Evaluating Message Understanding Systems: An Analysis of the Third Message Understanding Conference (MUC-3)

Nancy Chinchor*
Science Applications International Corp.

Lynette Hirschman†
Massachusetts Institute of Technology

David D. Lewis‡
University of Chicago

This paper describes and analyzes the results of the Third Message Understanding Conference (MUC-3). It reviews the purpose, history, and methodology of the conference, summarizes the participating systems, discusses issues of measuring system effectiveness, describes the linguistic phenomena tests, and provides a critical look at the evaluation in terms of the lessons learned. One of the common problems with evaluations is that the statistical significance of the results is unknown. In the discussion of system performance, the statistical significance of the evaluation results is reported and the use of approximate randomization to calculate the statistical significance of the results of MUC-3 is described.

1. MUC-3 Purpose, History, and Methodology

1.1 Introduction

The Third Message Understanding Conference (MUC-3) represented a significant step for the message processing community and for computational linguistics as a whole. The conference brought together 15 systems that were tested on a naturally occurring corpus of newswire reports on terrorism in Latin America. The systems were evaluated on their ability to extract significant information from the newswire reports in the form of "templates" summarizing who did what to whom. We can enumerate the successes of the conference in many dimensions: the number of participating systems (15 systems, up from 8 at the previous Message Understanding Conference, MUCK-II), the scale of the application (100 times more text than the previous conference), the rigorous definition of the evaluation method (including automated and interactive scoring procedures), and the cooperative framework that enabled the participants to develop both the training data and the evaluation procedures. These are significant accomplishments in a field that has only recently begun to address system evaluation issues.

The MUC-3 conference has already been described in a conference proceedings (*Proceedings of the Third Message Understanding Conference (MUC-3)*, 1991) and in an *AI*

* 10260 Campus Point Drive M/S A2-F, San Diego CA 92121
† The MITRE Corporation, M/S K329, 202 Burlington Road, Bedford, MA 01730.
‡ AT&T Bell Laboratories, Murray Hill NJ 07974

Magazine article (Lehnert and Sundheim 1991). However, both of these reports were produced simultaneously with the system-building and evaluation activities of MUC-3. It became clear that more time and additional analysis were needed to digest the findings of MUC-3 and to make them accessible to the broader computational linguistics community. This paper addresses these issues, providing an overview of the conference, the participating systems, the evaluation methodology, and system performance. We also present information on the statistical significance of the reported test results and an analysis of what we have learned, both about message understanding and about system evaluation.The authors participated extensively in the conference as consultants, but were not associated with a specific system undergoing formal evaluation.

1.2 Background

The Third Message Understanding Conference evolved out of the two earlier message understanding conferences held in 1987 and 1989. Prior to 1987, formal evaluation of message processing (or text understanding) systems was virtually nonexistent. The dominant research paradigm was to build a natural language system and then to debug it on some set of examples. Sometimes the examples were made up for the purpose of debugging. In other cases, the system was debugged on a corpus of naturally occurring sentences. However, there was no notion of a "blind test" using previously unseen test data.

The first Message Understanding Conference (MUCK) was held at the Naval Command, Control, and Ocean Surveillance Center RDT&E Division (NRaD)[1] in May 1987. Organized by Beth Sundheim, its goal was a qualitative evaluation of the state of the art in message understanding. Six sites agreed to demonstrate running systems at the conference. A small set of training messages (ten tactical naval operations reports on ship sightings and engagements) was distributed, plus several hundred additional messages of various types, provided for general background. At the conference, each actively participating group demonstrated its system running on one of the twelve training messages. In addition, two previously unseen messages were distributed at the conference and all active participants were asked to get their systems running on these two new messages. The sites reported on what they needed to do to get their systems to handle the test messages. However there was no "task" associated with the messages and no quantitative evaluation procedure.

A second conference, MUCK-II, was held in May 1989 at NRaD. Based on their experience at the first MUCK, participants requested that a specific task be provided, along with an evaluation procedure to determine whether a system's answers were right or wrong. The domain chosen for MUCK-II was another type of tactical naval operations message, also about ship sightings and engagements. The task consisted of filling in templates for incidents mentioned in each report. The template types included "DETECT," "ATTACK," and so on. For each template there was a set of slots describing the actors involved in the incident, as well as the time, date, outcome, and other related information. Eight sites submitted systems to be evaluated. There was a larger amount of training data provided (105 messages) and two rounds of test data (20 messages, run first "blind" and then with system fixes, followed, just before the conference, by an additional 5 messages for a second blind test). In addition to the test data, NRaD furnished a list of specialized naval terminology, a hierarchy of terms for the relevant portion of the naval domain, and extensive documentation on filling

[1] Formerly the Naval Ocean Systems Center (NOSC).

Abbreviation	Participating Site	Location
ADS	Advanced Decision Systems	Mountain View, CA
BBN	BBN Systems and Technologies	Cambridge, MA
GE	General Electric Research and Development Center	Schenectady, NY
GTE	GTE Government Systems	Mountain View, CA
Hughes	Hughes Research Laboratories	Malibu, CA
ITP	Intelligent Text Processing, Inc.	Santa Monica, CA
LSI	Language Systems, Inc.	Woodland Hills, CA
MDESC	McDonnell Douglas Electronics Systems Company	Santa Ana, CA
NYU	New York University	New York, NY
PRC	PRC, Inc.	McLean, VA
SRI	SRI International	Menlo Park, CA
Synch/UMD	Synchronetics, Inc. and University of Maryland	Baltimore, MD
U Mass	University of Massachusetts	Amherst, MA
UNL/USL	University of Nebraska-Lincoln and University of Southwestern Louisiana	Lincoln, NE and Lafayette, LA
Unisys	Unisys Center for Advanced Information Technology	Paoli, PA

Figure 1
MUC-3 participants.

templates. One portion of the evaluation that was not completely determined before the conference was a scoring procedure. Scoring guidelines were provided but sites scored their own runs by hand. This led to substantial variation in how answers were scored among sites, and resulted in a consensus that a more rigorous and objective scoring procedure was needed.

Out of these experiences at the earlier message understanding conferences came MUC-3, held in May 1991. The major changes with MUC-3 were:

1. A new, more general domain, namely Latin American terrorist activity, with text originating from foreign news sources.
2. A ten-fold increase in the number of training documents (1,300), and more than a hundred-fold increase in number of words of text (approximately 400,000 words of text, with a vocabulary of 18,000 words).
3. An automated, interactive scoring program and a rigorous definition of the scoring procedure, to promote bias-free, reproducible scores and to allow off-site testing.
4. A broadening of the task to require distinguishing relevant from irrelevant messages.

