. Our central claim is that NLP capabilities and "end-to-end" extraction success can and should be evaluated separately. (This observation parallels insights in Sundheim 1989 and Palmer and Finin 1990.) The contribution of this paper is to illustrate "lessons learned": we have implemented such an evaluation system in our work at SRA, and will present some of the attendant pitfalls and triumphs.

Successful evaluation requires, first, that two seemingly peripheral issues be addressed: test samples and the categorization of errors.

- *Representative test samples*

The test can serve as a predictor of how the system will fare only insofar as the test materials are representative of the actual corpus the system must handle.

- *Categorization of Errors*

Results depend not only on capabilities, but on the correctness of the supporting data. That is, does a particular capability succeed, assuming the data in the test sample are correct? Capabilities cannot be tested without using data. One way to acknowledge the contribution of data to the assessment of capabilities is to count data errors separately.

## 2. "BLACK BOX" VS. "GLASS BOX"

"Black box" evaluation focuses on input and output rather than on methods. It refers to evaluation of a full system, in this case from text to database fill. Black box evaluation has the advantage of providing quantitative results, but, because it offers no insight into what caused the failures and why, developers need further information if they are to use testing results to guide their work. "Glass box" evaluation looks inside the system to judge specific techniques and algorithms, but its very complexity makes it difficult to generate or interpret quantitative results.

In general, black box testing is appropriate only for a completed system. It gives only an end-to-end assessment, making it difficult to pinpoint areas of strength or weakness. It may even make it difficult to tell whether the system is excellent with only one significant flaw, or good but slightly flawed throughout. The culprit here is the cascade effect, as

Greenstein and Blejer 1990 note: The overall score can be no better than the worst score of any single component, or module. If every module were 80% accurate, the final results would be the same as if four modules were 100% accurate and one was 33% accurate. Though the difference in individual module performance may not be important to the end user, it is crucial to the developers.

It is not necessary to abandon black box testing entirely, and with it the quantitative measures that make it attractive: it is possible to do black box testing on *individual modules*. That is, quantitative results can be obtained using fixed input, and the core of the black box methodology, "Look at the input and output, not how it's processed," can be retained. Performing module-level black box testing helps to overcome the major disadvantage of end-to-end black box evaluation, since module-level output can provide useful feedback to developers.

Module-level black box evaluation is possible only when the system is modularly designed and the interfaces between modules are clearly defined. This idea is not new: black box evaluation of a single component is proposed in Palmer and Finin 1990 as a "glass box methodology" of choice (glass box in that it offers a look inside the system, but black box in that it does not look inside each module). They do not really discuss implementation, however. SRA's experience in evaluation highlights some of the successes and pitfalls in module-level black box testing. The modules, evaluation criteria and procedures are described below.

### 3. TESTING AT THE MODULE LEVEL

#### 3.1 Setting up the Procedures

SRA's natural language understanding system has separate modules for preprocessing (morphological analysis, lexicon lookup, multiword phrase handling); syntactic analysis (phrase structure grammar and bottom-up parser); semantic interpretation (compositional and frame-based); and discourse analysis (reference resolution and other intersentential links). We have designed and implemented a testing protocol that evaluates the output of each module separately, as well as standard end-to-end black box testing.

Performance of the testing procedures necessitated some changes to our original test design. Those changes are the topic of the remainder of this section. The testing cycle for both types of test, under the original design, was to follow these steps:

- 1) Define the *correct input*
- 2) Define the *correct output*
- 3) Process correct input through system under test to produce the *actual output*
- 4) Compare the actual output with the correct output to calculate a *score*
- 5) Analyze and evaluate the scores

- 6) Generate design and implementation tasks to correct errors
- 7) Execute those tasks, modifying the system under test
- 8) Return to step 3, above, and repeat

This cycle is represented schematically in Figure 1, below. The goal is to produce a series of scores over time that can be compared. If the only piece of the system that changes is the system under test, then the change in the scores should reflect the change in the capabilities of the system.

Certain statements follow from the above underlined phrase:

- 1) The correct input is definable, can be provided, and is fixed throughout development
- 2) The same is true for the correct output
- 3) The scoring method is complete and fixed; combined with 1 and 2 above, this means that a perfect score can be calculated at the beginning of testing

The reality of our development was that we violated all three of the above assumptions, in whole or in part. None of these negates the validity of our results, but they did serve to complicate our testing and evaluation.

The testing cycle with which we began to obtain our baseline results was:

- 1a) Define input using English words, phrases, and sentences to test capabilities that had been coded, or would be coded, into the system under test
- 1b) Use existing modules to process English input into properly structured correct input for the system under test
- 2) Don't explicitly create a file of correct output
- 3) Process correct input through system under test to produce actual output
- 4a) Analyze the actual output, marking correct output explicitly for future tests, and implicitly define remaining correct output by describing problems with incorrect output
- 4b) Score the actual output, based on analysis; enhance the scoring methodology as necessary to account for all the observed phenomena
- 5) Analyze and evaluate the scores

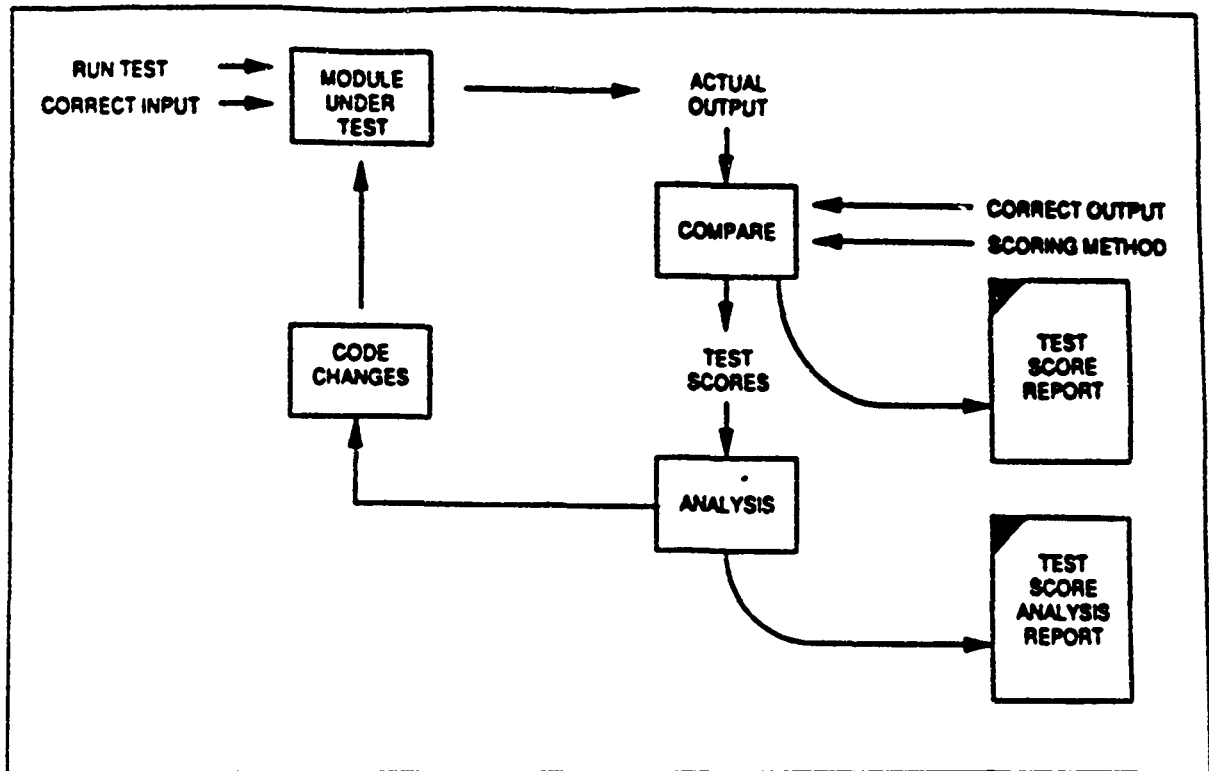


Figure 1: Test Methodology As Originally Presented

- 6) Generate design and implementation tasks to correct errors; coordinate with on-going tasks already specified as scheduled for execution during the project
- 7) Execute those tasks, modifying the system under test
- 8) Return to step 1a, above, and repeat

This is shown schematically in Figure 2.

The changes to the process take into account several aspects of actually developing the prototype system (especially the natural language understanding modules):

Defining the correct input and output for each module is extremely difficult. The modules are manipulating large and complex data structures. The only tools we have to *generate* the structures are the other modules of the system, which are themselves still under development. Therefore it is not always possible to get from the English input to the correct input for the module, making it difficult to test certain parts of a module until the other modules on which it depends are working properly. The ideal procedure also assumes that the interfaces to the systems under test are fixed. The project described

here is a prototype, so this assumption is false by definition.

It is particularly difficult to design the correct input and output for capabilities that haven't been designed yet. This is in fact an early part of the design process. It is therefore reasonable to provide test cases only for code that is implemented. This makes it difficult to compare scores from one test to the next.

- The scoring method should be expected to evolve, and needs to be customized for each module. Useful scoring data must contain more information than "wrong" or "missing" or "right for the wrong reason." As we learned what kinds of errors needed to be distinguished in each module in order to provide useful feedback for development, we refined the scoring methods. This leads to good results on the methodology, but inconsistencies in the scores themselves -- the results are often so voluminous that it is not worth the time it would take to go back and apply an insight gained in mid-scoring to previously scored output.
- The tasks generated to fix the errors must be merged into the workplan with the tasks already identified in the project workplan. In a purely testing-driven methodology, the only tasks to be done would be those identified by test errors. One would then expect to complete the implementation of those tasks before the next round of testing. Work on these tasks went on in parallel with testing. As tasks were defined by testing analysis, we determined if any of the workplan tasks superseded them. If not, we adjusted our workplan to accommodate these "new" tasks. This created certain scheduling problems, noted here only to suggest how testing affects project planning and progress.

Because the input and scoring methods vary over time, comparing scores over time is not as meaningful as we would like. An approach to this problem, if the interfaces to the modules remain fixed, is to run the "current" input on both the current and previous versions of the modules. This yields more comparable results, at the expense of an exponential number of scoring tasks.

### **3.2 Testing is Time-Consuming**

Selecting the input correctly depends on several criteria. Of course, it should be representative of the domain. The set should be small enough so that evaluation does not supplant system development. We intended to run the tests every two months, but the process was so time-consuming that we would recommend tests every six months instead.

We developed tests and testing procedures and ran three rounds of formal testing. Specifically, we devoted time to these tasks:

- Establish test suites for all four modules (Preprocessing, Syntactic Analysis, Semantic Interpretation, and Discourse Analysis)
- Conduct formal tests
- Analyze test results to determine the cause of the error
- Catalogue, assess and route the errors to the appropriate person to be fixed

Hours spent on testing as percentage of overall development and testing time: 13%

Most important, as noted in the previous section, every attempt should be made to minimize dependence on other modules. The effect of excessive interdependence is devastating in terms of time. If the syntax test includes items that do not correctly preprocess, then the input to the syntax test will have to contain hand-generated preprocessing output, a time-consuming procedure. Worse, if the semantics test uses items that do not parse correctly, many hours will be devoted to hand-generating output to mimic correct syntactic results. Discourse depends on correct results from all three previous modules, or hand-generated input. We originally intended to generate correct input by hand, but the process proved overwhelming. Instead we recommend that extreme care be taken in choosing the test suites so they test capabilities in a targeted way: embed discourse tests in sentences that parse, and use a restricted vocabulary unless you want to use your

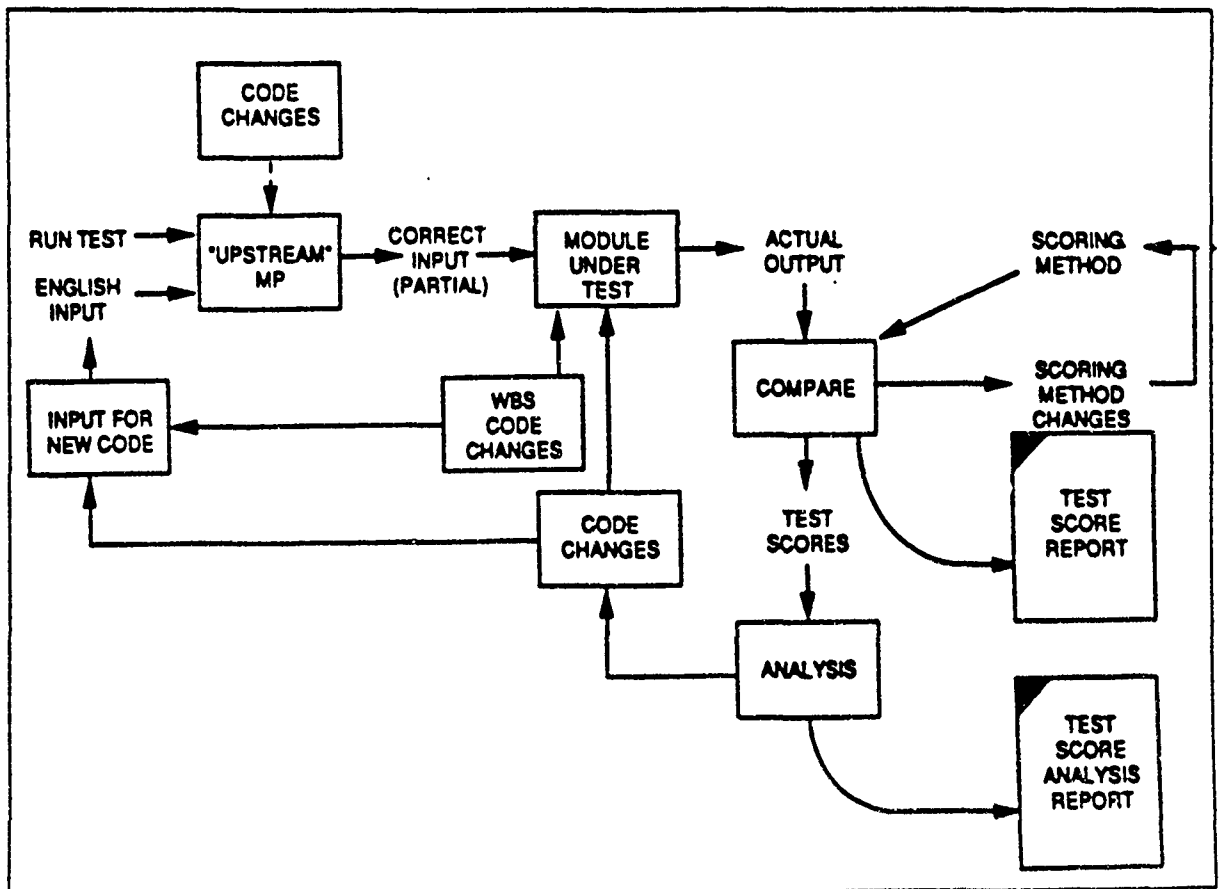


Figure 2: Test Methodology As Actually Executed

evaluation time to add data.

To give specific examples, if handling appositives correctly is an issue, then there is no point in testing the semantics of appositives if they don't yet parse correctly. Some examples, especially in semantics and discourse, could not be used because they required earlier results that could not be generated at that point in development.

On the output side, specifying expected results is slow and painful, particularly for capabilities still in the design phase -- it may be a research issue even to design the optimal output. We addressed this issue by assigning a zero score, but that created inconsistencies later, when the capability was added. Scoring is discussed in the next section.

## **4. SCORING**

To retain the advantages of black box evaluation, it is important to produce quantitative results. To provide useful feedback during development, qualitative results are necessary. This section describes how we used scoring to address both of those objectives.

### **4.1 Categorization of Errors**

Errors were each categorized as one of two types, Type I and Type II: Type I errors are those where some or all of the correct output is missing, in other words, the system did not get something it should have. Type II errors are those where an incorrect piece of output is generated, i.e., the system got something that it shouldn't have. These two types of errors have analogues in Information Retrieval: Type I errors are those in recall (the system did not retrieve a document that it should have) and Type 2 errors are those in precision (the system retrieved a document that it shouldn't have). This type of categorization into types occurred for both blackbox and glassbox evaluation.

Errors in the glassbox evaluation were also categorized in one of ten categories:

- 1) Algorithm Design Incomplete
- 2) Algorithm Design Incorrect
- 3) Algorithm Implementation Absent
- 4) Algorithm Implementation Incomplete
- 5) Algorithm Implementation Incorrect
- 6) Data Design Incomplete
- 7) Data Design Incorrect
- 8) Data Implementation Absent
- 9) Data Implementation Incomplete
- 10) Data Implementation Incorrect

These categories helped to determine what the nature was of errors in the system. Over the

three rounds of testing that occurred, this categorization had a predictable trend: the first round saw mostly Algorithm Design Incomplete Errors; the second round saw fewer of these errors but an increase in Data Implementation Errors; this was followed by a decrease in errors in general in the third testing round. This is indicative of the nature of system development: at first algorithms are either missing or incorrect; when they are developed, then inadequacies in the existing data are revealed; this is followed by resolution of those errors and thus a more accurate system.