1.3 Participating Systems

Fifteen sites participated in MUC-3, as shown in Figure 1. The official results of the sites on a test of 100 previously unseen messages are summarized in the plots in Figures 2 and 3, showing recall versus precision and recall versus overgeneration. These metrics are defined and discussed in Section 2.3. Note that perfect performance would be at 100% recall and 100% precision in Figure 2, whereas, in Figure 3, it would be at 100% recall and 0% overgeneration. The systems are listed below and described briefly in Section 3.2. The reader should consult the conference proceedings for detailed system

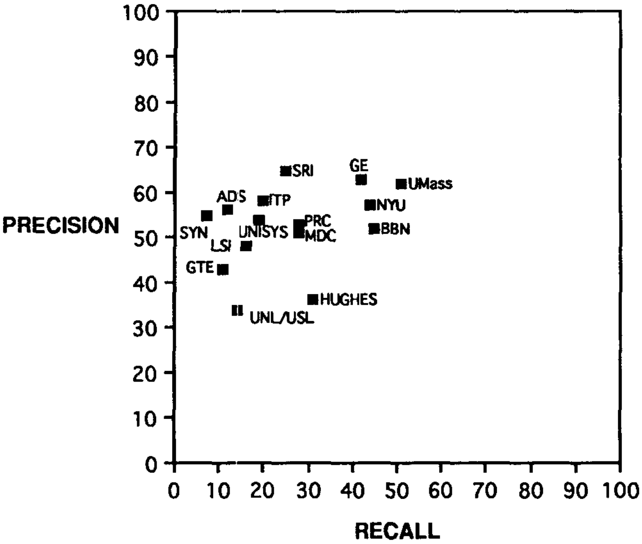


Figure 2
Recall vs. precision.

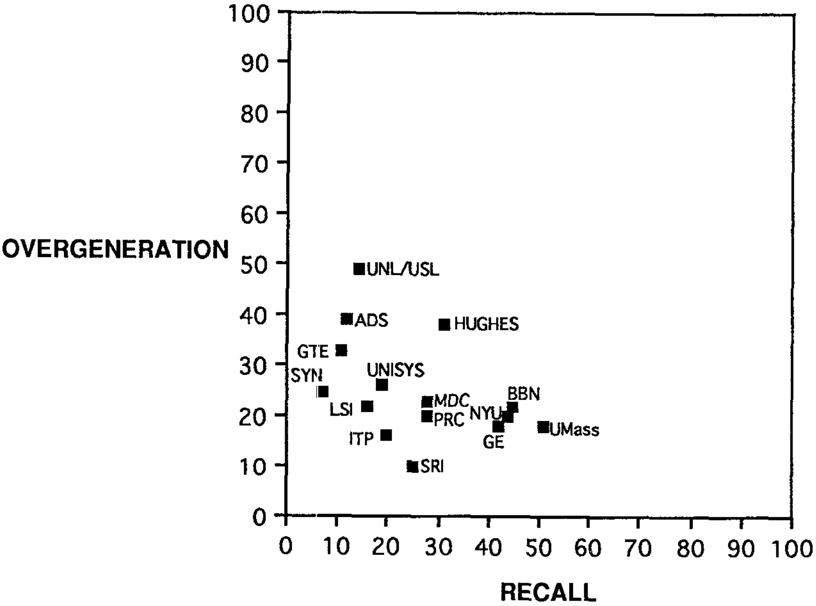


Figure 3
Recall vs. overgeneration.

descriptions and references. For descriptive purposes, we can group the systems into three broad classes:

Pattern-Matching Systems
These systems were characterized by fairly direct mappings from text to fillers, without the construction of elaborate intermediate structures. The mapping methods used varied widely. Some treated a text as an unordered set of words, as in traditional text categorization techniques from information retrieval (Lewis 1991). Slot fillers were defined in terms of the presence of words, Boolean combinations of words, or weighted combinations of words. Other methods scanned text for patterns specified in some extension of the language of regular expressions and produced corresponding fillers. Some systems also used hierarchies of word-based concepts or patterns defined in terms of other patterns.

While many groups used pattern-matching as an aid to producing more structured analyses, five groups used pattern-matching exclusively. For two groups (Hughes and UNL/USL), use of pattern-matching techniques was the focus of research. Three other groups (ADS, MDESC, and Unisys) implemented a pattern-matching front end with the intention of integrating it with a linguistically based component. However, for reasons of time, the linguistic component was not included in the MUC-3 evaluation.

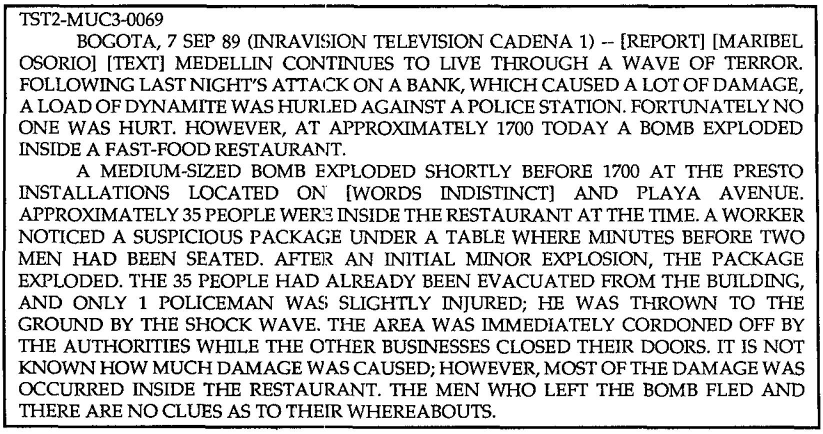
Syntax-Driven Systems
A second group of systems processed the messages by first obtaining a syntactic representation of sentences from the text, which was used as input to semantics and subsequent processing. This group included BBN, ITP, LSI, NYU, PRC, and SRI. These systems varied in their completeness of parsing (from partial parsing to full parsing), their use of feedback from semantic and pragmatic processing to influence syntactic parsing, and in their use of pattern-matching to supplement other forms of analysis.

Semantics-Driven Systems
The third group of systems were guided by semantic predictions and evolving semantic structures. These mechanisms drove varying amounts of syntactic analysis, ranging from limited phrase parsing to syntactically rich partial parsing, in support of the semantic predictions. The systems also used pattern-matching techniques heavily, but with a closer coupling to a domain model than the systems classified as pattern-matching systems. We put the systems from GE, GTE, Synch/UMD, and U Mass in this group.

1.4 Evaluation Methodology

1.4.1 Introduction to the MUC-3 Task. The task for the MUC-3 evaluation was to extract data about terrorist incidents from newswire articles. The extracted data were put into simulated database records defined by a template. The template database slots described important aspects of those incidents, such as the type of incident, the targets, perpetrators, date, location, and effects. An automated system that could perform this task would be useful in an environment where many incoming messages make it too expensive and time-consuming for personnel to do the data extraction and quality control.