## 4.2 Module-Level Scoring Methodology

Each module that was tested glass-box had to have the testing approach described above tailored to the needs and representations of that module. The following paragraphs describe how these were scored.

Preprocessing scoring Preprocessing was scored using an all-or-nothing approach. As the input suite is large (over 650 separate inputs), it seemed that 1 point per correct answer would yield a meaningful trend over time. For Preprocessing, Type I errors represented any incorrect or missing output. Type II errors catalogued output that was right for the wrong reasons (e.g., Chuck E. Cheese was recognized as a name because "cheese" wasn't in the lexicon, thus it was an unknown word, thus it defaulted to a name; as "cheese" is pretty common, we might expect it to be in a proper lexicon, and the name would not be recognized). Being the first module in the natural language understanding pipeline, Preprocessing has the easiest time getting correct input (though it was not 100% successful the first round, as it relies on a previous module to tokenize the English, and there were some problems with that module).

Syntactic Analysis scoring Syntax errors are counted by assigning one point for each constituent correctly tagged, designated as Type I, and one point for every constituent correctly attached, designated as Type II. That is, a Type I error is counted for each constituent that is mislabelled, and a Type II error is counted for each constituent that is attached to the wrong place in the parse tree.

Syntactic Analysis returns all possible parses that make sense according to our syntactic and semantic constraints, ordered from most to least plausible. In general, Semantic Interpretation works on the first (or top) parse in the list. Therefore it seems to make sense to evaluate how often the parse ranked best by the module is actually the correct parse. However, this makes little sense for scoring purposes, mostly because of the term "correct parse." If the top parse has zero errors, then it is, by our scoring, the correct parse. If the correct parse were in the list of returned parses, but not the top parse, then we would have to potentially score all returned parses to discover the one with zero errors. Given that Syntax may return a large number of parses, this is not practical with our current technology. Another assumption underlying such a measure is that all (or at least, several) of the parses returned are syntactically correct, but one is to be preferred. This implies that the grammar is perfect, but the weighting scheme is not. As our grammar is not yet perfect,



we would like to first concentrate on errors in the grammar. Afterwards, we can isolate the weighting scheme, and design it to return the correct parse as the first parse in the list returned. Therefore, though we considered reporting scores for both first parses and best parses, in the end we only reported scores for first parses.

Semantic Interpretation scoring A Type I error is scored for slots not produced that should have been produced, and correct values inserted in wrong slots. Type II errors are incorrect slots, and wrong values in correct slots. "Decisions" that Semantic Interpretation must make were scored, including:

- mapping to thematic roles
- sentential modifier mapping (adjunct handling)
- noun phrase modifier handling
- modifiers within sentential clauses or nominalizations
- correct predicate identification
- correct modification and scoping of stacked noun phrases
- correct updating of referenced knowledgebase objects

Each of these decisions is also worth a point.

Discourse Analysis scoring Phenomena are more easily isolated within Discourse Analysis than in other modules. Each test sentence or group of sentences (the Discourse Analysis subsuite has groups of interrelated sentences to test intersentential phenomena) generally tests one capability. This capability is either the successful resolution of a discourse object or an attribute being filled in an object through discourse phenomena. We score one point when the resolution or attribution succeeds, and zero if not. Type I errors are assigned when a resolution or attribution is either missed or gotten wrong. Type II errors are assigned when a reference or attribution was performed when it shouldn't have been (the subsuite has "negative" examples as well as "positive," i.e., to make sure that solutions that are too general are also penalized).

The complexity of the module-level error analysis requires that it be done by developers familiar with the system. Less detailed categorization serves black box needs (quantitative results) but does not show what needs to be fixed to remedy the errors. Even more important, a developer must decide which corrections will yield the most "bang for the buck," setting sensible priorities for error remediation.

When the capability to be tested is entirely absent, a score of zero is, of course, appropriate. But it is not always clear what the denominator should be; in some sense the solution must be designed in order to count how many points will be assigned to a correct solution. We dealt with this problem as best we could, by revising the denominators of previous test sessions once we had a clearer vision of the ideal solution.

## **5. CONCLUSION**

We have examined a module-level evaluation methodology for natural language understanding, illustrating insights gained during implementation. It is possible to produce useful quantitative and qualitative results using such a methodology, provided certain pitfalls are avoided.

## REFERENCES

- Greenstein, S. and H. Blejer (1990) A Test and Evaluation Plan for Natural Language Understanding. SRA Working Paper.
- Palmer, M. and T. Finin (1990) Workshop on the Evaluation of Natural Language Processing Systems. *Computational Linguistics* 16(3):175-181.
- Sundheim, B. M. (1989) Plans for a Task-Oriented Evaluation of Natural Language Understanding Systems. *Proceedings of the DARPA Speech and Natural Language Workshop*. 197-202.

## **Maintaining and Enhancing a NL DBMS Interface**

R. E. Cullingford, IBS, Inc., Milford, CT

M. Graves, Dept. of CS, University of Michigan, Ann Arbor, MI

### **Introduction**

The problem of evaluation of natural language (NL) technology pre-supposes a solution to a prior problem: that of creating the NL system in the first place. The most significant problem in this development process is circumscribing the system's linguistic, semantic and pragmatic capabilities. This problem is exacerbated when the system being evaluated is concurrently undergoing modification: the process of enhancement often causes breakages in pre-existing capabilities. This presentation will describe a practical attempt to grapple with the development problem in the context of a commercial English-language DBMS access tool called EasyTalk. A partial solution is embodied in the Lexicon Maintenance Facility (LMF), a software system designed to document the interrelationships among the words and phrases "understood" by the tool, and to control the tool's testing.

### **Corpus-Based Testing**

The standard method of documenting and testing the capability of a NL system is with a corpus of sentences which the system should process correctly. For commercial systems, these typically consist of several thousand queries (for example, BBN's FCCBMP corpus [RADC80]). Running the whole corpus can become unwieldy, and thus it may be useful to break the corpus into smaller, more manageable, corpora.

Several factors may affect the manner in which the corpus is divided, such as the class of NLI used. For example, testing of a syntax-first parser (e.g. [HEND78, KAPL83]) may lead to splitting the corpus to emphasize syntactic constituents, while a more semantically oriented parser (e.g., [BURT76, BIRN81]) might base its division on semantic categories.

The process of the division will be based on the desired granularity, organization, and coverage of the corpora. Should there be several small corpora, or relatively few large ones? Should they be arranged in a flat, unstructured set, or hierarchically arranged? Are the corpora to be in some sense "complete", say over a sublanguage (e.g., [KIT787]), or relatively sparse (i.e., an ad

hoc collection). Also, is the corpus to contain sentences which do not yet work, i. e., which represent new problems to be studied or desired enhancements?

The Lexicon Maintenance Facility described here houses a large (~2500) test query collection. The query set is organized around the so-called General English component of the EasyTalk lexicon, the collection of words and phrases supplied with the system and expected to have substantially the same meaning across database applications. (The Application-Specific part of the lexicon is supplied by the customer at setup time.)

We have divided our corpus in two ways. The first division is bottom-up via the lexicon. Each entry in the lexicon has a small corpus associated with it to test its functionality. This corpus documents the sentences that were used in the design of the lexical entry and also contains a sufficient number of sentences to adequately exercise it in testing.

Each of these corpora are grouped together into two different hierarchies. The first hierarchy is conceptual. The second is physical. This is useful for testing. If a file is changed, it is easy to test the corpora for all the lexicon entries in that file. More seriously, when a General English syntactic/semantic primitive changes, the hierarchy allows a quick computation of a query test set representing linguistic constructions which are probably affected by the change.

The second division of the corpus is top-down via linguistic/database Capabilities. In the DBMS access setting, a Capability is a characteristic database interaction (mediated by a query in a DBMS-interface language such as SQL) triggered by certain linguistic constructions. These Capabilities are arranged hierarchically. For example, EasyTalk supports Computations; Computations can have single or multiple operands; Addition is a Binary-Operand Computation. Each Capability has a corpus which demonstrates the functionality supported.

### **Conceptual Analysis**

EasyTalk's NL processor is based on the model of language understanding called Conceptual Analysis. Here, the attempt is to get directly to a representation of the meaning of a query without an intervening syntactic analysis. The analysis is managed by organizing syntactic and semantic expectations at the level of individual words and phrases. (See [Cull86] for a discussion of the approach and its methodologies.) For example, the query "Show average customer balance" is in the corpus for the lexeme "average." The effect of the processing

information stored with the lexeme is to mark the meaning representation as calling for a certain kind of single-operand computation. At the same time, query also belongs to the Capability corpus for Statistics Queries. As in all database retrieval queries, the ultimate meaning of the query depends upon the database structure of the application. Thus the capability tested will also depend upon the structure of the database (or the user's view of it).

### **Testing a Conceptual Analyzer**

EasyTalk is a tool for ad hoc information retrieval: a NL query is formulated, is processed into a DBMS query, and yields data formatted into a report. From the standpoint of testing, the LMF supports both "black box" and "glass box" evaluation. From the "black box" view, a query is correct if its report is correct. A "glass box" evaluation is supported by the storage of several key intermediate data structures, for example, the so-called Initial Meaning Representation which is the output of the conceptual analyzer; the so-called Final Meaning Representation, which may contain augmentations due to context analysis; the results of database structure ambiguity resolution rules, and so forth.

Within the LMF, a query can be scored as either correct or incorrect. If incorrect, the reason may be stated. Since a query may appear in several corpora, it is important to prevent a query from having more than one status. Inter-sentential ellipsis (follow-ons) pose a problem in this regard, since a given follow-on can legitimately be associated with more than one query, and thus, in principle, it can inherit the prior query's status

### **System Functionality**

The Lexicon Maintenance Facility is composed of three primary systems: Lexicon Control, Testing, and Maintenance; and two supporting systems: the Menu Selection System, and Lisp Reporter-Interface.

The Lexicon Control System performs four tasks:

- Inform the user of the contents of the lexicon upon demand.
- Define the corpora to test the General English lexicon. Each corpus is a collection of sentences that exercises the functionality of one lexical entry.
- Maintain these corpora, keeping track of changes to them.

- Assign unique, system-wide id's to query sentences. Jointly these queries form the Master Test Collection.

The Testing System performs two types of testing:

- Partial Testing of the lexcon. The natural language developer may need to test a portion of the lexcon that was changed or recently added. The developer specifies: a set of corpora, the name of a recently modified file, a set of natural language Capabilities (which exercises a NL or DB feature), or a collection of ad hoc queries.
- Complete Testing of the lexcon. All corpora are tested. This is a test of the Master Test Collection.

To support the Testing System, three utilities are provided:

- Use the stored records and indications of changes to generate the collection of queries to test.
- Run a log against this collection.
- Verify that the queries completed successfully. This is done by the Verification Utility. It does this by checking that the current results of the query match the results stored as being correct. The query results consist of the Meaning Representation and optionally the reports.

The Maintenance System tracks changes in the results from the Testing System. It allows the natural language developer to update the query results that the Verification Utility should consider to be correct.

The Menu Selection System allows the LMF user to make quick and efficient use of the facilities described above. The menu system consists of a (possibly tangled) hierarchy of choice menus. These choice menus allow the user to choose either another menu or a command to execute. If a command is chosen, the user will be prompted for its arguments. Additionally, the system will only allow responses of the correct (user-defined) type.

The Lisp-Reporter Interface generates the inputs necessary for the Batch Reporter to generate reports from a specification of either:

a set of corpora,

a set of natural language Capabilities, or  
a collection of ad hoc queries

and out of that collection of queries, whether or not reports should be generated for:  
queries that generate mql that have been specified to be correct,  
queries that generate mql that have been specified to be incorrect,  
queries that generate mql whose correctness has not been specified,  
all of those queries, or  
queries which have a different report or MR than is what stored.

### Conclusions

The LMF has been in use for almost two years. During that time EasyTalk has undergone explosive enhancement. We would probably not have been able to control its development without this facility, and thus the LMF embodies a first pass at the kind of technology that is needed to first create, then evaluate, large NL systems.

### References

- [RADC89] RADC-TR-89-302, Rome Laboratory, Griffiss AFB, NY.
- [HEND78] Hendrix, G.G., et. al., "Developing a natural language interface to complex data." ACM Trans. Database Systems, 3, 2, 1978.
- [KAPL83] Kaplan, S.J., "Co-operative response from a portable natural language system. In Brady and Berwick (eds.), Computational Models of Discourse, Cambridge: MIT Press, 1983.
- [BURT76] Burton, R.R., "Semantic Grammar," BBN TR-3453, 1976.
- [BIRN81] Birnbaum, L., and Selfridge, M., "Conceptual analysis of natural language." In Schank, R., and Riesbeck, C. (eds.) Inside Computer Understanding, Hillsdale, NJ: Erlbaum, 1981.
- [KIT78] Kittredge, R., "The significance of sublanguage for machine translation." In Nirenburg, S. (ed.), Machine Translation, Cambridge, UK: Cambridge University Press.
- [CULL86] Cullingford, R.E., Natural Language Processing: A Knowledge Engineering approach.



# Evaluation for Generation

**Marie Meter**  
BBN  
10 Moulton St.  
Cambridge, MA 02138  
MMETER@BBN.COM

**David McDonald**  
Content Technologies  
14 Brantwood Rd.  
Arlington, MA 02174  
MCDONALD@CS.BRANDEIS.EDU

## 1. Introduction

While evaluation is in full swing for speech and natural language understanding (NLU), with concrete metrics and competitions across sites, evaluation for natural language generation (NLG) has remained at the discussion stage. While some may argue this is because the field is smaller and less mature (and perhaps more stubborn), as we see it, the reason is that evaluating generation is more difficult. Not only is it hard to define what the input to a generator should be, but it is also hard to objectively judge the output.

In this paper we first set out some goals for NLG evaluation and point out potential pitfalls. We then look more closely at the particular problems for evaluating generation systems. Finally we explore some short term possibilities for evaluation and look to the long term.

## 2. Goals

It can be instructive to draw an analogy between evaluation and exams in school. As we tell students, exams are both to show them their strengths and to point out where they need improvement. We also give exams so that we can know who the better students are, who is paying attention and working hard. While exams succeed to some extent in meeting these goals, they also have their negative side: they encourage poor study habits, such as cramming, and they discourage creative problem solving, since the best grades most often go to those who answer in the way the teacher expects.

Evaluation for generation will also have positive and negative sides. The main goal should be to show us which techniques are succeeding and which problems still need work. In addition

(and whether we intend it or not) evaluations will make comparisons between systems and researchers by which funding agencies can rank projects; this real competition can lead to the adoption of the analogs of the poor study habits as researchers direct their work specifically to win the competitions. An inescapable effect of evaluation metrics is that they will lead the field to work on some problems over others, just as exams direct students to study some aspects of a subject over others. For example, a generation task requiring an exact paraphrase of an input text will discourage work on lexical choice, since a system with fewer possibilities is less likely to make an error.

To avoid the negatives, the metrics need to be designed carefully, so that they do not encourage last minute hacking and so that the judgements reward real progress rather than showmanship.

## 3. Issues in Evaluation of Generation Systems

One of the major problems with evaluating generation systems is the complex relationship between the input and the output. If we simply considered the output and attempted to do a post-hoc evaluation of generation systems, we would have to conclude that the field has regressed over the last 15 years. Consider the following samples of generated text:

*"The game started with my taking a corner, and you took an adjacent one. I threatened you by taking the middle of the edge opposite that and adjacent to the one which I had just taken, but you blocked it and threatened me. I blocked your diagonal and forked you. If you had blocked mine, you would have forked me, but you took the middle of the edge opposite of the corner*

which I took first and the one which you had just taken and so I won by completing my diagonal." **PROTEUS**, Davey 1974

"You know for some reason I just thought about the bill and payment again. (You shouldn't give me a bill.) <Uh> I was thinking that I (shouldn't be given a bill) of asking you whether it wouldn't be all right for you not to give me a bill. That is, I usually by (the end of the month know the amount of the bill), well, I immediately thought of the objections to this, but my idea was that I would simply count up the number of hours and give you a check at the end of the month." **ERMA**, Clippinger 1974

"Assume that there is some barber who shaves everyone who doesn't shave himself (and no one else). Call him Giuseppe. Now, anyone who doesn't shave himself would be shaved by Giuseppe. This would include Giuseppe himself. That is, he would shave himself, if and only if he did not shave himself, which is a contradiction. This means that the assumption leads to a contradiction. Therefore it is false, there is no such barber." **MUMBLE**, McDonald 1978.

"Kennedy only got a small number of delegates in the election on 20 February. Carter just lost by a small number of delegates. He has several delegates more than Kennedy in total." **PAULINE**, Hovy 1985.

"Little dogs chase mice." "Floyd bought Helga a book to read on the plane." **MUMBLE-86**, Meteor, McDonald, et al. 1987.

"Knox, which is C4, is en route to Sasebo. It is at 79W 18E heading SSW. It will arrive 4/24, and will load for four days." **PENMAN**, Hovy 1988.