The specific task domain for MUC-3 was restricted to terrorist acts involving nine Latin American countries. Terrorist acts were defined as violent acts perpetrated with political aims and a motive of intimidation. These acts could be perpetrated by a terrorist or guerrilla group, the government or the military, or by unknown individuals. The targets excluded terrorist or guerrilla groups, the military, and police; attacks



TST2-MUC3-0069

BOGOTA, 7 SEP 89 (INRAVISION TELEVISION CADENA 1) -- [REPORT] [MARIBEL OSORIO] [TEXT] MEDELLIN CONTINUES TO LIVE THROUGH A WAVE OF TERROR. FOLLOWING LAST NIGHT'S ATTACK ON A BANK, WHICH CAUSED A LOT OF DAMAGE, A LOAD OF DYNAMITE WAS HURLED AGAINST A POLICE STATION. FORTUNATELY NO ONE WAS HURT. HOWEVER, AT APPROXIMATELY 1700 TODAY A BOMB EXPLODED INSIDE A FAST-FOOD RESTAURANT.

A MEDIUM-SIZED BOMB EXPLODED SHORTLY BEFORE 1700 AT THE PRESTO INSTALLATIONS LOCATED ON [WORDS INDISTINCT] AND PLAYA AVENUE. APPROXIMATELY 35 PEOPLE WERE INSIDE THE RESTAURANT AT THE TIME. A WORKER NOTICED A SUSPICIOUS PACKAGE UNDER A TABLE WHERE MINUTES BEFORE TWO MEN HAD BEEN SEATED. AFTER AN INITIAL MINOR EXPLOSION, THE PACKAGE EXPLODED. THE 35 PEOPLE HAD ALREADY BEEN EVACUATED FROM THE BUILDING, AND ONLY 1 POLICEMAN WAS SLIGHTLY INJURED; HE WAS THROWN TO THE GROUND BY THE SHOCK WAVE. THE AREA WAS IMMEDIATELY CORDONED OFF BY THE AUTHORITIES WHILE THE OTHER BUSINESSES CLOSED THEIR DOORS. IT IS NOT KNOWN HOW MUCH DAMAGE WAS CAUSED; HOWEVER, MOST OF THE DAMAGE WAS OCCURRED INSIDE THE RESTAURANT. THE MEN WHO LEFT THE BOMB FLED AND THERE ARE NO CLUES AS TO THEIR WHEREABOUTS.

Figure 4
Example of a MUC-3 message.

on these organizations or their members were considered acts of guerrilla warfare. Figure 4 shows an example of a MUC-3 message as excerpted in the Foreign Broadcast Information Service (FBIS) Daily Reports. The original source of the example is Inravision Television Cadena 1 as shown in the dateline. FBIS was the secondary source for all of the articles used in MUC-3.

The message in Figure 4 describes three violent acts. The hurling of dynamite at the police station was not considered a relevant terrorist act because the police were the target and no civilians were involved. The attack on the bank is referred to briefly, while the bombing of the restaurant is quite detailed. The erroneous phrase "was occurred" appeared in the original message text. The templates in Figure 5 are the *answer key* templates for the attack on the bank and the bombing of the fast-food restaurant.[2]

1.4.2 Gathering and Preparing Texts for the Training and Test Corpora. The texts in the MUC-3 corpus were gathered using a keyword query on an electronic database containing articles in message format from open sources worldwide. Most of the articles in the MUC-3 corpus were translated from Spanish sources by FBIS.

The keyword query for choosing texts for the MUC-3 corpus was a two-part query using country or nationality name and inflected forms of common words associated with terrorist acts. The nine countries of interest were Argentina, Bolivia, Chile, Colombia, Ecuador, El Salvador, Guatemala, Honduras, and Peru. The word list used for selecting messages having to do with terrorism was based on the verbal and nominal forms *abduct, abduction, ambush, arson, assassinate, assassination, assault, blow [up], bomb, bombing, explode, explosion, hijack, hijacking, kidnap, kidnapping, kill, killing, murder, rob, shoot, shooting, steal,* and *terrorist.*

Messages fulfilling both parts of the query were downloaded from the mixed case electronic source into uppercase ASCII format.[3] The messages were prepared for use

[2] The answer key contains all possible correct ways of filling in the template. Alternative fillers are separated by slashes, and optional fillers are preceded by question marks.

[3] The change was a consequence of the downloading. It would be preferable to preserve the case of the original text. However, the capability to process both mixed case and uppercase only is required for

0. MESSAGE ID	TST2-MUC3-0069
1. TEMPLATE ID	1
2. DATE OF INCIDENT	(06 SEP 89) / (06 SEP 89 - 07 SEP 89)
3. TYPE OF INCIDENT	ATTACK
4. CATEGORY OF INCIDENT	? TERRORIST ACT
5. PERPETRATOR: ID OF INDIV(S)	-
6. PERPETRATOR: ID OF ORG(S)	-
7. PERPETRATOR: CONFIDENCE	-
8. PHYSICAL TARGET: ID(S)	"BANK"
9. PHYSICAL TARGET: TOTAL NUM	1
10. PHYSICAL TARGET: TYPE(S)	FINANCIAL: "BANK"
11. HUMAN TARGET: ID(S)	-
12. HUMAN TARGET: TOTAL NUM	-
13. HUMAN TARGET: TYPE(S)	-
14. TARGET: FOREIGN NATION(S)	-
15. INSTRUMENT: TYPES(S)	-
16. LOCATION OF INCIDENT	COLOMBIA: MEDELLIN (CITY)
17. EFFECT ON PHYSICAL TARGET(S)	SOME DAMAGE: "BANK"
18. EFFECT ON HUMAN TARGET(S)	-

0. MESSAGE ID	TST2-MUC3-0069
1. TEMPLATE ID	2
2. DATE OF INCIDENT	07 SEP 89
3. TYPE OF INCIDENT	BOMBING
4. CATEGORY OF INCIDENT	TERRORIST ACT
5. PERPETRATOR: ID OF INDIV(S)	"TWO MEN" / "MEN"
6. PERPETRATOR: ID OF ORG(S)	-
7. PERPETRATOR: CONFIDENCE	-
8. PHYSICAL TARGET: ID(S)	"FAST-FOOD RESTAURANT" / "PRESTO INSTALLATIONS" / "RESTAURANT"
9. PHYSICAL TARGET: TOTAL NUM	1
10. PHYSICAL TARGET: TYPE(S)	COMMERCIAL: "FAST-FOOD RESTAURANT" / "PRESTO INSTALLATIONS" / "RESTAURANT"
11. HUMAN TARGET: ID(S)	"PEOPLE" "POLICEMAN"
12. HUMAN TARGET: TOTAL NUM	36
13. HUMAN TARGET: TYPE(S)	CIVILIAN: "PEOPLE" LAW ENFORCEMENT: "POLICEMAN"
14. TARGET: FOREIGN NATION(S)	-
15. INSTRUMENT: TYPES(S)	*
16. LOCATION OF INCIDENT	COLOMBIA: MEDELLIN (CITY)
17. EFFECT ON PHYSICAL TARGET(S)	SOME DAMAGE: "FAST-FOOD RESTAURANT" / "PRESTO INSTALLATIONS" / "RESTAURANT"
18. EFFECT ON HUMAN TARGET(S)	INJURY: "POLICEMAN" NO INJURY: "PEOPLE"