"Mon. 08-MAY 89 10:49AM Abbie is at Lotus Point, which is at 125 Main Street. Her skill is managing. Abbie is a plant manager. Abbie likes watching movies. She watched "The Lady Vanishes" on SUN 07 May 7:20 pm." **SPOKESMAN**, Meteor, 1990.

There are many differences between the early systems, exemplified by the first three examples, and today's systems, exemplified by the last two, that cannot be seen by inspecting just the output text. First, the range of texts today's systems can produce is drastically greater. Clippinger's

**ERMA**, for example, only produced that one paragraph, whereas **SPOKESMAN** produces text for eight different applications and many different texts for each, covering most of what the applications are capable of representing.

### 3.1 Progress in NLG

The above examples make it clear that measuring progress in NLG is not a simple matter.

On one dimension, as the application becomes more complex, the more possible inputs the generator has to deal with and the more complex structurally those inputs will be. Generators that ignore some of that complexity may produce more fluent text by "canning" some sets of decisions; however, in the long term generators will have to handle the complexity in order to accurately reflect the situation/state the underlying program is in.

Related to this is how closely tied the generator is with its underlying program. In systems such as Davey's **Proteus** (Davey 1976) and Kukich's **Ana** (Kukich 1988) the application was custom designed to suit the needs of the generator. This close fit makes very fluent text possible, but at the cost of markedly more work in developing the applications, and the result cannot be used for any other applications. In general, as generators have become more portable and, thus able to be used with a variety of applications, the less fluent their text has become. In Figure 1, we label this dimension "complexity of the situation" for the number of different situations/states the generator can handle both within one application and across different applications.

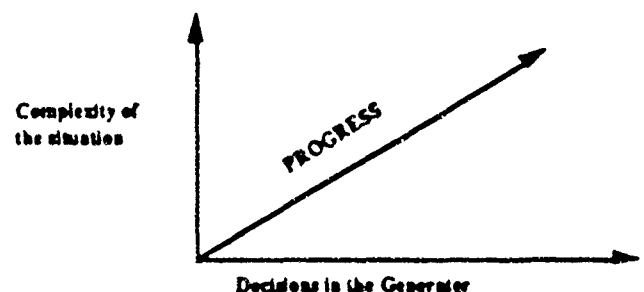


Figure One

On another dimension, as the number of decisions the generator ("consciously") makes increases, the less fluent the text becomes, at least

in the early stages. Template driven generators are doing less work, since many decisions are made simultaneously by executing the template. The extreme of this is a print statement, which is a single "decision" and can produce text as "fluent" as the programmer desires. Progress in this dimension is increasingly less stipulative components. Early generation systems, such as Proteus and Mumble didn't address areas such as content selection or text planning. As these and other problems are taken on more systematically in modern generators, the overall competence of the system initially falls off and then gradually increases as the new component becomes more sophisticated.

For example in Mumble-86 (Meeter, et.al 1987) and Nigel (Mann & Matthiessen 1985), where the focus was on linguistic competence, typical examples were single sentences (in contrast to the full paragraphs of earlier generators). As systems such as Spokesman and Penman grow today (using Mumble-86 and Nigel, respectively, as their linguistic components, the focus is on text planning, and they are producing paragraphs again. While they are not as fluently as the early systems, the paragraphs are more directly motivated by their underlying programs. However, until these text planners become more sophisticated, they will not fully exercise the competence of their linguistic components.

As Figure One shows, progress in the field can be seen as a progression from the simplest situation and a single decision point (such as a print statement in a compiler) toward composing text for complex underlying programs, programs that can not only represent a great deal of information, but also how that information is more and less salient in a given situation. However, as we have seen, the progress will not necessarily be reflected in the output: a print statement can reproduce Shakespeare.

### 3.2 "Glass Box" Evaluation

It is clear that in order to measure progress, we need to look at more than the output of generation systems. In a "glass box" evaluation, the goal is to look inside a system and evaluate the individual components. At first glance, it would seem easy to compare generators in this way. Nearly all divide the processing into two

components, linguistic realization and text planning, and most use the same kinds of knowledge resources: grammar, lexicon, plan library. However at closer inspection, the similarity of these terms is deceiving. For example, in some cases lexical choice is part of text planning, in others it is in linguistic realization. This difference also has effects on what information is in the lexicon: is it simply the words and their inflectional endings? Are different derivational forms in the same entry or different entries?

The conclusion of the AAI-90 Workshop on Evaluation of Generation Systems was the the field was not yet ready for glass box evaluation. We need to first be able to describe/define the generation process, and to do that we need to determine the space of decisions in the process overall. Currently, there is little agreement in the field on what the decision are, how alternatives should be represented, what control structure determines the order in which the decisions should be made, or the effect of a decision on subsequent decisions. Different researchers focus on different parts of the generation process (e.g. text planning vs. linguistic realization) and take into account different kinds of knowledge (e.g. discourse structure vs. user models, vs. taxonomic domain knowledge).

Addressing these issues within the generation community is the topic of the UCAI-91 Workshop on Decision Making in the Generation Process. We hope that this and subsequent workshops on the topic can provide a firm base for doing glass box evaluation in the future.

### 3.3 "Black Box" evaluation

Given the complexities of evaluating progress in the field and individual components of generation systems, an alternative is treating systems like a "black box" and only looking at input/output behavior. This is at the approach taken in speech, and to some extent in speech/language systems. In this section, we discuss the viability of such an approach for generation.

#### 3.3.1 The input to generation

The most obvious problem in a black box approach is determining what an appropriate input

for generation is. Researchers in generation define the boundaries to the process differently and can have quite different requirements on what information must be represented even at the points where they overlap. There has been considerable effort on determining a workable common input for purposes of evaluation: The topic was one of the sessions in the AAAI-90 Workshop on Evaluating Generation Systems and it was a panel session at the 1991 European Workshop on Generation last spring. But these forays are only the beginning of a long process, and concrete results should not be counted on for evaluations in the near term. Determining the necessary properties of the input that are required to produce fluent text is a key part of the generation problem, and hypothesizing that the input should take a certain form is part of how progress is made. In this light, it is only natural that different generation projects should presume different character inputs, and it would be wrong to penalize a project by making the choice arbitrarily.

One possible way around the problems created by stipulating the input for the evaluation is to provide each project with the source code to a complete application program and have them extract whatever input they happen to need from that program and give it the representations that they use. Unfortunately this approach has its own deficits since it is not just the representation of the input that projects differ on but the amount of the information it supplies and the even ontological assumptions behind how that information is conceptualized. If the application program is taken off the shelf or written by some "neutral" third party then much of what will be tested may not be generation per se but the projects' ability to bridge the ontology and representational formalisms of the off-the-shelf system to their own requirements, augmenting the information that the program supplies with more information that they need. This is a practical problem in the real world, and one that we could consider evaluating, but it would not be an evaluation of the generation systems per se.

### 3.3.2 The output of generation

A second problem in an I/O evaluation is judging the output of a generator: it is very difficult to be objective in evaluating texts. As teachers of college composition can attest, once the students no longer make grammatical and

punctuation errors, the line between a "B" and a "C" becomes very subjective. You cannot easily point to what is wrong with a composition and there is no simple set of instructions for how to correct it, as there is for a calculus exam.

This suggests that the right approach for an objective evaluation would be put the generator in the context of some larger system, where the generator is answering a question (see Section 3.3 on task based evaluation). However, now only one aspect of the text is being judged, its content. In order to judge other aspects (fluency, effectiveness, style), we are going to have to accept non-objective evaluations.

This is not to say that subjective evaluations cannot be fair. Many kinds of subjective rulings are handled by having a panel of judges with a set of criteria that each contribute a score to the final ruling. This is accepted practice in such wide ranging domains as essay contents, figure skating, and beauty contests. Since most people are "experts" at judging language, using panels of judges to rate the output of generators is not out of the question. Note however, that this kind of evaluation is costly: there is no automatic scoring program that can read these texts and make judgements.

### 3.4 Task based evaluation

Given the difficulties in determining the input for a generator and evaluating the output, we might consider embedding the generator into a task that is evaluable, and then evaluating the performance in that task. This is the approach taken for language understanding, both for speech language systems, where the task is database access, and message processing, where the task is template fill. This approach avoids the problems of evaluating language per se; however, it introduces the problem of designing the task so that it will be the better generators that will actually perform better at the task, and not, for example, the ones with the best interfaces.

Two questions arise in defining such a task for generators. First is how to objectively judge performance without having to judge the language per se, which, as we pointed out earlier, is subjective. There are many ways to effectively communicate the same message, so enumerating the possible answers is not in general possible.

An alternative is to define a task in which the generator produces instructions or directions which a person then has to follow. The system could then be judged on how well the person follows the instructions/ directions. Of course, we then run into the problem of credit assignment: what is being evaluated, the instructions or the person following them?

#### 4. Conclusion

So far, we have raised more questions than we have provided answers. A simple black box evaluation for generation systems requires defining an input and comparing the output, both of which we have shown are very difficult for generation. A task based analysis, which avoids these problems, must address the issue of credit assignment. Does this mean there is no hope for objective evaluation of generations systems in the near term? Perhaps. As in the examination process, some subjects don't adapt well to exams with "right" and "wrong" answers. Does it mean that we cannot compare our systems? No. The latest methodology in teaching composition is to get the students to work together to informally critique each others work. In generation, this translates to putting more effort into hands-on "working workshops" such as the AAI-90 and the upcoming IJCAI-91 generation workshops, and less effort into designing formal metrics. What does it mean for the long term? As the students go on to higher level courses, the need for such objective measures decreases. Advanced courses have essay exams or require only papers. As our generators and the applications they work with mature, we must put effort into designing evaluation techniques that let us quantitatively compare the effectiveness of our systems.

#### References

- Clippinger, John (1977) *Meaning and Discourse: a computer model of psychoanalytic speech and cognition*, Johns Hopkins.
- Davey, Anthony (1974) *Discourse Production*. Edinburgh University press. Edinburgh, Scotland.
- Hovy, Eduard (1985) "Integrating Text Planning and Production in Generation", *Proceedings IJCAI-85*, Los Angeles, August 1985, 848-851.
- Hovy, Eduard (1988) "Planning Coherent Multisentential Paragraphs" In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, Buffalo, New York, June 7-10, 1988, p. 163-169.
- Kukich, Karen (1988) "Fluency in Natural Language Reports" in McDonald & Bolc (eds) *Natural Language Generation Systems*. Springer-Verlag, New York. p. 280-311
- Mann, William & Matthiessen, Christian (1985) "Nigel: A Systemic Grammar for Text Generation" in Freedle (ed.) *Systemic Perspectives on Discourse: Selected Theoretical Papers of the 9th Intl. Systemic Workshop*, Ablex.
- McDonald, David D. (1975) "A Preliminary Report on a Program for Generating Natural Language", *Proceedings IJCAI-75*, Wm. Kaufman, 401-405.
- Meteer, Marie W. (1990) *The Generation Gap: The problem of expressibility in Text Planning*. Ph.D. Thesis. University of Massachusetts.
- Meteer, Marie W., David D. McDonald, Scott Anderson, David Forster, Linda Gay, Alison Huettner, Penelope Sibun (1987) *Mumble-86: Design and Implementation*, UMass Technical Report 87-87, 173 pgs.

# Evaluating Natural Language Generation Facilities in Intelligent Systems

Johanna D. Moore  
University of Pittsburgh  
Department of Computer Science  
and  
Learning Research and Development Center  
Pittsburgh, PA 15260  
*jmoore@cs.pitt.edu*

## 1 Introduction

Natural language generation capabilities are an important component of many classes of intelligent systems. Expert systems rely on generation facilities to produce explanations of their knowledge and behavior (e.g., [McKeown, 1988, Moore and Swartout, 1989, Paris, 1990]), intelligent tutoring systems employ generation components to instruct students, provide hints, and correct misconceptions (e.g., [Cawsey, 1989]), and help systems employ generation techniques to advise users about how to achieve their goals (e.g., [Wilensky *et al.*, 1984, Wolz, 1990]). In these contexts, systems must produce complex multi-sentential texts, e.g., justifications of results, definitions of terms, descriptions of domain objects, instructions about how to perform domain tasks, and comparisons of alternative methods for solving problems or achieving goals.

In the field of expert systems, it was recognized that the explanation capabilities of a system are tightly coupled to the knowledge base and reasoning component of that system. Evaluations of early expert system explanation facilities showed that many types of questions users would like to ask could not be answered satisfactorily because the knowledge needed to justify the system's actions, explain general problem solving strategies, or define the terminology used by the system was simply *not* represented and therefore could not be included in explanations [Clancey, 1983, Swartout, 1983]. However, simply improving the knowledge bases did not alleviate all of the explanation limitations; knowledge about language was also required, see [Moore, 1989]. This led to the realization that the range of user questions an explanation facility is able to handle and the sophistication and quality of the responses it can produce depend on (at least) two knowledge sources: (1) knowledge about the domain and how to solve problems in that domain as represented in the intelligent system's knowledge base, and (2) knowledge

about how to construct an adequate response to a user's query in some communication medium or combination of media.

As intelligent systems become more sophisticated and include capabilities for tailoring presentations to the knowledge and beliefs of the individual user, the current problem-solving situation, and the previous discourse, the quality of the natural language utterances produced relies on an increasing number of additional knowledge sources. Participating in such interactions requires methods for interpreting users' utterances and actions, methods for recognizing users' goals and plans, strategies for choosing relevant information to include in response to different types of questions in different situations, and knowledge about how to organize and express the desired content in a coherent (multimedia) presentation tailored to a particular user and dialogue situation. Natural language generators for such systems must deal with a wide range of issues including: discourse management, content planning and organization, planning referring expressions, and choosing grammatical structures and lexical items.

In general, the quality of the responses produced is dependent not only on the "linguistic capabilities" of the natural language generator, but on the intelligent system's domain model, user modeling component, dialogue manager, plan recognizer, etc. This leads to a fundamental problem for those interested in the problem of evaluating natural language generation facilities. Two possible approaches for evaluating the generation facilities of such systems seem appropriate: (1) Devise a set of evaluation criteria that can be applied to a generation component in isolation and systematically and objectively measured. This requires an identification of the aspects of the problem of presenting a text to the user that are to be considered the responsibility of the generation facility and the types of inputs and contextual factors it must be able to take into account. (2) Design experiments which allow evaluation of the natural language generation facility in the context of the larger intelligent system of which it is a part.

In this abstract, I will first discuss some of the problems I see with the first approach. The second approach involves evaluating user satisfaction and/or performance as a result of the natural language generation facility. Almost any set of evaluation criteria proposed will include criteria such as "understandability". Because such criteria can only be judged by human users, I believe that all evaluations must ultimately involve experiments with human subjects. The abstract concludes with a study I have designed to evaluate the natural language component I am building for an intelligent tutoring system.

## **2 Methods for Evaluation**

### **2.1 Devise Evaluation Criteria for NL Generation Facilities**

I see three main problems with attempting to devise a set of evaluation criteria for natural language generations facilities.

1. **Where to draw the line.** One of the first issues that must be settled is to determine exactly what aspects of the problem of presenting an utterance are considered under the purview of the generation facility. One distinction often quoted in the text

generation literature divides the generation problem into a *strategic component* which is responsible for selecting the content to include in the text from a knowledge base and ordering that information, and a *tactical component* which is responsible for producing grammatically correct sentences expressing the chosen content. Decisions such as where to put sentence boundaries and choosing lexical items are typically made by the strategic component or by an interface between the two components. While many would agree that the capabilities of the tactical component are rightly the responsibility of a generation facility, there is much less agreement regarding the strategic decisions. Should tasks such as choosing a discourse strategy (e.g., analogy vs. definition), content planning, planning referring expressions, and lexical selection be considered the jurisdiction of the generation facility? If so, it becomes more difficult to separate out the generation facility from other parts of the system because these issues are affected by various other components, e.g., the knowledge base, user model, and plan recognizer. If these issues are not considered part of the generation facility, then many of the most important factors affecting the quality of the utterances produced will not be evaluated.

This problem may be equivalent to the problem of identifying what the inputs to a generation facility should be. Should the input consist of intentional goals to be achieved, a set of topics to be expressed, or both? How should knowledge about the user and current context be provided to the generator? Until we make some headway on these issues, it will be difficult to devise a satisfactory set of evaluation criteria.

2. Evaluation criteria must be task dependent. One of the problems with attempting to decide what to include as part of the generation facility is that this decision is task dependent. For example, it is common to expect that a generation facility used for expert system explanation should select the content to be included in an explanation. However, the generation component of a machine translation system need not select content since this comes from the parse of the source text.

Recently, Swartout has made a similar observation. He has argued that the development of evaluation criteria for a generation system depends heavily on how that system will be used and that the development of task-independent criteria for evaluating generation systems is very difficult, if not impossible, as not all criteria are relevant to all tasks. For example, Swartout argues that one important criteria that a generation facility for expert system explanation must meet is that of *fidelity*, i.e., the explanations produced by the generator must be an accurate representation of what the expert system actually does. Clearly such a criterion would not be appropriate for evaluating the generation component of a machine translation system.

Swartout has called for development of several sets of criteria customized to the major uses of generation systems and has put forth a set of desiderata for explanation facilities [Swartout, 1990]. These criteria place constraints on the the explanations themselves, the mechanism by which explanations are produced, the adequacy of the expert system's knowledge base, and the effects of an explanation facility on the construction and execution of the expert system of which it is a component. For a more extensive discussion of the criteria and their implications, see [Swartout, 1990].

3. Evaluating subjective factors. The criteria proposed by Swartout are very



comprehensive and are quite useful as qualitative guidelines. However, it would be desirable to form evaluation metrics from these criteria with objective methods for assigning ratings to an explanation system. In some cases, the task of devising a method for assigning a value to the metric seems straightforward. For example, fidelity can be assessed by comparing traces of the system's problem-solving behavior with the natural language explanations it produces. Another of the criteria calls for low construction overhead, and techniques from software engineering could be helpful in estimating the overhead in system construction due to the explanation facility and how much savings in the maintenance and evolution cycles are due to design decisions attributable to the requirements imposed by explanation. Even some aspects of the understandability criterion could be objectively measured. For example, one way to evaluate the factor of composability (smoothness between topic transitions in a single explanation) would be to analyze the system's explanations to determine whether they adhere to constraints governing how focus of attention shifts, as defined by Sidner (1979) and extended by McKeown (1982).

However, in other cases, it is difficult to envisage how objective measures for assessment could be devised. For example, how can we assign a value to an explanation's naturalness (linguistic competence) and coherence? Furthermore, what is understandable to one user may be obscure to others. The ratings of such factors are inevitably subjective and can only be judged by human users. The most promising way to assess the understandability of a system's explanations will involve techniques that attempt to measure users' satisfaction with the explanations or the impact of the explanations on users' performance.

## 2.2 Evaluating Generation Facilities in Context

The purpose of a generation component in an intelligent system is to facilitate the user's access to the information and knowledge stored in that system. Thus one way to assess the generation component, is to assess the impact of natural language utterances on users' behavior and/or satisfaction with the system. This can be done using direct methods such as interviewing users to determine what aspects of the system they find useful and where they find inadequacies, or by indirect methods which measure users' performance after using the system or monitor usage of various facilities.

**Assessing user satisfaction.** One of the best sources of assessment information is the user population. Soliciting user input regarding the appropriateness and helpfulness of the natural language utterances produced by a system provides valuable feedback. In the case of the MYCIN explanation facility, user reports of inadequate responses led to identification of limitations in expert systems and inspired research efforts to address the limitations.

**Monitoring usage of natural language generation facility.** Another telling assessment of an any automated tool is whether or not users actually avail themselves of the tool and whether or not that usage is successful, i.e., they are able to get the information they seek or are able to make the system perform the task they desire. Such an evaluation has been done in the area of help systems. An empirical study of usage of the Symbolics DOCUMENT EXAMINER, an on-line documentation system that supports

keyword searches, indicated that a substantial number of interactions with the system ended in failure, especially when users were inexperienced [Young, 1987].

**Assessing impact on users' performance on task.** Another way to assess the contribution of a generation component is to determine how the natural language capabilities of the system contribute to users' learning or effectiveness in using the system to achieve their goals. For example, if an explanation component is included as part of an intelligent tutoring system, then it should be possible to design a simple experiment that assesses the explanation component's contribution to learning. Moreover, by varying the explanation strategies employed by the system, we can design studies that compare alternative explanation strategies.

### 3 A Proposed Evaluation Study

I am currently building an explanation component for SHERLOCK II, an intelligent coached practice environment developed to train avionics technicians to isolate faults in a complex electronic device. Using SHERLOCK II, trainees acquire and practice skills by solving a series of problems of increasing complexity using a simulation of the actual job environment in which these skills will be required. In SHERLOCK II, natural language texts are generated in response to students' requests for help during problem solving and also in the *Reflective Follow-Up Phase (RFU)* which allows students to review their own problem-solving behavior and compare it to that of an expert. During problem-solving, hints review the student's steps, explain the normal function of a component, tell the student what component to test next, or present a method for testing a component. During RFU, students can ask the system to justify its conclusions about the status of components, suggest what should have been tested in what order, or compare alternative strategies for diagnosing faults.

Currently, SHERLOCK's natural language interaction with the user is accomplished with simple, non-adaptive strategies using canned text. The system does not have a conceptual model of what it has said to the user and previous hints affect later hints in only the simplest possible way, i.e., if the system has given the first hint attached to a particular problem-solving goal and the trainee asks for more, the system gives the next hint in the list for that goal. While SHERLOCK has been successful in field testing [Nichols *et al.*, in press], SHERLOCK project members feel that further improvements will come from enhancements to the explanation facility and a more sophisticated student modeling component.

The explanation generator I am building plans presentations, using text and graphics, from a set of strategies that are being derived from analyses of human-human tutoring interactions in this domain. By adapting my previous work [Moore, 1989], the presentation planner will be able to plan explanations tailored to the individual user, problem-solving situation, and dialogue context.<sup>1</sup> The explanation generator will be capable of answering follow-up questions in context and elaborating on previous explanations.

---

<sup>1</sup>The planner makes use of a student modeling component described in [Kats, 1991].

I will test two hypotheses: (1) a practice environment that is capable of providing hints and explanations is better than one that simply simulates the environment allowing the student to explore with no explanatory feedback, and (2) an adaptive explanation facility (capable of tailoring explanations to users and situations, providing elaborations and answers to follow-up questions) is better than a non-adaptive facility (using canned hints and responses to questions) in terms of students' satisfaction with the system as well as their learning of the troubleshooting task, retention of skills, and transfer of knowledge to related tasks.

To test our hypotheses, we will run a study in which three groups of subjects are compared: Group 1 will use a system which provides no hints or explanations, Group 2 will use the existing, non-adaptive explanation facility, and Group 3 will use the adaptive explanation facility currently being built.

Students' satisfaction will be assessed using direct observation and interview techniques. Subjects from each group will use the system with an observer present. They will be instructed to make any comments they have about the system to the observer. The observer will also interview each subject to solicit subject opinion about the appropriateness and helpfulness of explanations. Comparing the data gathered for each group will allow us to determine which type of system is favored.

To assess learning, subjects in each group will work through a sequence of problems. After solving each problem, they will engage in an RFU session. Each student's performance on solving each troubleshooting problem will be measured. SHERLOCK II contains tools for automatically monitoring and assessing both higher-level (e.g., ability to choose next component to test) and lower-level skills (e.g., ability to use the oscilloscope). We will make use of these facilities to measure students' overall performance as well as performance gains during the problem sequence. If our hypotheses are correct, we will expect subjects in Group 3 to show the greatest performance gains and overall score.

## References

- [Cawsey, 1989] Alison Cawsey. *Generating Explanatory Discourse: A Plan-based Interactive Approach*. PhD thesis, University of Edinburgh, 1989.
- [Clancey, 1983] William J. Clancey. The epistemology of a rule-based expert system: a framework for explanation. *Artificial Intelligence*, 20(3):215-251, 1983.
- [Katz, 1991] Sandra Katz. Modelling the student in an intelligent coached practice environment for electronics troubleshooting, 1991. submitted to the Third International Workshop on User Modelling.
- [McKeown, 1982] Kathleen R. McKeown. *Generating Natural Language Text in Response to Questions About Database Structure*. PhD thesis, University of Pennsylvania, 1982. Published by University of Pennsylvania as Technical Report MS-CIS-82-5.
- [McKeown, 1988] Kathleen R. McKeown. Generating goal-oriented explanations. *International Journal of Expert Systems*, 1(4):377-395, 1988.

- [Moore and Swartout, 1989] Johanna D. Moore and William R. Swartout. A reactive approach to explanation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August 20-25 1989.
- [Moore, 1989] Johanna D. Moore. *A Reactive Approach to Explanation in Expert and Advice-Giving Systems*. PhD thesis, University of California, Los Angeles, 1989.
- [Nichols *et al.*, in press] P. Nichols, R. Pokorny, G. Jones, S. P. Gott, and W. E. Alley. Evaluation of an avionics troubleshooting system. Technical Report Special Report, Air Force Human Resources Laboratory, Brooks AFB, TX, in press.
- [Paris, 1990] Cécile L. Paris. Generation and explanation: Building an explanation facility for the explainable expert systems framework. In *Natural Language Generation in Artificial Intelligence and Computational Linguistics*. Kluwer Academic Publishers, Norwell, MA, 1990.
- [Sidner, 1979] Candace L. Sidner. *Toward a Computational Theory of Definite Anaphora Comprehension in English Discourse*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1979.
- [Swartout, 1983] William R. Swartout. XPLAIN: A system for creating and explaining expert consulting systems. *Artificial Intelligence*, 21(3):285-325, September 1983. Also available as ISI/RS-83-4.
- [Swartout, 1990] William R. Swartout. Evaluation criteria for expert system explanation, July 29 1990. Presented at the AAAI90 Workshop on Evaluation of Natural Language Generation Systems, Boston, Massachusetts.
- [Wilensky *et al.*, 1984] Robert Wilensky, Yigal Arens, and David Chin. Talking to unix in english: An overview of uc. *Communications of the Association for Computing Machinery*, 27(6):575-593, 1984.
- [Wolz, 1990] Ursula Wolz. The impact of user modeling on text generation in task-oriented settings. In *Proceedings of the Second International Workshop on User Modeling*. AAAI and the University of Hawaii, 1990.
- [Young, 1987] Young. Using the full power of computers to learn the full power of computers. Technical Report 87-9, Institute of Cognitive Science, University of Colorado, Boulder, CO, 1987.

# MEASURING COMPOSITIONALITY IN TRANSFER-BASED MACHINE TRANSLATION SYSTEMS\*

Björn Gambäck

Swedish Institute of Computer Science

Box 1263, S - 164 28 KISTA, Stockholm

gam@sics.se

Hiyan Alshawi, David Carter and Manny Rayner<sup>†</sup>

SRI International

Cambridge Computer Science Research Centre

23 Millers Yard, Cambridge CB2 1RQ, U.K.

hiyan@cam.sri.com, dmc@cam.sri.com, manny@cam.sri.com

## Abstract

The paper describes a simple method for objectively evaluating the compositionality of a transfer-based Machine Translation system. The question is the extent to which rule interaction gives rise to (unwanted) side-effects. An example is given of the use of the method in the context of the BCI (Bilingual Conversation Interpreter), an interactive transfer-based bidirectional Machine Translation system.

## Introduction

When trying to evaluate a Machine Translation system, two different approaches are possible: either the system's behaviour in its proposed environment is assessed, or the theoretical coverage and worth of the transfer formalism is evaluated. The first type of evaluation concentrates on translation quality and effectiveness, while the latter seeks to specify which linguistic constructions the system can handle. Most work in the field have been concerned with system behaviour; here, we will concentrate on linguistic coverage.

In the literature on Machine Translation, a number of criteria are mentioned as significant

\*Part of the research described in this paper was also reported on at the Meeting of the International Working Group on Evaluation of Machine Translation Systems, Les Rasse, Switzerland, April 1991.

<sup>†</sup>The work reported here was funded by the Swedish Institute of Computer Science, and the greater part of it was carried out while the fourth author was employed there.

when evaluating the worth of a transfer formalism; among these are *expressiveness, simplicity, generality, reversibility, language-independence, monotonicity* and *compositionality*. Unfortunately, when trying to convince others of the worth of one's own approach, it soon becomes evident that most of these are not easy to measure objectively, if they are not absolute properties of the formalism. (In particular, a pure unification-based formalism is guaranteed to be monotonic). To say, for example, that a formalism is "good" from the point of view of expressiveness, and then back this up with five carefully-chosen examples, is not really to say very much.

Compositionality, however, can be measured objectively. Here, we will describe a simple method for evaluating the compositionality of a transfer-based MT system, and give an example of its use in the context of the BCI (Bilingual Conversation Interpreter) (Alshawi *et al* 1991), an interactive transfer-based bidirectional system currently being developed in a co-operation between SICS and SRI Cambridge. The main components of the BCI are English (Alshawi ed. 1991) and Swedish (Gambäck, Lövgren & Rayner 1991) versions of the SRI Core Language Engine, transfer taking place at the level of Quasi Logical Form (QLF) (Alshawi & van Eijck 1989); the transfer formalism is unification-based and bidirectional. Our approach to Machine Translation is aimed at keeping the transfer component as simple as possible, while depending on fully constrained reversible monolingual grammars for correct analysis and synthesis.

## Measuring compositionality

Perhaps the most important factor in keeping transfer simple is the degree to which the transfer relation is a homomorphism, i.e. the degree to which transfer rules are compositional.

For compositionality to be a meaningful notion in the first place, it must be possible for transfer rules to apply to partial structures. These structures can consequently occur in different contexts; other transfer rules will apply to the contexts as such. The question is the extent to which particular combinations of rules and contexts give rise to special problems. In a perfectly compositional system, this will never happen, although it seems a safe bet that no such system exists today. What we want is a method which objectively measures how closely we approach the compositional ideal.

Our first step in this direction has been the construction of *compositionality tables*, in which a set of rules and a set of contexts are systematically combined in all possible meaningful combinations. This is done in order to figure out the extent to which the complex transfer rules continue to function in the different contexts.

In the following three diagrams, we give an example of such a table for the current version of the BCI. Table 1 gives a set of rules, which exemplify six common types of complex transfer. Table 2 gives a set of twelve common types of context in which the constructions referred to by the rules can occur. Finally, Table 3 on the next page summarizes the results of testing the various possible combinations.

To test transfer compositionality properly, it is not sufficient simply to note which rule/context combinations are handled correctly; after all, it is always possible to create a completely *ad hoc* solution by simply adding one transfer rule for each combination. The problem must rather be posed in the following terms: if there is a single rule for each complex transfer type, and a number of rules for each context, how many extra rules must be added to cover special combinations? It is this issue we will address.

*Table 1: Types of complex transfer used*

Type	Example
Different particles	John likes Mary John tycker om Mary
Passive to active	Insurance is included Försäkring ingår
Verb to adjective	John owes Mary \$20 John är skyldig Mary \$20
Support verb to normal verb	John had an accident John råkade ut för en olycka
Single verb to phrase	John wants a car John vill ha en bil (lit.: "wants to have")
Idiomatic use of PP	John is in a hurry John har bråttom (lit.: "has hurry")

*Table 2: Transfer contexts used*

Context	Example
Perfect tense	John has liked Mary John har tyckt om Mary
Negated	John doesn't like Mary John tycker inte om Mary
YN-question	Does John like Mary? Tycker John om Mary?
WH-question	Who does John like? Vem tycker John om?
Passive	Mary was liked by John Mary blev omtyckt av John
Relative clause	The woman that John likes Kvinnan som John tycker om
Sentential complement	I think John likes Mary Jag tror John tycker om Mary
Embedded question	I know who John likes Jag vet vem John tycker om
VP modifier	John likes Mary today John tycker om Mary idag
Object raising	I want John to like Mary Jag vill att John ska tycka om Mary ( <i>"I want that J. shall like M."</i> )
Change of aspect	John stopped liking Mary John slutade tycka om Mary ( <i>"J. stopped like-INF M."</i> )

*Table 9: Compositionality Table  
(Swedish-English shown above English-Swedish)*

Transfer context	Different particles	Active to passive	Verb to adjective	Support verb to normal verb	Single verb to phrase	Idiomatic use of PP
Present tense	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
Perfect tense	OK OK	generator OK	OK OK	OK OK	OK OK	OK OK
Negated	pres-not pres-not	pres-not pres-not	pres-not pres-not	past-not past-not	pres-not pres-not	transfer transfer
YN-question	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
WH-question	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
Passive	OK OK	- -	- OK	- -	- OK	- -
Relative clause	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
Sentential complement	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
Embedded question	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
VP modifier	OK OK	transfer transfer	OK OK	OK OK	OK OK	OK OK
Change of aspect	OK OK	OK OK	OK OK	OK OK	OK OK	OK OK
Object raising	transfer OK	transfer OK	transfer OK	transfer OK	transfer OK	transfer OK

Each square in Table 3 consists of two entries, the first for the Swedish-English, and the second for the English-Swedish direction. The entries are to be interpreted as follows:

- - means that the combination was not applicable, i.e. that the construction referred to by the rule cannot occur in this context.
- OK means that analysis, transfer and generation all functioned correctly, without any extra rule being necessary to deal with the particular context.
- generator means that the generator component was unable to generate the correct target language sentence.
- transfer means that the transfer component was unable to make a correct transfer.
- All other entries are names of rules needed to deal with special combinations of rule and context. For this table, only two extra rules were needed: *pres-not*, which reverses the relative scope of the operators for negation and the present tense and *past-not*, which performs a similar function for the past tense.

The actual results of the tests were as follows. There were 136 meaningful combinations (some constructions could not be passivized); in 115 of these, transfer was perfectly compositional, and no extra rule was needed.