Figure 5
Filled templates for the example MUC-3 message.

by creating or augmenting the dateline and text type information at the front of the article, and then removing the original headers and routing information. Exact duplicate messages were removed. Other minor preparations included modifying the

real-world applications.

articles to compensate for control characters and punctuation (brackets and exclamation marks) lost during the download, removing idiosyncratic features of the texts (including parentheses added by the transcribers to mark uncertainties), and improving readability by double-spacing between paragraphs.

The downloaded corpus of over 1,600 messages was divided into one development corpus containing 1,300 messages and 3 test corpora each containing 100 messages. Neither the chronological order of messages nor the fact that the same real-world event was described from different perspectives in different messages was considered when separating the texts into development and test corpora. The choice of messages for the test sets was based solely on the frequency with which incidents concerning a given country were represented. Unique identifiers were assigned to each message in the MUC-3 corpus; for example, DEV-MUC3-0001 indicates the first development message for MUC-3, and TST2-MUC3-0050 indicates the 50th message in the official (phase two) test set. The first and second 100-message test sets were used for the two phases of MUC-3, the dry run and the official test, respectively. The third test set of over 100 messages was saved for use in MUC-4.

1.4.3 Nature of the Original Textual Data. The original textual data consisted of newspaper stories, radio and television broadcasts, speeches, interviews, news conference transcripts, and communiqués. The average message length was 12 sentences and the average sentence length was 27 words.[4] The texts varied widely in complexity, style, and richness of information. Newspaper stories were written in normal reporting style containing much factual information and frequent long, complex sentences. The stories contained numerous quotes as well as excerpts from all of the other types of textual data.

Radio and television broadcasts consisted of transcribed utterances that contained commentary and/or excerpts from reports by correspondents and other sources. Speeches were in transcribed form and contained rhetorical and colorful language with heavy use of figures of speech. Speeches were of interest because they contained metaphorical uses of some "perpetration" verbs, for example, "philosophers...have launched a new attack" and "...destroy the economic infrastructure." Some excerpts of speeches appeared embedded in news reports. Interviews with single or multiple interviewers appeared as entire messages or as excerpts embedded in news reports. Infrequently, messages contained transcripts of news conferences with moderator, guests, and reporters speaking. Rebel communiqués often contained lists of demands, allegations, and intentions. Communiqués sometimes contained lines identifying date and place of origin and signatures of their originators. Some of these communiqués were embedded in news reports complete with commentary, date, location, and signatures. The discourse styles and sentence structures varied considerably across all of these types of textual data.

The texts of all types contained numerous anomalies. There were words and passages marked as "indistinct" in transcripts (e.g., first sentence in second paragraph of Figure 4) and passages marked as "omitted." Articles were sometimes split into message-length segments of approximately two pages each and were annotated to indicate continuation. The texts contained nonstandard constructions resulting from the translation of Spanish to English and also contained unflagged errors, including typographical errors, misspellings, omitted words, grammatical errors, and punctuation errors. In addition, the texts posed other challenges including the frequent use of

[4] Additional statistics concerning the MUC-3 corpus are presented elsewhere (Hirschman 1991b).

Spanish names, untranslated words, and Spanish acronyms followed or preceded by the corresponding full phrase in English or Spanish.

1.4.4 Nature of the Template Fills. The MUC-3 template fill task was complex not only because of the richness of the textual data but also because of the variety of information required to fill template slots and because of the interdependencies among the slot fillers. The templates contained 18 slots. The *message-id* and *template-id* slots identified the template. If the message did not contain information on terrorist events, it was considered irrelevant. In this case, only an empty template was generated with just the *message-id* slot filled. If the message was relevant, there could be one or more templates generated for the message. The possibility that a message contained more than one relevant incident made the MUC-3 task considerably more difficult than a task requiring only one template per message.

To determine the relevance of an incident in a message, a system needed to determine the dates of any possible terrorist acts, discriminate between old and new information, discriminate between vague and specific descriptions, determine what country the perpetrators and/or targets belonged to, discriminate between terrorism and either general criminal activity or guerrilla warfare, and determine whether the incident was actual, attempted, or threatened as opposed to planned, denied, or hypothetical.

To fill the slots in the template, the systems needed to isolate the incidents in each message and to extract information about the associated perpetrators and targets. Aspects of the terrorist action such as the date, location, instruments used, and the effects on the targets also had to be extracted. In order to extract the data and place it in the template correctly, systems had to resolve ambiguous information, infer information from vague references, categorize some of the information into predetermined classes of entities, put data into canonical form, and show links among certain data items.

The template slots fell into formal categories depending on the nature of correct fillers. A slot filler could be a string directly extracted from the text (e.g., "FAST-FOOD RESTAURANT"), a member of a finite set of fills (e.g., "TERRORIST ACT"), a numerical response (e.g., "36"), a specially formatted response such as a date or location (e.g., "07 SEP 89"), or a null value (e.g., "-" or "*") that indicated that no information was available or required for the incident.

The slots that took fillers from a finite set could either take one filler for each template (*message-id, incident-type,* and *incident-category* slots), several fillers for each template with all of the fillers being different (*instrument-type* slot), or several fillers for each template with multiple instances of the same filler allowed (*perpetrator-confidence, physical-target-type, human-target-type, foreign-nation, physical-target-effect,* and *human-target-effect* slots). The latter slots required that their filler(s) be cross-referenced to the filler(s) of another slot to capture interdependencies. For example, in template 2 of Figure 5, the *human-target-type* "LAW ENFORCEMENT" is cross-referenced to the *human-target-id* "POLICEMAN." There could be multiple instances of "LAW ENFORCEMENT" in the *human-target-type* slot that refer to different *human-target-id*s.

For each type of incident, there was a different set of slots that were allowed to be filled. For example, *physical-target* slots were not filled for MURDER incidents, since physical damage accompanying a murder was defined to be part of a separate ATTACK. The *effect-on-human-target* slot was required to be blank as well, since this information was redundant for the MURDER incident type.