Of the remaining 21 rule/context/direction triples, seven failed for basically uninteresting reasons: the combination "Perfect tense + Passive-to-active" did not generate in English, and the six sentences with the object-raising rule all failed in the Swedish-English direction, since that rule is currently uni-directional. The final fourteen failures are significant from our point of view, and it is interesting to note that all of them resulted from mismatches in the scope of tense and negation operators.

The question now becomes that of ascertaining the generality of the extra rules that need to be added to solve these fourteen unwanted interactions. To reorder the scopes of tense, negation and modifiers, and account for the scope differences between the English and Swedish QLFs arising from the general divergences in word-order and negation of main verbs relevant here, two rules involving general transformations of the QLF structure were added. These solved ten of the outstanding cases.

The four bad interactions left all involved the English verb *to be*; these were the combinations "Passive to active + VP modifier" and "Idiomatic use of PP + negation", which failed to transfer in either direction. Here, there is no general solution involving the addition of a small number of extra rules, since the problem is caused by an occurrence of *to be* on the English side that is not matched by an occurrence of the corresponding Swedish word on the other. The solution must rather be to add an extra rule for each complex transfer rule in the relevant class to cover the bad interaction.

Summarizing the picture, to solve the specific examples in the test set, two extra rules were thus required. The tests revealed that all bad interactions between the transfer rules and contexts shown here could be removed by adding four extra rules to cover the 124 possible interactions.

## Extending the framework

It should be pointed out that the compositionality table presented here is still too small to detect more than a fraction of the bad rule interactions

that may occur in the current system. Most important is to extend systematically the set of contexts, taking note of the fact that many of the features they are intended to represent are in fact orthogonal to each other.

A full set of contexts would include at a minimum all legal combinations of independent choices along the following dimensions:

- *Tense*: Present, past or future.
- *Mood*: Active or passive.
- *Negation*: Positive or negative.
- *Modification*: Unmodified, PP modification, ADVP modification, modified by fronted constituent.
- *Clause-type*: Declarative sentence, Y-N question, WH-question, relative clause, sentential complement, embedded question, progressive VP complement, object raising.

Multiplying out all the choices gives a total of 384 distinct contexts; this must then be multiplied by the number of transfer rule types to be tested, and doubled to get both directions of transfer. With the figures given above, 4608 sentences would have to be tested. In practice, of course, not all combinations are possible. Specifically, passives don't interact well with other rule-contexts, leading to a total size of the test set of 3082 sentences.

Developing the software support needed to be able to run tests of this size regularly is clearly not a trivial task, but our opinion is that being able to do so greatly contributes to maintaining the system's reliability and integrity. We are thus giving high priority to constructing the necessary tools in the current phase of the project.

Also worth noting is that the tests described above are exclusively at the sentence level. For complete tests of the compositionality of transfer, one would have to construct test schemes for at least the noun phrase level, as well. The compositionality tables for NPs should account for the interactions (in various positions) of different NP-modifiers. Thus, the transfer contexts should be something like the ones suggested in Table 4 and the transfer types should include the ones given in Table 5. This will be further studied in the next phase of the project.



## Conclusions

We have described a straight forward way of measuring the compositionality of transfer-based MT systems by the use of "compositionality tables". We claim this to be a good method for the objective evaluation of one aspect of MT systems, even though the tables given in this paper should be further extended to capture more transfer contexts and types of transfer rules, as well as NP-structures.

Transfer context	Example
Plural	car parks parkeringsplatser
Definite	the car park parkeringsplatsen
Genitive	car park's parkeringsplatsens
Pre-modified by Adjective	big car park stor parkeringsplats
Pre-modified by Genitive	his car park hans parkeringsplats
Post-modified by PP	car park here parkeringsplats här
Post-modified by Relative clause	car park which I use parkeringsplats som jag använder

Transfer type	Example
Adjective Noun to Noun	bad luck otur
Noun PP to Noun	chairman of the board styrelseordförande
Noun Noun to Noun	car park parkeringsplats
Past Participle to Adjective	The broken cup Den trasiga koppen
Adjective to Relative clause	The uninsurable car Bilen som inte kan försäkras
PP to Genitive	The end of the story Sagens slut

## References

- Alshawi, H. and J. van Eijck. 1989. "Logical Forms in the Core Language Engine". *27th Annual Meeting of the Association for Computational Linguistics*, Vancouver, British Columbia, 25-32.
- Alshawi, H., ed. 1991 (to appear). *The Core Language Engine*. Cambridge, Massachusetts: The MIT Press.
- Alshawi, H., D. Carter, B. Gambäck and M. Rayner. 1991. "Translation by Quasi-Logical Form Transfer". *29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, California.
- Alshawi, H., D. Carter, B. Gambäck and M. Rayner. 1991 (to appear). "Swedish-English QLF Translation". In H. Alshawi (ed.), op. cit.
- Falkedal, K. Forthcoming. "Evaluation Methods for Machine Translation Systems: A Historical Overview and a Critical Account". *Technical Report*, ISSCO, Geneva.
- Gambäck, B., A. Lövgren and M. Rayner. 1991 (to appear). "The Swedish Core Language Engine". *SICS Research Report*, Swedish Institute of Computer Science, Stockholm.
- Lehrberger, J. and L. Bourbeau. 1988. *Machine Translation: Linguistic characteristics of MT systems and general methodology of evaluation*. Amsterdam: John Benjamins Publishing.
- Way, A. 1991. "Developer Oriented Evaluation of MT Systems". *Technical Report*, The University of Essex.

# Good Applications for Crummy Machine Translation

Kenneth W Church<sup>1</sup>  
Eduard H Hovy

USC Information Sciences Institute  
4676 Admiralty Way  
Marina del Rey, CA 90292

## 1. Introduction

We have recently begun work in machine translation and felt that it would probably make sense to start by surveying the literature on evaluation. As we read more and more on evaluation, we found that the success of an evaluation often depends very strongly on the selection of an appropriate application. If the application is well-chosen, then it often becomes fairly clear how the system should be evaluated. Moreover, the evaluation is likely to make the system look good. Conversely, if the application is not clearly identified (or worse, poorly-chosen), then it is often very difficult to find a satisfying evaluation paradigm. We begin our discussion with a brief review of some evaluation metrics that have been tried in the past, and then move on to a discussion of how to pick a good application.

Why work on machine translation now, and what kind of MT is most likely to be commercially and theoretically profitable? Though the ALPAC report concluded in the sixties that there should be more basic research in MT, it stated clearly that this basic research could not be justified in terms of short-term return on investment.<sup>2</sup> In particular, when compared with human capabilities (still the ultimate test), MT systems of the time were not deemed a success, and might never be.

This belief may help explain the resistance of many MT researchers to take evaluation questions seriously. The EUROTRA project, for example, consciously decided to delay evaluation discussions as long as possible: "Exact procedures for evaluation will be decided by the programme's management committee

toward the end of each phase..." (Johnson *et al.*, 1985, p. 168) Others argue against any human-related evaluations as follows:

Performance of operational MT systems is usually measured in terms of their cost per 1,000 words and their speed in pages per post-editor per hour vs. the relative cost and speed of human translation.... In my opinion, it is becoming increasingly uninformative to compare the performance of MT systems with that of human translators, even though many organizations tend to do that to justify their MT investments. (Tucker, 1987, p. 28)

We believe that these attitudes hurt the cause of MT in the long run. As is proved by the increasing availability of commercial MT and MAT systems (such as Systran, Fujitsu's Atlas, Logos, IBM's Shalt, and several others, for less than \$100,000), MT today is beginning to find areas of real (commercial) applicability. Thus, to the questions "Has anything changed since ALPAC? How can one build MT systems that make a difference?", we answer that the community needs to find evaluation measures and applications that highlight the value of MT research in those areas where systems can be employed in a real (and economically measurable) way. Human and machine translation show complementary strengths. In order to design and build a theoretically and practically productive MAT system, one must choose an application that exploits the strengths of the machine and does not compete with the strengths of the human. This point is well put in the following:

"The question now is not whether MT (or AI, for that matter) is feasible, but in what domains it is most likely to be effective.... The object of an evaluation is, of course, to determine whether a system permits an adequate response to given needs and constraints." (Lehrberger and Bourbeau, 1988, p. 192)

<sup>1</sup> The first author's permanent address is AT&T Bell Laboratories, Murray Hill, NJ.

<sup>2</sup> "The Committee recommends expenditures in two distinct areas. The first is computational linguistics as a part of linguistics—studies of parsing, sentence generation, structure, semantics, statistics, and quantitative linguistic matters, including experiments in translation, with machine aids or without. Linguistics should be supported as science, and should not be judged by any immediate or foreseeable contribution to practical translation.... The second area is improvement of [human] translation [with respect to practical issues such as speed, cost, and quality]" (Pace *et al.*, 1966, p. 34)

What then are appropriate evaluation measures? It would be nice if the evaluations were to identify those (aspects of) MT systems that make them suitable for, and then steer them towards, high-payoff niches of functionality. But in spite of all the literature on MT evaluation, the general evaluation measures that are proposed often fail to pinpoint the strengths of systems and lead them toward real utility; instead, they seem to confound important and less important aspects. Tucker's review of *Taum-Meteo* and *Metal*, for example, might give one the mistaken impression that both systems work about equally well (namely, approx. 80%).<sup>3</sup>

"*Taum-Meteo* has been operational since 1977, translating about five million words annually at a rate of success of 80% without post-editing." (Tucker, 1987, p. 31)

"[T]he *Metal* system is reported to have achieved between 15% and 85% 'correct' translations, using an experimental base of 1,000 pages of text over the last five years." (Tucker, 1987, p. 32)

However, these numbers do not accurately reflect the crucial difference between these two systems. *Taum-Meteo* is generally regarded as a fairly complete solution to the domain-restricted task of translating weather forecasts whereas *Metal* is widely regarded as a less complete solution to the more ambitious task of translating unrestricted text. The evaluation measure ought to be able to highlight the strengths and weaknesses of a system. Apparently, the "success rate" measure fails to meet this requirement, presumably because it is too vague to be of much use.<sup>4</sup>

Unfortunately, this failure seems to be characteristic of many of the task-independent evaluation metrics that have been proposed thus far. Since, in our opinion, the blame is to be laid on the desire for generality, we propose that MT evaluation metrics should be sensitive to the intended use of the system. In this paper, we begin by outlining metrics that have been proposed and end by concluding that it becomes crucial to the success

<sup>3</sup> According to Isabelle (personal communication), *Meteo* currently achieves 97% success on a volume of 30 million words per year. The increased performance is largely due to improvements in the communication system; communication noise used to be responsible for a large percentage of the failures.

<sup>4</sup> The success rate of 80% reported in (Isabelle, 1984, p. 265) probably should not be compared with the numbers reported for *Metal*. In addition to translating the input, *Meteo* also attempts to determine if the translation should be checked by a professional translator. The 80% figure reported in (Isabelle, 1984) refers to the fraction of the input that *Meteo* handles by itself without assistance from a professional translator. The figures reported for *Metal* refer to an evaluation of the correctness of the output.

of an MT effort to identify a high-payoff niche application so that the MT system will stand up well to the evaluation, even though the system might produce crummy translations.

## 2. Traditional Evaluation Metrics

### 2.1. System-based Metrics

We identify three major types of evaluation metrics: *system-based*, *text-based* and *cost-based*. System-based metrics count internal data resources such as the number of words in the lexicons, rules in the grammars, semantic, grammatical, or lexical features, the number of representation elements in the semantic ontology or *Interlingua* (if any), and the number of translation rules (if any). The literature contains many examples of system-based metrics, for instance:

At the moment there are about sixty subgrammars for analysis and about 900 rewriting rules in total... number of rewriting rules for transfer and generation processes is around 800, and it will be increased in the coming few months. The dictionary contains about 16,000 items at present, and will be increased to 100,000 items at the end of the project. (Nagao, 1987, p. 276)

An advantage of these metrics is that they are easy to measure, which makes them popular. But since these metrics are tied to a particular system, they cannot be used very effectively for comparing two systems. They are much more effective for calibrating system growth over time. The major disadvantage of these metrics is that they are not necessarily related to utility.

### 2.2. Text-based Metrics

#### 2.2.1. Sentence-Based Metrics

These metrics, the most common class, are applied to individual sentences of target texts by counting, for example, the number of sentences semantically and stylistically correct, the number of sentences semantically correct, but with odd style, the number of sentences partially semantically correct, the number of sentences semantically and syntactically incorrect, and the number of sentences missed altogether. A good example appears in (Nagao et al., 1986), in which sentences are classed into one of five categories of decreasing intelligibility and into one of six categories of decreasing accuracy. Another example is the evaluations developed to measure the results of *Eurotra* systems (see Johnson et al., 1985).

Given the subjective nature of semantic, syntactic, and (especially) stylistic "correctness", these metrics are impossible to make precise in practice. In addition, their limitation to single sentences makes them too simplistic (for example, it is not clear how to scale the metric when several source sentences are combined in the target text, or when parts of them are grouped into sentences differently)

### 2.2.2. Comprehensibility Metrics

These metrics seek to measure translation quality by testing the user's comprehension of the target text as a whole. They include counting the number of texts translated well enough for full comprehension, the number of texts in which enough could be gleaned to get a reasonably good understanding of the content, though details may be missing, the number of texts in which some content could be gathered, enough to tell whether the text is of interest to the user or not, the number of texts with fatal inconsistencies or omissions, and the number of texts missed altogether

These evaluation metrics enjoy some significant advantages. First, they can be performed by the intended user of the translation, requiring little or no source language expertise. Second, they take in stride the mis- or even non-translation of text due to certain relatively isolated phenomena which have proven very hard to handle in computational systems in a general way (but which people can figure out themselves fairly easily). A major disadvantage of these metrics is the difficulty of quantifying them. One approach to overcome this difficulty is to create comprehension questionnaires that measure (in SAT-test-like manner) how understandable translations are to their intended users with respect to their intended uses. An example, using a test suite of texts, is proposed in (King and Falkedal, 1990). A second approach is to determine how willing users would be to pay for professional translation of the text, given the translated version. Since professional translation is expensive, the users will be motivated to identify the more useful systems.

### 2.2.3. Amount of Post-Editing

Metric in this subclass are based on the amount of work required to turn the translated text into a form indistinguishable from a human translator's effort. Ways of quantizing this include counting the number of editing keystrokes required per page, timing the revision process per page, and counting the percentage of machine-translated words in final text. An example is the keystroke count reported as follows:

"As an alternate measure of the system's performance, one of us corrected each of the

sentences in the last three categories (different, wrong, and ungrammatical) to either the exact or the alternate category. Counting one stroke for each letter that must be deleted and one stroke for each letter that must be inserted, 776 strokes were needed to repair all of the decoded sentences. This compares with the 1,918 strokes required to generate all of the Harvard translations from scratch." (Brown *et al.*, 1990, p. 84)

Some researchers object to keystroke counting because they don't believe that the counts are correlated with utility.

### 2.3. Cost-based Measures

The third major type of metric concentrates on the system's efficiency in producing a translation, as in:

1. cost per page of acceptable translation (machine, human, or mixed),
2. time per page of acceptable translation (machine, human, or mixed).

One such evaluation was done on Taum-Aviation (Isabelle and Bourbeau, 1985)

<i>Task</i>	<i>Machine</i>	<i>Human</i>
Preparation /input	\$0.014	\$0.000
Translation	\$0.079	\$0.100
Human revision	\$0.068	\$0.030
Transcription /proofreading	\$0.022	\$0.015
Total (Can. \$ per page)	\$0.183	\$0.145

The problem with cost-based metrics is that they often don't make the systems look very good. As can be noted from the table above, the evaluation shows that Taum-Aviation is actually more expensive than human translation (HT). If one wants the system to look good, it is important to pick a good niche application.

### 3. Characteristics of a Good Niche

We believe a good niche application should meet as many of the following desiderata as possible:

- (a) it should set reasonable expectations,
- (b) it should make sense economically,
- (c) it should be attractive to the intended users,
- (d) it should exploit the strengths of the machine and not compete with the strengths of the human,
- (e) it should be clear to the users what the system can and cannot do, and
- (f) it should encourage the field to move forward toward a sensible long-term goal.<sup>5</sup>

<sup>5</sup> Many long-term goals have been proposed over the years. FAHQ (fully-automatic high-quality translation) (Bar-Hillel,

#### 4. Extensive Post-Editing (EPE): An Inappropriate Niche

It is not easy to identify a good niche application. One cannot simply take a state-of-the-art MT program and give it to a bunch of salesmen and expect a miracle. One has to find an application that makes sense.

The extensive post-editing (EPE) application would appear to be a natural way to get value out of a state-of-the-art MT system. But unfortunately, the application fails to meet most of the desiderata proposed above.

##### 4.1. (a) Realistic Expectations

One can find numerous testimonials in the literature that sound too good to be true (and probably are):

"Although you can expect to at least double your translator's output, the real cost-saving in MT lies in complete electronic transfer of information and the integration into a fully electronic publishing system." (Magnusson-Murray, 1985, p. 180)

"Substantial rises in translations output, by as much as 75 per cent in one case, are being reported by users of the Logos machine translation (MT) system after only a few months." (Lawson, 1984, p. 6)

"For one type of text (data description manuals), we observed an increase in throughput of 30 per cent." (Tschira, 1985)

Statements such as these run the risk of setting unrealistic expectations, and consequently, in the long run, it is possible that they could actually do more harm than good. (We discuss the dangers of unrealistic expectations in section 7.) If users could really expect even modest gains in productivity, then one would expect that EPE products offered by ALPS, Logos, Systran, Weidner and others would stand on their merits in the marketplace, and would not need all the hype.

##### 4.2. (b) Cost Effectiveness

In fact, careful trials appear to indicate that EPE is actually more expensive than human translation (HT). Van Slype (1979) estimated that EPE costs 475 Bfrs. per 100 words, almost twice as much as HT (150-250 Bfrs. per 100 words). The Canadian government found more or less the same result in their trial of the Weidner product:

"[T]he HT production chain was significantly faster than the MT production

1960, p. 94) is perhaps one of the more well-known proposals

chain. How much faster depends on which phases of the MT chain are counted. If we count all the steps on the log form, human translation was nearly twice as fast as machine translation. If we discount the time that the machine actually takes to translate (on the assumption that the participants could use this time to do other useful tasks), as well as the time for the second dictionary update (on the grounds that these new or modified entries are not intended for the current text), MT remains 27% slower than HT. If, in addition, we discount the time for text entry, assuming that source texts arrive in machine readable form that Weidner could import, MT still remains 5% slower than HT for all the texts translated during the operational phase of the trial." (Macklovitch, 1991, p. 3)

Thus, there are serious indications that it may not be commercially viable to use professional translators as post-editors. In fact, there have been questions about the cost effectiveness of the EPE application dating back to the ALPAC report, well before many of these products were introduced into the marketplace.<sup>6</sup>

"The postedited translation took slightly longer to do and was more expensive than conventional human translation... Dr. J. C. R. Licklider of IBM and Dr. Paul Garvin of Bunker-Ramo said they would not advise their companies to establish such a service." (Pierce *et al.*, 1966, p. 19)

##### 4.3. (c) Attractiveness to Intended Users

In addition, EPE has failed to gain much acceptance among the intended target audience of professional translators, because post-editing turns out to be an extremely boring and tedious chore.<sup>7</sup>

"Most of the translators found postediting tedious and even frustrating. In particular, they complained of the contorted syntax produced by the machine. Other complaints concerned the excessive number of lexical alternatives provided and the amount of time required to make purely mechanical

<sup>6</sup> The cost effectiveness of the EPE application is discussed in more detail in Appendix 14 of the ALPAC report. The appendix observed that postediting tends to "impede the rapid translators and assist the slow translators" (Pierce *et al.*, 1966, p. 94). This would suggest that EPE products might be more appropriate for casual use by an amateur rather than daily use by a professional.

<sup>7</sup> Perhaps the task would be less tedious if the user interface were made more flexible and more user-friendly.

revisions" (Pierce *et al.* 1966, p. 96)

"Many, but not all, translators decided, after the first phase of the MT experiment, that Systran was not a translation aid, because they found that it took too long, and was too tedious, to convert raw MT into a translation to which they would be prepared to put their name." (Wagner, 1985, p. 203)

"When asked by the consultant if they would like to continue working with Weidner on the same texts after the end of the trial, not a single participant accepted." (Macklovitch, 1991, p. 4)

After reading Macklovitch's description of some of the errors in (Macklovitch, 1986), one can easily appreciate why some of the translators would be frustrated with the post-editing task. Macklovitch observed that approximately half of the errors in one sample involved the overuse of French articles. In translating an English noun phrase into French, it is a pretty good bet that the French noun phrase should begin with an article even if there isn't one in English. However, this rule does not hold in tables, where the French use of articles is apparently somewhat more like English. As it happened, one of the texts used in the trial contained a very long list of crop varieties published by Agriculture Canada, most of which should not have been translated with an article. Unfortunately, the Weidner system did not know that noun phrases work differently in tables, and consequently, the post-editor was faced with the rather tedious task of deleting the article and adjusting the capitalization for each of the crop varieties in this very long list. The professional translator probably would have found it quicker and more rewarding to translate the list from scratch.

#### 4.4. Kay's Characterisation of EPE

One can continue to go through the list of desiderata proposed above and find even more reasons why EPE is an inappropriate niche. Rather than beat a dead horse ourselves, we thought we would let Martin Kay do it for us, as only he can:

"There was a long period -- for all I know, it is not yet over -- in which the following comedy was acted out nightly in the bowels of an American government office with the aim of rendering foreign texts into English. Passages of innocent prose on which it was desired to effect this delicate and complex operation were subjected to a process of vivisection at the hands of an uncomprehending electronic monster that transformed them into stammering streams

of verbal wreckage. These were then placed into only slightly more gentle hands for repair. But the damage had been done. Simple tools that would have done so much to make the repair work easier and more effective were not to be had presumably because of the voracious appetite of the monster, which left no resources for anything else. In fact, such remedies as could be brought to the tortured remains of these texts were administered with colored pencils on paper and the final copy was produced by the action of human fingers on the keys of a typewriter. In short, one step was singled out of a fairly long and complex process at which to perpetrate automation. The step chosen was by far the least well understood and quite obviously the least apt for this kind of treatment." (Kay, 1980, "The Proper Place of Men and Machines in Language Translation," p. 2)

#### 5. A Constructive Suggestion: The Workstation Approach

Having established that EPE is inappropriate, Kay then suggested a workstation approach. At first, the workstation might do little more than provide word-processing functionality, dictionary access and so on, but as time goes on, one might imagine functionality that begins to look more and more like machine translation.

"I come now to my proposal. I want to advocate an incremental approach to the problem of how machines should be used in language translation. The word *approach* can be taken in its original meaning as well as the one that has become so popular in modern technical jargon. I want to advocate a view of the problem in which machines are gradually, almost imperceptibly, allowed to take over certain functions in the overall translation process. First they will take over functions not essentially related to translation. Then, little by little, they will approach translation itself. The keynote will be modesty. At each stage, we will do only what we know we can do reliably. Little steps for little feet!" (Kay, 1980, p. 11)

In his concluding remarks, Kay expressed the hope that his approach be implemented by someone with enough "taste" to be realistic and pragmatic.

"The translator's amanuensis [workstation] will not run before it can walk. It will be

called on only for that for which its masters have learned to trust it. It will not require constant infusions of new *ad hoc* devices that only expensive vendors can supply. It is a framework that will gracefully accommodate the future contributions that linguistics and computer science are able to make. One day it will be built because its very modesty assures its success. It is to be hoped that it will be built with taste by people who understand languages and computers well enough to know how little it is that they know" (Kay, 1980, p 20)

In fact, Kay's approach has recently been implemented by people who understand the practical realities well enough to take an even more modest approach than Kay himself probably would have taken. CWARC (Canadian Workplace Automation Research Center) has undertaken to provide the Canadian government's Translation Bureau with a translator's workstation that could be deployed in the near-term to the bureau's 900 full-time translators (Macklovitch, 1989). For obvious pragmatic considerations, they have decided to use the following off-the-shelf components:

- (a) a PC/AT,
- (b) network access to the Termium terminology database on CD-ROM,
- (c) WordPerfect, a text editor,
- (d) CompareRite, a program for comparing two versions of a text file,
- (e) TextSearch, a program for making concordances and counting word frequencies,
- (f) Mercury/ Termex, a program for maintaining a private terminology database,
- (g) Procomm, a program providing remote access to data banks via a telephone modem,
- (h) Seconde Memoire, a program that deals with French verb conjugations, and
- (i) Software Bridge, a program for converting word processing files from one commercial format into another.

This is clearly an ideal starting point for introducing technology into the translator's workplace. They will hopefully be able to demonstrate that the PC-based workstation is clearly superior to dictation machines. After they have achieved a trackrecord of success and the new technology has been in place for a while, they will be in a much better position to introduce additional tools, which might be more exciting to us, but also more risky for the managers at the translation bureau.

One might imagine all kinds of exciting tools. For example, the workstation could have a "complete" key, like control-space in Emacs, which would fill in the rest of a partially typed word/phrase from context. One might take this idea a step further and imagine that it ought to be able to build a super-fast typewriter that would be able to correct typos and fill in context given relatively few keystrokes. Peter Brown (personal communication) once remarked that such a super-fast typewriter ought to be possible in the monolingual case observing that there is so much redundancy in language that the user should only have to type a few characters per word, or about the equivalent of 1.25 bits per character (Shannon, 1951),<sup>8</sup> which is only slightly more than a byte (ascii character) per English word on average. The user should have to type even less in the bilingual case because the source language should provide quite a number of additional bits of information.

The super-fast typewriter may still be a ways off, but we are almost already in a position to provide some very useful but less ambitious facilities. In particular, the Translation Bureau currently spends a lot of resources retranslating minor revisions of previously translated materials (e.g., annual reports that generally don't change much year after year). It would be very useful if there were some standard tools for archiving and retrieving previously translated texts so that the translators would have access to the previous translations, when appropriate. It is also becoming possible to use bilingual concordances to help with terminological issues.

The workstation application stands up to the six desiderata proposed above much better than the EPE application. It is (a) much more realistic, so it should have a better chance of (b) economic success. After all, it ought to be able to beat dictation machines, at least in many cases. In addition, it has a better chance of (c) being attractive to the intended users and (d) exploiting the strengths of the machine as well as those of the human since it is being developed and tested by professional translators at the request of a translation organization. Since it is so modest it should be (e) fairly clear what it can and cannot do. Finally, there is a (f) clear path plan toward a desirable long-term goal, since the strategy explicitly calls for more and more ambitious tools as time goes on.

<sup>8</sup> Shannon's estimate that English has an entropy of 1.25 bits per character is probably too optimistic. In practice, one would probably expect a practical system to have an entropy somewhat closer to 1.76 bits per character (Brown et al., 1991).

## 6. Another Constructive Suggestion: Appeal to the End-User

The workstation approach is a direct appeal to the professional translators; it uses the benefits of office-automation as a way to sneak technology into the translator's workplace. An alternative approach, which also seems promising to us, is to use the speed advantages of raw (or almost raw) MT to appeal to the end-user who may not require high-quality.

### 6.1. Rapid Post-Editing

After noting the translators were unlikely to support the EPE application because they are unlikely to choose MT over HIT, Wagner found that end-users would often opt for crummy quick-and-dirty translation, if they were given a choice.

"We therefore decided to use Systran in a different way -- to provide a faster translation service for those translation users who wanted it and were willing to accept lower-quality translation." (Wagner, 1985, p. 203)

The output from Systran was passed through a 'rapid post-editing' service that emphasized speed (4-5 pages per hour) over quality. When the project was first presented to the translation staff, it was well-received and 13 out of 35 volunteered to offer the rapid post-editing service on the understanding that they could opt out if they did not enjoy it. Wagner found that "the option is popular with a number of users and perhaps surprisingly, welcomed with some enthusiasm by CEC [Commission of the European Communities] translators who find rapid post-editing an interesting challenge" (Hutchins, 1986, p. 261).

Wagner's rapid post-editing service is a much better application of crummy MT than EPE because it gives all parties a choice. Both the users and the translators are more likely to accept the new technology, warts and all, if they are given the choice to go back and do things the old-fashioned way. The trick to being able to capitalize on the speed of raw MT is to persuade both the translators and the end-users to accept lower quality. Apparently, the end-users are more easily convinced than the translators, and therefore, for this approach to fly, it is important that the end-users be in the position to choose between speed and quality.

### 6.2. No Post-Editing

The Georgetown system was used extensively at the EURATOM Research Center in Ispra, Italy, and the Atomic Energy Commission's Oak Ridge National Laboratory from 1963 until 1973. Translations were delivered without pre-editing or post-editing. In 1972-

1973, Bozena Henisz-Dostert (now Bozena Thompson) conducted an evaluation and concluded that users were quite happy with raw MT.

"The users presented a rather satisfied group of customers, since 96 percent of them had or would recommend machine-translation services to their colleagues, even though the texts were said to require almost twice as much time to read as original English texts (humanly-translated texts also were judged to take longer to read, but only about a third longer), and that machine-translated texts were said to be 21 percent unintelligible. In spite of slower service than desired and a high demand on reading time, machine translation was preferred to human translation by 87 percent of the respondents if the latter took three times as long as the former. The reasons for the preference were not only earlier access, but also the feelings that the 'machine is more honest', and that since human labor is not invested it is easy to discard a text which proves of marginal interest. Getting used to reading machine-translation style did not present a problem as evidenced by the answers of over 95 percent of the respondents." (Henisz-Dostert, 1979, p. 206)

It is also interesting to compare the attitudes of the users of this service with attitudes of the translators mentioned above. Henisz-Dostert found that end-users were generally quite supportive, and would recommend the service to a friend, whereas Macklovitch found that professional translators were generally unwilling to continue using the service themselves, let alone recommend the service to a friend.

"A grateful word is in order on the users' attitudes, who were most cooperative and friendly, and interested in what was involved in machine translation. They showed their familiarity with the aberrations of the texts, some of which were considered quite amusing 'classics', e.g., 'waterfalls' instead of 'cascades' (the users asked that this not be changed!). Very commonly, and understandably, they were interested in improvements and offered many suggestions. An example of an extreme attitude on the part of one user in this respect was that of 'cheating' on the questionnaire by giving less positive answers than in oral discussions. When subsequently asked about this, he reacted with something like: 'I use it so much, I want you to improve it, and if



I show that I am satisfied, you will not work on it any more." (Henisz-Dostert, 1979, p. 151)

Why are these users so much more satisfied with MT than the translators involved in the Canadian government's trial of Weidner? We believe the difference is the application. It makes sense to offer end-users the option to trade off speed for quality, whereas it does not make sense to try to force translators to become post-editors. Consider the example of the crop varieties mentioned above. Many end-users might not be bothered too much by the extra articles because they can quickly skim past the mistakes, but the professional translator might feel quite differently about the extra articles because he or she will have to fix them.

### 6.3. More Modest Attempts to Appeal to the End-User

Consider, for example, the problem of reading email from other countries. The first author currently receives several messages a day in French such as the following:<sup>9</sup>

Pour répondre aux questions de Maurizio LANA, j'ai entendu dire de bonnes choses concernant le programme ALPS de Alan MELBY. C'est au moins le nom de sa société (ALPS) qui se trouve à Provo ou à Orem (Utah, USA). Il est également professeur de linguistique à la Brigham Young University (Provo, Utah).

It might be possible to provide a tool to help recipients whose French is not very good. Imagine that the email reader had a "cliff-note" mode that would gloss many of the content words with an English equivalent:

Pour répondre aux questions de Maurizio LANA,  
answer questions

j'ai entendu dire de bonnes choses concernant  
heard say good things concerning

Cliff-note mode could be used as a way to sneak technology into the email reader, just as Kay's workstation approach is a way of sneaking technology into the translator's workplace. At first, cliff-note mode would do little more than table lookup, but as time goes on, it might begin to look more and more like machine translation. In the future, for example, the system might be able to gloss the phrase *le nom de sa société* as *the name of his company*, but currently the system would gloss *nom* as *behalf* and *société* as *society*, because these translations are more common in the Canadian Hansards

<sup>9</sup> These messages usually arrive without accents.

(parliamentary debates), which were used to train the system.

Cliff-note mode stands up fairly well to the six desiderata. (a) It sets reasonable expectations. (b) It doesn't cost much to run. (c) It ought to be attractive to users. After all, those who don't like it, don't have to use it. (d) It is well-positioned to integrate the strengths of the machine (vocabulary) without competing with the strengths of the user (knowledge of function words, syntax and domain constraints). (e) It is so simple that user shouldn't have any trouble appreciating both the strengths as well as the weaknesses of the word-for-word approach. Finally, (f) the strategy of gradually introducing more and more technology is ideally suited for advancing the field toward desirable long-term goals.

### 7. Conclusion

We have identified six desiderata for a good niche application. Two marketing strategies appear to meet these six desiderata fairly well:

- (1) use the benefits of office-automation to sell to the professional translator, or
- (2) use the speed advantages of raw (or almost raw) MT to sell to the end-user who may not require high-quality.<sup>10</sup>

The discussion has stressed pragmatism throughout. The speech processing community, for example, has been somewhat more successful recently in making it possible to report crummy results. It is now quite acceptable in the speech community to work on very restricted domains (e.g., spoken digits, resource management (RM), airline traffic information system (ATIS)) and to report performance that doesn't compare with what people can do. No one would even suggest that a machine should be able to recognize digits as well as a person could. Because the field has taken a more realistic approach, the field now has a fairly good public image, and is appearing to be making progress at a reasonable rate:

"Slowly but surely, the technology is making its way into the real world." (Schwartz, 1991, *Business Week*, p. 130)

<sup>10</sup> Other possibilities have also been successful in the past. Xerox for example, has obtained impressive results by introducing a restricted language into the document preparation organization (Hutchins, 1988, p. 294). Smart Systems has also explored the use of a restricted language in organizations that generate text. Limiting the domain is another formula for success. The classic example is Meteo (Isabelle, 1984). Unfortunately, however, it is very hard to find very many other naturally-occurring limited domains that people care about, and consequently, this strategy is unlikely to be repeated very many times in the future.