The template design for MUC-3 structured the template in such a way that the filling of the slots was a well-defined but complex problem. In addition to the features of template design mentioned above, optional templates and optional and alternative slot fillers were specified in the answer keys to capture borderline cases. The template design provided several interesting problems for the development of scoring algorithms, scoring guidelines, and answer keys.

1.4.5 Generating Answer Keys. To evaluate system performance on the test sets, the systems' responses to the template fill task were compared with the responses in answer keys using a semi-automated scoring system developed for MUC-3. NRaD generated the answer keys for the MUC-3 test sets. However, the development messages also required answer keys for use as training material. Early in the evaluation schedule, NRaD completed the answer key for the first 100 development messages, and the participants completed the answer keys for the other 1,200 development messages. Participants were assigned blocks of 75 messages that overlapped partially with blocks from other sites. Sites had to agree on templates for those overlapping messages. The purpose of the overlapping blocks was to encourage uniformity of the answer keys for the development messages. In actuality, template design, the development of guidelines for filling templates, and the generation of development set answer keys were all interdependent activities. The answer keys for the development corpus had to be updated whenever there were changes in template design or in the guidelines for filling templates.

During the generation of the official answer key, an attempt was made to measure the consistency of human template filling. Two evaluators at NRaD prepared answer keys for the official test set in the following manner. Each evaluator read the test messages and determined which templates were to be generated for the messages. The evaluators discussed the template-level decisions concerning relevancy of the violent incidents mentioned in the messages, came to an agreement as to how many templates were to be generated for each message, and then independently generated template fills for the relevant incidents. One of the evaluators made a final decision as to the contents of the official answer key based on the two independently produced sets of templates.The other evaluator's key was then scored against the official answer key and received scores of 87% recall and 91% precision as shown in the MATCHED/MISSING row of Figure 8.[5] These scores reflect the degree to which well-trained evaluators agree. They also give some idea of what generous upper bounds on system performance might be. It should be noted that the evaluators took several days to complete the template-filling task for 100 test messages.

2. Measures of Effectiveness for Data Extraction

Given the official answer key, it was necessary to come up with measures of how well each system's output agreed with the key. The MUCK-II evaluation used a simple, ad hoc scoring scheme for this purpose. For MUC-3, the organizers and participants felt a more rigorous approach was needed. Effectiveness measures have long been the subject of research in information retrieval (IR), and the decision was made to adapt and extend IR measures of effectiveness for MUC-3. In this section we first describe the formal model underlying the most widely used IR measures of effectiveness. We then discuss the changes made in order to adapt these measures to the data extraction task. Finally, we describe the procedures and software used for computing these measures, given system output and the official answer key.

[5] Recall, precision, and MATCHED/MISSING are defined in Section 2.3; briefly, recall measures completeness, precision measures accuracy, and MATCHED/MISSING is a manner of scoring that penalizes for missing information but not spurious information.

	Yes is Correct	No is Correct	
Decides Yes	a	b	a+b
Decides No	c	d	c+d
	a+c	b+d	a+b+c+d=n

Figure 6
Contingency table for a set of binary decisions.

2.1 Effectiveness Measures in Information Retrieval

A primary focus of IR research is text classification. If we consider a single category, then a text classification system must make *n* binary decisions to categorize *n* units of text. The result of *n* such decisions can be summarized in a *contingency table*, as shown in Figure 6. Each entry in the table specifies the number of decisions with the indicated result. For instance, *a* is the number of times the system decided *Yes* (assigned the text to the category), and *Yes* was in fact the correct answer.

Given the contingency table, three important measures of the system's effectiveness are:

1. recall = a/(a+c)
2. precision = a/(a+b)
3. fallout = b/(b+d)

Recall (true positive rate) and fallout (false positive rate) originated in signal detection theory (Swets 1964). They measure the ability of a system to detect a signal in the presence of noise, and the system's relative willingness to make errors of commission versus errors of omission. Recall is widely used in IR, but fallout is often replaced by precision, a measure that is more intuitive though sometimes less informative (Swets 1969). For MUC-3, a fourth measure, overgeneration, was defined. It is discussed in Section 2.3 and does not have a direct interpretation in terms of the contingency table.

A system can achieve perfect recall by never deciding *No;* or perfect precision, fallout, and overgeneration by never deciding *Yes*. Therefore, at least recall, plus one of the other three measures, is necessary for a nontrivial evaluation of a system's effectiveness under the contingency table model. Note that all of these measures assume a uniform cost for errors across texts and across type of error (cell *b* versus cell *c*). Treating some fillers as being more important than others may be desirable in evaluating data extraction systems for particular applications.

2.2 Adapting the Contingency Table Model to MUC-3

The contingency table measures seemed a natural approach for measuring the ability of data extraction systems to minimize uncertainty about correct fillers, and for measuring the systems' trade-off between generating incorrect fillers and missing correct fillers. However, the MUC-3 task differs from text classification and other signal detection tasks in many ways, and these had to be taken into account in producing effectiveness measures for MUC-3.

Consider a simplified version of the MUC-3 task: a template consists of m slots, with slot i having n_i possible fillers. Slot i can be filled with between 0 and n_i of those fillers, and no filler can be repeated. The contingency model applies straightforwardly to this simplified case, since a filled-out template can be viewed as the result of $n_1 + n_2 + \cdots + n_m$ decisions to assign or not assign each of the legal slot fillers.

The actual MUC-3 task departs in several ways from this simplified model. Most set fill slots do allow the same fill to occur several times. String fills, dates, and locations are difficult to view as the result of binary decisions. The optional and alternative versions of correct answers in the MUC-3 answer keys call into question a dichotomy between correct and incorrect decisions, as does the allowing of partial credit for answers. Our (imperfect) approach to all these difficulties was to model systems as making a binary decision between generating a correct filler and generating an incorrect filler. Partial matches were scored as one-half a correct decision. This obviously is a considerable simplification of the task that actually faced systems.

The number, as well as the character, of decisions in MUC-3 also departed from this simple model. Several slots allowed an unbounded number of fillers, and thus a potentially unbounded number of decisions, whether or not decisions were viewed as binary. Our solution to this was to assume that the number of *Yes* decisions made by the system was the number of fillers generated by the system, while the number of decisions for which *Yes* was correct was the number of fillers in the key, with optional fillers being counted only if attempted by a system. This gave us values for contingency table cells *a*, *b*, and *c*, but not *d*. Recall and precision, but not fallout, could be computed for slots with an unbounded number of fillers and/or fillers from an unbounded set of alternatives. A slightly different method was used for set fill slots, and this method did allow a variant on fallout to be computed.

Other issues arose because the decisions made in template filling are not independent of each other. Complex rules for assigning partial credit in cases of cross-referenced slots had to be defined. In addition, real-world relationships made some fillers completely dependent on others. For instance, the filler for EFFECT ON HUMAN TARGET would always have been DEATH if the INCIDENT TYPE was MURDER. To avoid treating such cases as two decisions, we forbade slots to be filled in certain cases. This decision was made for evaluation purposes. In viewing the templates as database entries, however, it may be desirable to make redundant information explicit.

At the template level, a system might generate more or fewer templates than were present in the answer key. The appropriate treatment of unmatched templates was not at all clear, so the scoring program was designed to generate overall scores under three different assumptions (MATCHED ONLY, MATCHED/MISSING, and ALL TEMPLATES), which are described in the next section.

2.3 Scoring Procedure

Each template fill generated by a MUC-3 system was compared with the corresponding fills in the answer key. The fill was counted as falling into one of six categories, as shown in Figure 7. The counts COR (correct), SPU (spurious), MIS (missing), and NON (noncommittal) are roughly analogous to the values *a*, *b*, *c*, and *d*, respectively, in the contingency table of Figure 6. PAR (partial) indicates the number of decisions that are counted as adding only 0.5 to cell *a*. A decision in INC (incorrect) would correspond to two incorrect decisions in the contingency table model, one counted in cell *b* (for

Category	Criterion	Column
Correct	response = key	COR
Partial	response ≅ key	PAR
Incorrect	response ≠ key	INC
Spurious	key is blank and response is not	SPU
Missing	response is blank and key is not	MIS
Noncommittal	key and response are both blank	NON

Figure 7
Scoring criteria.

entering a spurious answer) and one counted in cell *c* (for failing to get the correct answer).

The comparison between system response and answer key was done using a semi-automated scoring program, which produced a summary score report like that shown in Figure 8. The summary score report showed the total over all templates for each of the six response categories. The user of the scoring program could override its decision and interactively score an answer as partially or completely correct even when it did not match the key. The ICR (interactive correct) and IPA (interactive partially correct) columns indicated the number of fills scored interactively as correct and partial, respectively. The POS column contained the number possible, which was the sum of the number correct, partial, incorrect, and missing. The number possible was computed from the system responses in comparison with the answer key rather than from the answer key alone because of the optional templates and slot fillers appearing in the key. The ACT column contained the number of actual fills, which was the sum of the number correct, partial, incorrect, and spurious.

The evaluation metrics of recall, precision, overgeneration, and fallout (shown in the last four columns of Figure 8) were calculated using the totals in these columns both for the individual slots and for all of the slots combined[6] as follows:

- Recall was the degree of completeness of attempted fills.

  $\text{Recall} = (\text{COR} + (0.5 * \text{PAR}))/\text{POS}$

- Precision was the degree of the accuracy of attempted fills.

  $\text{Precision} = (\text{COR} + (0.5 * \text{PAR}))/\text{ACT}$

- Overgeneration was the degree of spurious generation.

  $\text{Overgeneration} = \text{SPU}/\text{ACT}$

- Fallout was the degree of producing incorrect fills relative to the number of possible incorrect fills for set-fill slots.[7]

  $\text{Fallout} = (\text{INC} + \text{SPU})/(\text{NUMBER\_POSSIBLE\_INCORRECT})$

[6] A more detailed mathematical description of the evaluation metrics complete with examples can be found in Chinchor (1991a).

[7] The number of possible incorrect fills is the cardinality of the set of allowable fills minus the number of fills in the key.

SLOT	POS	ACT	COR	PAR	INC	ICR	IPA	SPU	MIS	NON	REC	PRE	OVG	FAL
template-id	117	115	114	0	0	0	0	1	3	39	97	99	1	
incident-date	113	110	90	10	10	31	10	0	3	4	84	86	0	
incident-type	117	114	112	1	1	0	1	0	3	0	96	99	0	0
category	90	109	88	0	0	0	0	21	2	6	98	81	19	15
indiv-perps	104	61	59	0	2	10	0	0	43	49	57	97	0	
org-perps	69	68	58	0	1	15	0	9	10	47	84	85	13	
perp-confidence	69	68	56	1	2	12	1	9	10	47	82	83	13	2
phys-target-ids	59	57	54	3	0	14	3	0	2	76	94	97	0	
phys-target-num	41	41	39	0	2	0	0	0	0	76	95	95	0	
phys-target-types	59	57	52	4	1	11	4	0	2	76	92	95	0	0
human-target-ids	144	133	129	2	0	33	2	2	13	23	90	98	2	
human-target-num	93	88	79	6	2	0	6	1	6	23	88	93	1	
human-target-types	144	133	126	2	3	23	2	2	13	23	88	95	2	0
target-nationality	19	19	14	2	0	4	2	3	3	103	79	79	16	0
instrument-types	24	22	16	1	0	0	0	5	7	88	69	75	23	0
incident-location	117	113	88	24	1	0	1	0	4	0	85	88	0	
phys-effects	41	44	37	3	0	8	3	4	1	88	94	88	9	0
human-effects	56	55	43	2	2	10	2	8	9	80	78	80	14	1
MATCHED ONLY	1442	1407	1254	61	27	171	37	65	100	827	89	91	5	
MATCHED/MISSING	1476	1407	1254	61	27	171	37	65	134	848	87	91	5	
ALL TEMPLATES	1476	1425	1254	61	27	171	37	83	134	852	87	90	6	
SET FILLS ONLY	619	621	544	16	9	68	15	52	50	511	89	89	8	0

Figure 8
Sample summary score report showing one evaluator's scores for the answer key.

As discussed in the previous section, these measures are considerably modified from those of the same name in information retrieval, and should not necessarily be interpreted in the same way.

At the bottom of the summary score report were four different summaries of system performance for the full complement of slots. The first three of the summary score rows corresponded to giving different importance to over- or under-populating the database. For a strict score, slot fills in a spurious template could all be scored as spurious or, for a lenient score, just the template could be scored as spurious. Similarly, slot fills in a missing template could be scored as individually missing or just the template could be scored as missing. The first summary score row was the MATCHED ONLY row, which indicated the scores resulting from scoring missing and spurious templates only in the *template-id* slot. The MATCHED/MISSING row contained the official MUC-3 test results. The missing template slots were scored as missing, whereas the spurious templates were scored only in the *template-id* slot. The totals in this row were the totals of the tallies in the columns as shown. The metrics were calculated based on the summary totals. The ALL TEMPLATES row had missing templates scored as missing for all missing slot fills and spurious templates scored as spurious for all spurious slot fills.[8] This row was the strictest score for the systems. Another way of describing the differences between these measures is that ALL TEMPLATES roughly corresponded to taking a message-level view of performance, MATCHED ONLY a template-level view, and MATCHED/MISSING something in between.