But there was a time when speech researchers were much more ambitious. According to Klatt's review (Klatt, 1977), the first ARPA Speech Understanding project (Newell *et al.*, 1973) had the objective of obtaining a breakthrough in speech understanding capability that could then be used toward the development of practical man-machine communication systems. Even though Harpy (Lowerre and Reddy, 1980) did in fact exceed the specific goals of the project (e.g., accept a thousand word-vocabulary connected-speech with an artificial syntax and semantics and produce less than 10% semantic error in a few times real time on a 100 mips machine), it didn't matter because Harpy had failed to obtain the anticipated breakthrough. And consequently, funding in speech recognition and understanding was dramatically reduced over the following decade. When activity was eventually resumed many years later, the community had learned that it is ok to strive toward realistic goals, and that it can be dangerous to talk about breakthroughs.

### 7.1. The GU Experiment

The experience in machine translation is perhaps even more sobering. The 1954 Georgetown University (GU) experiment was a classic example of a success catastrophe. In Zarechnak's 1979 review of early work on machine translation, he recalled that the GU experiment was originally seen as a huge advance:

"The result of GU machine translation was given wide publicity in 1954 when it was announced in New York. The announcement was greeted by astonishment and skepticism among some people. L. E. Dostert summarized the result of the experiment as being an authentic machine translation which does not require pre-editing of the input nor post-editing of the output." (Zarechnak, 1979, p. 28)

But now, we can look back and see that the 1954 GU experiment probably did more harm than good by setting expectations at such an unrealistic level that they could probably never be met. Ten years after the GU experiment, the ALPAC report compared four then-current systems with the earlier GU experiment and suggested that there had not been much progress.

"The reader will find it instructive to compare the samples above with the results obtained on simple, or selected, text 10 years earlier (the Georgetown-IBM Experiment, January 7, 1954) in that the earlier samples are more readable than the later ones." (Pierce *et al.*, 1966)

Zarechnak, a member of the Georgetown effort, complained rather bitterly that the comparison was unfair. In reality, the 1954 GU experiment had been a canned demo of the worst kind, whereas the four systems developed during the 1960s were intended to handle large quantities of previously unseen text.

"When ten years later a text of one hundred thousand words was translated on a computer without being previously examined, one would expect a certain number of errors on all levels of operations, and the need for post-editing. The small text in 1954 has no such random data to translate." (Zarechnak, 1979, p. 56)

In fact, the ALPAC committee had also appreciated the "toy"-ish aspects of the 1954 GU experiment, but they did not feel that that was an adequate excuse. They criticized both the 1954 experiment as well as the four systems in question, the former for setting expectations unrealistically high, and the latter for failing to meet those expectations, unrealistic as they may be.

"The development of the electronic digital computer quickly suggested that machine translation might be possible. The idea captured the imagination of scholars and administrators. The practical goal was simple: to go from machine-readable foreign technical text to useful English text, accurate, readable, and ultimately indistinguishable from text written by an American scientist. Early machine translations of simple or selected text, such as those given above, were as deceptively encouraging as 'machine translations' of general scientific text have been uniformly discouraging." (Pierce *et al.*, 1966, pp. 23-24)

If expectations had been properly managed and the waters had not been poisoned by the 1954 GU experiment, it is possible that we would now look back on the MT effort during the 1960s from a much more positive perspective. In fact, one of the four systems in question later became known as *Systran*, and is still in wide use today. In this sense, early work on MT was much more successful than early work on Speech Understanding; the first ARPA Speech Understanding Project did not produce any systems with the same longevity as *Systran*.

For some reason that is difficult to understand, the two fields currently have entirely different public images; on the one hand, the laymen can readily recognize that it is extremely difficult for a machine to recognize speech, while, on the other hand, even the manager of a translation service will blindly accept the most

preposterous pretensions of practically any MT salesman. Perhaps we can change this perception if we succeed in focusing our attention on good applications of state-of-the-art (i.e., crummy) machine translation.

#### References

- Brown, P., Cocke, J., Della Pietra, S., Della Pietra, V., Jelinek, F., Lafferty, J., Mercer, R., Rossin, P. (1990). "A Statistical Approach to Machine Translation." *Computational Linguistics*, 16:2, pp. 79-85.
- Brown, P., Della Pietra, S., Della Pietra, V., Lai, J., Mercer, R. (1991) "An Estimate of an Upper Bound for the Entropy of English," submitted to *Computational Linguistics*.
- Church, K. and Gale, W. (forthcoming) "Concordances for Parallel Text" Seventh Annual Conference of the UW Centre for the New Oxford English Dictionary and Text Research, Oxford, England.
- Henisz-Dostert, -B. (1979) "Users' Evaluation of Machine Translation," in Bozena Henisz-Dostert, R. Ross Macdonald, Michael Zarechnak (eds), "Machine Translation," Mouton Publishers.
- Isabelle, P. (1984) "Machine Translation at the TAUM Group," in King, M. (ed.) *Machine Translation Today: The State of the Art*, Edinburgh University Press.
- Isabelle, P. and Bourbeau, L. (1985) "Taum-Aviation: Its Technical Features and Some Experimental Results," *Computational Linguistics*, 11:1, pp. 18-27.
- Johnson, R., King, M., and des Tombe, L. (1985) "Eurora: A Multilingual System under Development," *Computational Linguistics*, 11:2-3, pp. 155-169.
- Kay, M. (1980) "The Proper Place of Men and Machines in Language Translation," unpublished ms., Xerox, Palo Alto, CA.
- King, M. and Falkedal, K. (1980) "Using Test Suites in Evaluation of Machine Translation Systems," COLING, pp. 211-216.
- Lawson, V. (1984) "Users of Machine Translation System Report Increased Output," *Language Monthly*, 11, pp. 6-10.
- Lelirherger, J. and Bourbeau, L. (1988) *Machine Translation: Linguistic Characteristics of MT Systems and General Methodology of Evaluation*, John Benjamins Press.
- Lowerre, B. and Reddy, D. (1980) "The Harpy Speech Understanding System," in *Trends in Speech Recognition*, Prentice Hall, reprinted in Waibel, A. and Lee, K. (eds) (1990) *Readings in Speech Recognition*, Morgan Kaufmann Publishers, Inc., San Mateo, California.
- Kay, M. (1980) "The Proper Place of Men and Machines in Language Translation," CSL-80-11, Xerox PARC.
- Klatt, D. (1977) "Review of the ARPA Speech Understanding Project," *Journal of the Acoustical Society of America*, reprinted in Waibel, A. and Lee, K. (eds.) (1990) *Readings in Speech Recognition*, Morgan Kaufmann Publishers, Inc., San Mateo, California.
- Macklovitch, E. (1986) "MT Trial and Errors," presented at the International Conference on Machine and Machine-Aided Translation, April 7-9, Aston University, Birmingham, United Kingdom.
- Macklovitch, E. (1989) "An Off-the-Shelf Workstation for Translators," *Proceedings of the 30th American Translators Conference*, Washington DC, 1989.
- Macklovitch, E. (1991) "Evaluating Commercial MT Systems," paper presented at the Evaluators' Forum on MT systems, organized by ISSCO at Ste. Croix, Switzerland.
- Magnusson-Murray, U. (1985) "Operational Experience of a Machine Translation Service," in Lawson, V. (ed.) *Tools for the Trade, Translating and the Computer 5*, Alden Press, Oxford.
- Nagao, M. (1987) "Role of Structural Transformation in a Machine Translation System," in Nirenburg, S. (ed.) *Machine Translation: Theoretical and Methodological Issues*, Cambridge University Press, pp. 262-277.
- Nagao, M., Tsujii, JI., and Nakamura, JI. (1986) "Machine Translation from Japanese to English" *Proceedings of the IEEE*, 74:7, pp. 993-1012.
- Newell, A., Barnett, J., Forgie, J., Green, C., Klatt, D., Licklider, J., Munson, J., Reddy, D., and Woods, W. (1973) *Speech Understanding Systems: Final Report of a Study Group*, North-Holland/American Elsevier, Amsterdam.
- Pierce, J., Carroll, J., Hamp, E., Hays, D., Hockett, C., Oettinger, A., Perlis, A. (1966), "Language and Machines: Computers in Translation and Linguistics," also known as the ALPAC report, National Academy of Sciences Publication 416, Washington D.C.
- Schwartz, E. (1991) "A Computer that Recognizes its Master's Voice," *Business Week*, June 3, pp. 130-131.

- Shannon, C. (1951) "Prediction and Entropy of Printed English," *Bell Systems Technical Journal*, vol 30, pp 50-64.
- Tschira, K. (1985) "Looking Back at a Year of German-English MT with Logos," in Lawson, V. (ed) *Tools for the Trade, Translating and the Computer 5*. Alden Press, Oxford.
- Tucker, A B (1987) "Current Strategies in Machine Translation Research and Development," in Nirenburg, S. (ed) *Machine Translation: Theoretical and Methodological Issues*, Cambridge University Press, pp. 22-41.
- Van Sylpe, G. (1979) "Evaluation of the 1978 Version of the SYSTRAN English-French Automatic System of the Commission of the European Communities," *The Incorporated Linguist*, 18:3, pp. 86-89.
- Wagner, E. (1985) "Rapid Post-Editing of Systran," in Lawson, V. (ed.) *Tools for the Trade, Translating and the Computer 5*, Alden Press, Oxford.
- Zarechnak (1979) "This History of Machine Translation," in Bozena Henisz-Dostert, R. Ross Macdonald, Michael Zarechnak (eds), "Machine Translation," Mouton Publishers

# Gross-Grained Software Evaluation: The Natural Language Software Registry

Elizabeth A. Hinkelman  
Center for Information and Language Studies  
1100 E. 57th St., University of Chicago  
Chicago, IL 60637

## 1. Introduction

There are three reasons to perform evaluation of natural language processing software: to judge progress in the field as a whole, to judge the success of a particular theory of language processing, and to judge the appropriateness of the software for a particular application. In this presentation, I will discuss the role of the Natural Language Software Registry in evaluation efforts aimed at progress in the field as a whole. Particular software and theories may be considered, but with an eye toward establishing a base of quality NLP software for research purposes. Thus, emphasis will be laid on properties that are common to software regardless of the level or levels of linguistic analysis being performed. Research, engineering, and logistical issues affecting software reusability emerge.

The Natural Language Software Registry was recently established at the University of Chicago's Center for Information and Language Studies. Its purpose is to facilitate the exchange and evaluation of noncommercial and commercial software. The Registry is sponsored by the Association for Computational Linguistics. Projected Registry activities, pending funding, include

- solicitation and collocation of software descriptions
- distribution of descriptions using print and electronic media
- establishment of a distribution mechanism for software not otherwise be accessible
- coordination of detailed reviews of noteworthy software, to be published  
in *Computers and the Humanities*, *Computational Linguistics*, and other journals.
- participation in ongoing software evaluation efforts

The initial task has been the solicitation of reports from software developers both academic and commercial, with the aim of constructing a concise, uniform summary of software sources and capabilities. Such a summary (Hinkelman 1991b) serves not only as a guide for researchers in determining where to direct their their software development efforts, but also as an index of the state of the natural language processing endeavor. A pilot survey of 33 software items (as of March '91) has enabled us to better understand evaluation criteria that can be applied at this gross level, and supports a preliminary assessment of the state of natural language processing. This gross-granularity approach to evaluation is complemented by extensive testing and review of selected software. This paper emphasizes metrics that reflect on the reusability of software, for purposes beyond the immediate goal of the designer. It describes the pilot survey, its findings, and how a particular software review reflects upon it.

## Evaluation using the NL Software Registry

### 2. Solicitation of Software Descriptions

The Registry project has been extensively publicized, and software descriptions solicited at ACL'90 and COLING'90, ACH/ALLC'91 [Hinkelman 1991a], the *Finite String Newsletter*, and several electronic bulletin boards. Respondents conform to what may be termed an "extroversion factor": they represent commercial ventures, well-established community-minded research projects, and active electronic bulletin board participants. Commercial enterprises tend to omit fields reflecting system internals and potentially negative system features from their software descriptions. Some established research projects have mandates and mechanisms for distributing their software; the best examples of this are national projects such as Alvey. In the US researchers are not specifically supported in beta-testing and distribution of software, but projects such as Penman, Sneps, and Rhet have a record of doing so. They tend to provide very detailed descriptions. Individual electronic bulletin board participants vary greatly in the modesty and number of hidden assumptions of software descriptions. We hope to achieve more participation from major projects in the future, as the notion of reusability gains widespread acceptance.

Software registered included processors for speech, morphology, syntax, and knowledge representation; several large multicomponent projects, and applications software in the areas of spelling checkers, database interfaces, computer aided education (poetry), and miscellaneous tools for linguists. (Table 1.)

Survey questions were designed to reveal the capabilities and limitations of NLP software, along with the conditions under which it can be acquired. They presumed that software being registered belonged to core areas of natural language processing, and therefore contained some biases less appropriate to software ultimately classed as application of NLP techniques. The survey information breaks down into

- basic administrative parameters such as developers' addresses
- conditions on availability, such as fees, licences, and support provided
- description of system goals and underlying principles
- basic technical parameters, such as language of implementation
- design features and test set size

**Table 1. Major Software Categories**

2	speech signal processing systems
3	morphological analysis programs
5	syntactic parsers
1	knowledge representation system
10	multicomponent systems
12	applications: interlinear text, dialect analysis, spellchecking, etc.

## Evaluation using the NL Software Registry

It is the final category that has the most relevance in evaluation of software quality. It encompasses

- separation of code from natural language data
- embeddedness vs. independence of major modules
- extensibility of the system
- technical and theoretical limits to the range of natural languages accommodated
- number, size, and nature of test sets

The gross-grain approach to metrics for these attributes is triage, for non-numerical attributes. Items are divided into three categories-- yes, no, and an intermediate value. The interpretation of the intermediate value varies according to the parameter being evaluated. It may indicate uncertainty about a binary parameter, such as the availability of source code, or an intermediate value on a spectrum, such as degree of independence of modules.

The gross-grain metric used for test set size is powers-of-ten, made comparable across systems by assuming that there are approximately ten words per sentence, and at least ten sentences per message. Provisionally, a "concept" in knowledge representation is assumed to contain approximately as much structure as a sentence, and a knowledge representation problem about the complexity of a message. This metric makes it possible to compare systems on paper.

### 3. Quality of Software Registered

The survey results using these metrics are given below.

#### 3.1. Modularity

Researchers are often interested in linking experimental modules with other NLP components, especially with modules that generate the input desired for the experimental module. This leads to the "cut and paste" criterion for modularity, which asks whether major processing components can be extracted and used in combination with different components. (Data is another issue.) Extraction and recombination was not possible for over half of the registered descriptions.

Some of the failures of modularity are pragmatic in nature: modules are packaged behind a user interface and source code is not provided. For source code, the triage method reveals that source code is definitely available in one third of the cases. The seven "maybe" cases include one piece of source code that is exceptionally expensive, two of unresolved status, and "cliqueware"-- available to selected collaborators only.

For modularity, we find that multicomponent systems are often decouplable with difficulty or not at all-- a shocking fact a priori. The fact is that there is ongoing experimentation in the types of control structures that relate syntactic and semantic processing, and in semantic representations. Semantic representations in turn vary with the nature of back-end tasks: only within the subarea of database interfaces does a representation (SQL) emerge as the equivalent of p-code in programming language compilers. Modularity is thus an issue from both a research and engineering standpoint.

The "no" category also includes several applications programs which are simply intended to be run in isolation. One solution, as for PC Kimmo, is to provide both a version compiled with interpretive interface and one compiled as a library function. In the end, it is possible to cut and paste using some software not described as modular, by special arrangement with the software developer or because, in the case of Parlance, a runtime interface to the semantic interpreter is available. (Table 2.)

## Evaluation using the NL Software Registry

Table 2. Modularity

	yes	maybe	some	no
Source Code	12	7		14
Modularity	10		6	14
Cut and Paste	12		8	10

### 3.2. Extensibility

Researchers may well want to add functionality to programs they acquire, and in general this requires source code. The exception is a signal processing system to which the user can add macros. The more extensive systems are so baroque as to inhibit extension even by their own developers. The number of "maybe" answers here will be reduced by better phrasing on future questionnaires. (Table 3).

### 3.3. Range of Languages

Researchers may wish to experiment with other languages, or other target domains. This is not possible in packaged, commercial systems, nor in systems such as STEMMA whose algorithms incorporate information specific to a language. In general, systems are designed with clear partitioning of code and data (PROLOG programs are considered case by case.) (Table 4.) However, there are further considerations that limit the retargetability of software to additional languages and domains. Technical issues include choice of character sets and other orthographic assumptions; in

Table 3. Extensibility by Programmer

	yes	maybe	some	no
Source Code	12	7		14
A Priori Extensibility	14	15		5
Extensibility	8	8		18



## Evaluation using the NL Software Registry

particular, the vast majority of the registered programs are limited to 7-bit ASCII and therefore the Roman alphabet. Theoretical considerations include orientation towards specific properties of languages, such as agglutination, simple inflectional morphology, cross-serial dependencies, and so on. Furthermore, the logical form into which some multicomponent systems translate sentences may prove to be optimised for the natural language first targeted by system developers.