The fourth summary score row gave the scores for SET FILLS ONLY. The totals here were for slots with finite set fills only. A global fallout score as well as recall, precision, and overgeneration could be calculated for these slots and was given in this row. Note that the denominator of the fallout score is determined by the number of possible incorrect fills for each slot instance and, thus, depends on the cardinality of the set of possible fills and how many of those fills appear in the answer key.

2.4 Scoring Software

The scoring for MUC-3 was carried out using an interactive scoring program especially developed for the evaluation. The program simultaneously displayed the system response and corresponding answer key template in different windows. It automatically scored the displayed templates, requiring user interaction only for instances of mismatching slot fills. The user would determine whether the mismatch should be scored as a full match, a partial match, or a mismatch, according to the official scoring guidelines. The scoring program kept a history of user interactions. It also produced a detailed score report of the template-by-template scores and a summary score report. In addition to being used for the official scoring, the program was used by the participating sites during development as a tool for determining progress and for regression testing. Additional features allowed the program to be adapted whenever the template design changed, allowed the scoring of subsets of slots and templates useful in linguistic phenomena testing (Chinchor 1991b), and allowed the merging of partial credit decisions across sites for more uniform scoring. During interactive scoring, the program kept track of what partial matches were allowed. By pooling these records across sites, the systems could be rescored, giving all systems the benefit of a partial match allowed for one site. Each site scored its answers individually for presentation at the conference, but the official scores were produced by volunteers from two sites working together.

3. Participant Methodologies

3.1 Technical Requirements

The nature of the task in MUC-3 imposed a number of requirements on the participating systems. Despite considerable divergence in technical approach, all the systems had to cope with issues such as large vocabulary, English and Spanish proper names, generation of well-formed templates, and discrimination of relevant from irrelevant messages. In Figure 9 we show a generic architecture for the MUC-3 systems.[9] It consists of a *preprocessing* module, a *lexical processing* module, a *linguistic* component (for

[8] The ALL TEMPLATES measure was adopted as the official measure for MUC-4.

[9] This is intended to be a generic architecture. Each system put together its own subset of these capabilities governed by a specific control strategy (not represented at all in the figure).

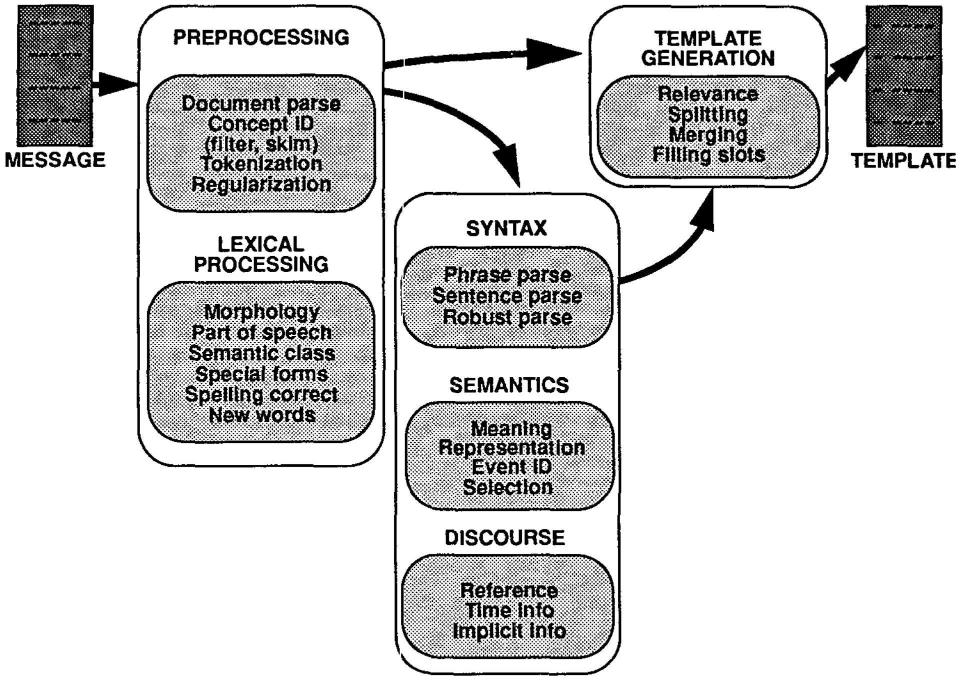


Figure 9
A generic message understanding system architecture.

the linguistically based systems), and a *template generation* module. It is interesting to note that two of the modules (*preprocessing* and *template generation*) are not really part of standard computational linguistics. This reflects the demands of a "realistic" application, where a substantial amount of effort is devoted to system engineering and data transformation issues.

3.1.1 Preprocessing. Almost all systems included a module that was referred to as a "preprocessor." The function of the module varied from system to system, but often included:

- Parsing of the message into fixed field information and free text.
- Identification of *interesting* segments of text (where the segment could range from the word level all the way up to the message level). Text segments or even entire messages that were categorized as uninteresting were often filtered out from further processing.
- Segmentation of the character stream into tokens.
- Regularization of the message text, including regularization of certain kinds of forms (dates, numbers), bracketing into phrases, identification of certain discourse transition markers, etc. These operations often overlapped with those in lexical processing.

3.1.2 Lexical Processing. The lexical processing stage was responsible for providing information on words in the text stream. This could include:

- Morphological processing: Decomposition of words into parts, using explicit morphological rules or string-matching methods.
- Assignment of syntactic classes: Methods used included lexicon lookup (of exact form or root), inference from affixes (detected by morphological analysis or string matching) and statistical tagging. Some systems used a small set of atomic classes, others specified more elaborate complement structures or sets of features.
- Semantic analysis: Tagging words as to semantic types, instantiating knowledge base concepts, or activating semantic predictions. This varied widely between systems.
- Special form processing: There were a number of productive expression types (e.g., times, dates) that needed special processing. In addition, many systems had modules for handling names (especially Spanish names).
- Spelling correction.
- New words: Many systems had a mechanism for categorizing unknown words; it often interacted heavily with morphological analysis, syntactic class assignment, and spelling correction.

3.1.3 Linguistic Processing. Both the syntax-driven and the semantics-driven systems provided a stage of linguistic processing. This processing could include:

- Syntactic processing to identify basic phrases and/or sentences; this often included robust or partial parsing.
- Semantic processing, which could include application of selectional restraints, reduction of information to a canonical semantic representation, most often in the form of a logical expression or an instantiated frame, and identification of individual events.
- Discourse processing, which handled resolution of referring expressions (including pronouns and definite reference), temporal analysis, and reconstruction of implicit information, as well as determination of other discourse relations.