### 3.4. Test Sets

The survey collected an order-of-magnitude report of the size of test sets to which the software has been applied. Unfortunately answers did not always distinguish size of auxiliary data (lexicons, rule sets) from size of input data sets; these are noted with a small 'a'. We expect that the ratio of auxiliary and test data would be a very good measure of system robustness, were it available. Likewise unavailable is detailed information on how the test sets were chosen or the actual quality of the results. (Table 5, 6.)

Because we are evaluating the field as a whole rather than individual NLP systems, we represent the systems with a capital letter rather than naming them. A few systems have been tested on several languages, as noted. For some systems additional information was available or could be inferred, as noted with an 'i'. For instance, although system 'D' was reported as performing message understanding, it likely does not handle any extrasentential phenomena and therefore would be better reported in terms of sentence semantics.

### 4. A Case Study: PC-Kimr.o

To date, the Registry has completed one extensive software review [Olsen 1991], that of the Summer Institute for Linguistics' implementation [Antworth 1990] of finite-state morphological analysis [Koskenniemi 1983, 1984]. The review's conclusions, based on an attempt to apply the program to a large-scale text database, are compared here with the description submitted to the Registry and with two other implementations [Karttunen 1990, Genikomsidis 1988] of the same theory.

The self-description in the Registry correctly reported good separation of code and data, and that major modules are callable (compiled as library functions, in addition to the interactive interface.) Extensibility was reported as possible by modifying C code. In practice, while the modules are well chosen from the point of view of a descriptive linguist, it was difficult to modify the code to produce closely related modules (recognition without a lexicon, generation using the lexicon, both provided by [Genikomsidis 1988].) It was possible to make other modifications with relative ease.

Table 4. Retargetability to other Languages and Domains

	yes	maybe	no
Retargetability	21	5	8

## Evaluation using the NL Software Registry

The reported range of languages handled by the system is "any". In practice, this involves inelegancies such as duplication of some portions of the lexicon. Furthermore, there are language groups such as Algonkian and Eskimo which are better modelled with a full context-free mechanism. Similarly, there are arguably rare occurrences in English that would require a context-free mechanism. The self-report thus makes assumptions which even its authors would be quick to qualify, but which would not be obvious to an outsider.

The reported test sets were nine languages at 10-100 lexicon items. The reviewers attempted to load a French lexicon of 8000 words, and encountered a bug (rather than a design limitation) which was promptly fixed by SIL. Coverage of French was then limited more by the lexicon-use requirement rather than the provided rule set, which thus remained incompletely tested.

The system had one large disadvantage that was not detectable from the self report: the interface. Users were expected to enter production rules as state tables for finite-state machines, a tedious task with perhaps some minor instructional value to the novice. This is true for several other versions of the technology in widespread circulation ([Genikomsidis 1988] inter alia), due no doubt to the presence of this deficiency in their common ancestor. While an alpha version of a production-rule interpreter has since become available, the kimmo family of programs stands testimony to the consequences of setting a bad precedent.

The self-report was thus substantially accurate, even modest, with the exception of theoretical and interface limits. One would expect this to be typical of academic systems, whereas one would expect better interfaces and less modest claims in commercial self-reports. One way to improve self-reports would be to include a small I/O sample. Other information that will be sought in future versions includes:

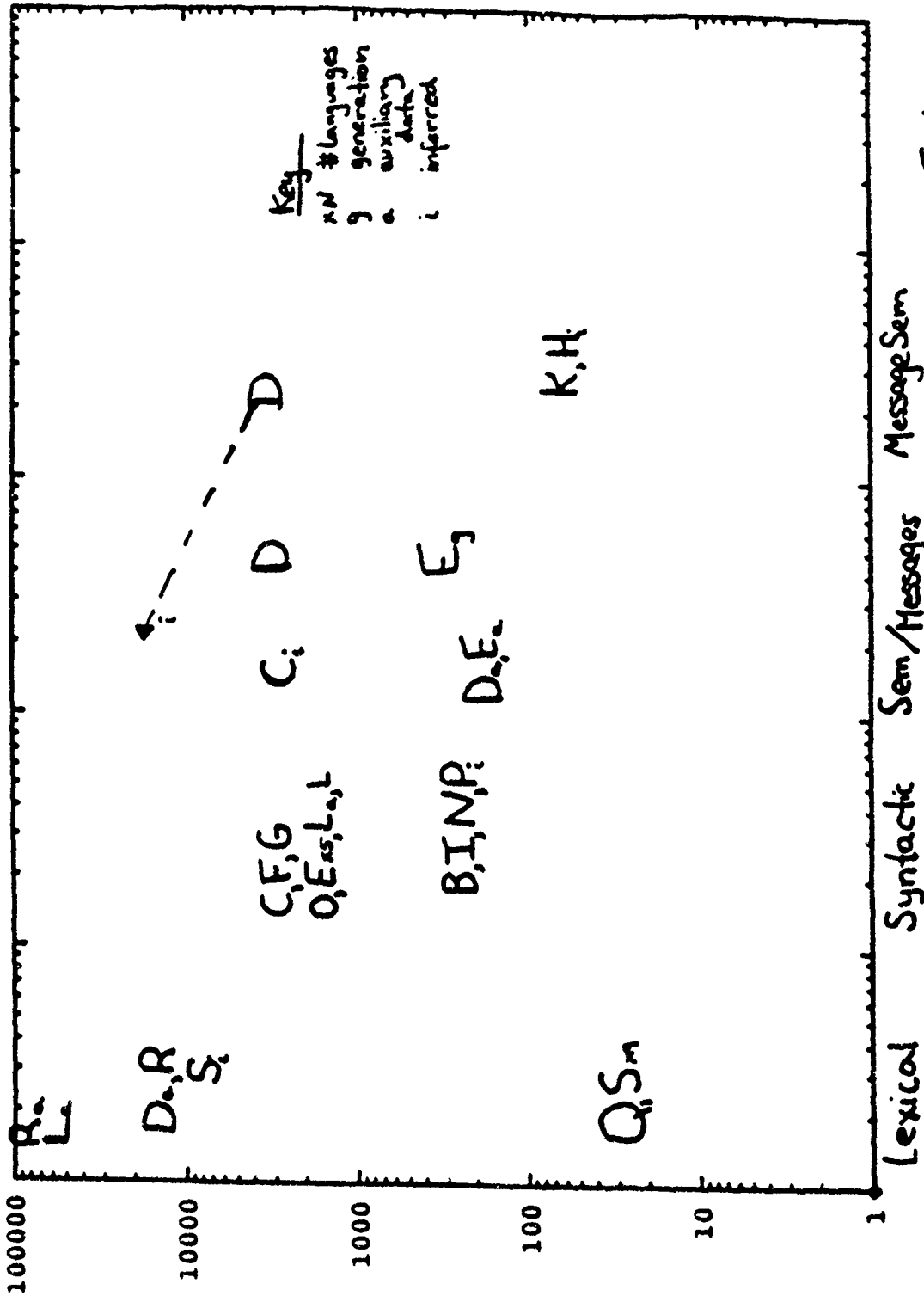
- number of work years the project represents
- some questions confuse auxiliary data sets with test sets
- specifications on individual major modules of multimodule systems
- character set used for text representation

## 5. Summary

Although the gross descriptions provided by software developers are limited in accuracy and detail, they have shown us the state of the practice with regard to several important parameters. It has pointed out the failure of the community to deal with orthography in any general way; its success in providing some tools, and occasionally broad coverage; and the need for further research on modular semantic processing. The large-grain metrics and parameters used are perhaps the only ones justified by our method of collecting developer-supplied information. Through the life of the Registry, we hope to see refinement of the metrics and of the software to which they are applied.

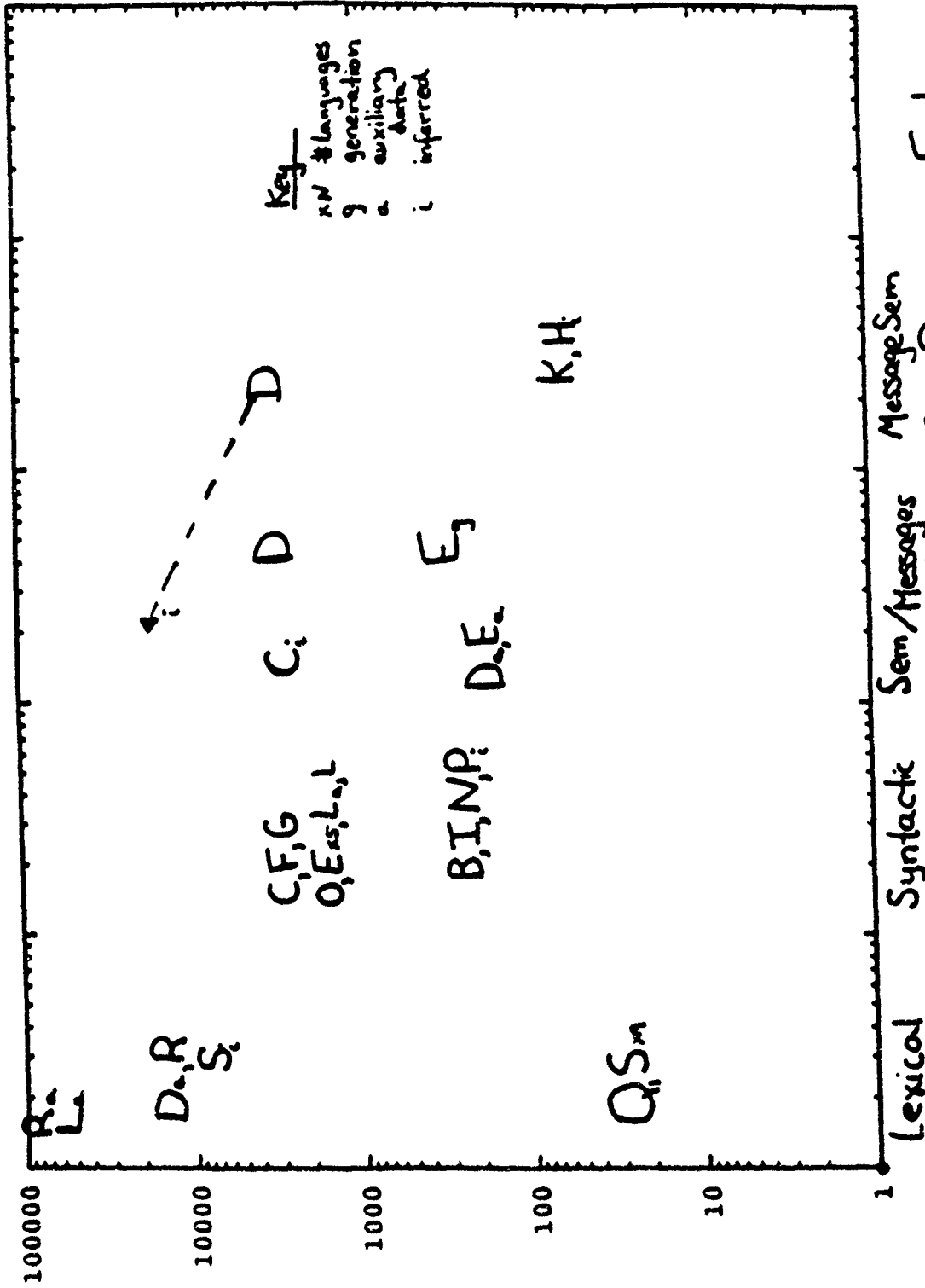
## 6. About the Author

Elizabeth Hinkelman completed her dissertation in computer science (natural language pragmatics) in 1989 at the University of Rochester. She became an Ameritech research fellow at the University of Chicago's Center for Information and Language Studies, where she has pursued applications of natural language processing to information retrieval and large literary databases. She is liason to the Association for Computational Linguistics for, and founder of, the Natural Language Software



Key  
 xN # languages  
 g generation  
 a auxiliary  
 d data  
 i inferred

Table 5. Test Set Sizes for Registry Entries



Key  
 xN # languages  
 g generation  
 a auxiliary  
 data  
 i inferred

Table 5. Test Set Sizes for Registry Entries

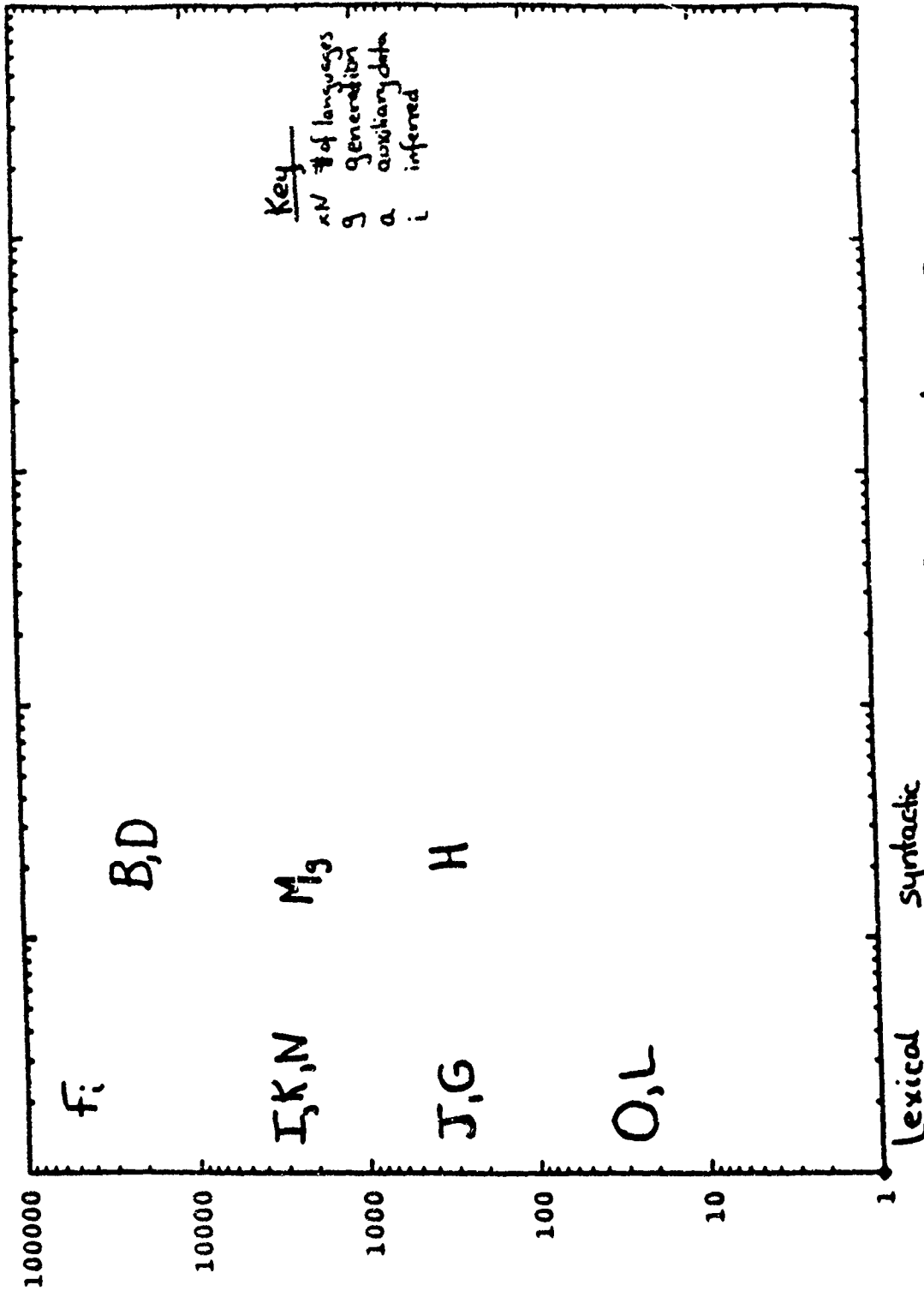


Table 6. Test Set Sizes: Utility Programs



**MISSION  
OF  
ROME LABORATORY**

*Rome Laboratory plans and executes an interdisciplinary program in research, development, test, and technology transition in support of Air Force Command, Control, Communications and Intelligence (C<sup>3</sup>I) activities for all Air Force platforms. It also executes selected acquisition programs in several areas of expertise. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C<sup>3</sup>I systems. In addition, Rome Laboratory's technology supports other AFSC Product Divisions, the Air Force user community, and other DOD and non-DOD agencies. Rome Laboratory maintains technical competence and research programs in areas including, but not limited to, communications, command and control, battle management, intelligence information processing, computational sciences and software producibility, wide area surveillance/sensors, signal processing, solid state sciences, photonics, electromagnetic technology, superconductivity, and electronic reliability/maintainability and testability.*