3.1.4 Template Generation. The final stage, template generation, was present in all of the systems, since templates were the output to be evaluated in MUC-3. However, there was often no one-to-one correspondence between event representations and templates, leading systems to develop various strategies for deciding how to map events found in a single message into zero or more templates. Thus issues for template generation included:

- Whether to generate a template at all, since there were complex rules concerning what constituted a terrorist incident;
- Whether to generate multiple templates for a message because a single message, or even a single sentence, could report more than one incident;

- Whether to merge multiple event representations into a single template; and
- How to fill in the slots of the templates.

3.2 System Summaries

The following subsections provide brief descriptions of the participating systems; all information is taken from the MUC-3 proceedings, unless otherwise noted. The reader is referred to these proceedings for detailed system descriptions and references and to Figures 2 and 3 for system results.

3.2.1 Pattern-Matching Systems.

ADS

The ADS system combined a pattern-matching text categorization system and a natural language system. The text categorization system was designed to serve as a loose filter in the preprocessing stage, and as a possible back-up in case of failure to parse. Because of difficulties in scaling the lexicon to the full MUC-3 training set, the natural language system was not run, and only the results from the text categorization system are reported for MUC-3. Heuristically weighted rules for this system were handcrafted for each of the significant event types. A template with only the *incident-type* slot filled was created for each sentence containing incident-type information.

Hughes

The Hughes system made use of machine learning methods to correlate input text with filled templates. Processing of a new message began by using a semantic grammar to decompose each sentence into phrases. Phrasal rules and content words were associated with nodes in a concept hierarchy. A sentence was represented as a list of binary features corresponding to concept nodes. This list of feature values was used to retrieve the k most similar cases from a case memory ($k = 12$ for MUC-3). Each case was a pairing of a previously analyzed sentence (i.e., a list of feature values) from the training corpus with its corresponding template. A tentative set of slot fills, including string fills, for each sentence was produced by combining fillers from retrieved cases. Competitive statistical filtering was used to form groups of adjacent sentences and assign them definite incident types. The tentative fills for the sentences in each group were combined to form a single template, though dates and locations were handled by special purpose code.

MDESC

The MDESC system was planned as a multicomponent system providing tools for developing customized text understanding systems, but only the first module, the skimmer, was used in MUC-3. The skimmer identified fixed (multiword) phrases, including proper names, replacing them with the corresponding concepts. Keywords were also used to tag segments of text. These segments were then grouped into actors and objects and used to fill templates. The templates were checked by a semantic trimming and rejection process.

Unisys

The Unisys system was also envisioned with a pattern-matching component as a preprocessor, followed by a linguistic component for fuller analysis of interesting segments. However, only the pattern-matching component was used in MUC-3. The pattern-matching component combined a statistical keyword-based retrieval of relevant text portions plus a knowledge-based component, which supported constraints stated in terms of various string relations, such as contiguity, same text, same paragraph, same sentence. These rules were used to infer facts, and the template-generation system then operated on these facts to produce filled templates.

UNL/USL

The UNL/USL system was primarily oriented toward filling set-valued slots. Subsets of the training corpus were used in training linear classifiers by the perceptron algorithm. A separate classifier was used to find an activation level for each set fill, based on the presence of words and multiword phrases in the text. An additional classifier was trained to find an activation level for overall relevance, that is, whether a message should produce any templates. Templates were generated for a message if the message scored high enough on overall relevance, or if handcrafted rules indicated that sufficient individual fillers had been highly activated. If the decision was made to generate templates, exactly one template was created for each incident type with a sufficient activation level. Then each sufficiently activated filler was assigned to the single incident type with the most similar pattern of activations across paragraphs. In addition, the string fill slots *perpetrator-id* and *incident-location* slots were filled if a string was found in the text that exactly matched a known filler of those slots on the training corpus. As with set fills, each string fill was assigned to a single template by comparison of paragraph level activation vectors.

3.2.2 Syntax-Driven Systems.

BBN

The BBN system used automated part-of-speech tagging and a robust parser to obtain partial syntactic parses. A set of partial parses spanning the input string was then "glued together" for semantic (case frame) analysis. Discourse analysis created event structures based on the semantic case frames and also performed limited reference resolution. To partially automate the knowledge acquisition process, the domain-dependent case frames for semantics were acquired using a tool to hypothesize case frames based on predicate occurrences in training data from the tagged TREEBANK corpus (Santorini 1990) validated by a developer.

ITP

A domain-independent naive semantics lexicon was at the heart of the ITP system. The syntactic analysis was done using a borrowed parser because a broad-coverage government-binding parser was still under development. The parser produced a single parse (indicating ambiguous attachments), which the system post-processed to resolve attachment ambiguities and to perform word sense disambiguation based on the naive semantics lexicon. Discourse analysis identified actors and events and included a treatment of modals, negation, coordination, temporal processing, locative processing, and reference resolution.

LSI

The LSI system eliminated uninformative messages by using Boolean combinations of keywords and other heuristics. For linguistic processing, the system used a government binding–based parser that was still under development. An "Unexpected Inputs" component shadowed the message-processing components and provided modules for handling unexpected words, parsing failures, and template generation problems. The system used a single handcrafted lexicon for syntactic and semantic knowledge with a tool that allowed system developers to link semantic information to the frame database (knowledge base) and lexicon.

NYU

The NYU system used a lexicon derived from the Oxford Advanced Learner's Dictionary, supplemented with hand-built entries for selected words, as well as for organizations, locations, and proper names found in training messages. The domain-specific semantic hierarchy and lexicosemantic models were developed using keyword-in-context indices derived from the training data. The system filtered out sentences if they did not contain any words corresponding to an interesting semantic concept. The broad coverage linguistic string grammar and underlying chart parsing mechanism were augmented with parse heuristics that enabled the system to process well over 90% of the input deemed relevant. Semantics was expressed in terms of frame-like entity and event structures and was followed by reference resolution. The template generation module suppressed generation of templates according to a set of heuristic rules. The system was run in several configurations, changing the heuristics that controlled when to generate templates.

PRC

The PRC system used its existing core modules for the lexicon, morphology, syntax, and semantics. These were extended and adapted to the MUC-3 domain using various interactive and batch tools. The system used a semantic concept analyzer for rejection of hypothesized parses. The parser also used a time-out mechanism: for sentences taking more than a threshold time (approximately 50% of the corpus), the parser simply returned the longest parsed substring or a run-on sentence analysis. Features new for MUC-3 were the addition of heuristics for guessing unknown words and a discourse module that collected conceptual frames for use in template generation.

SRI

For linguistic processing, the SRI system used its broad-coverage grammar with a handcrafted 12,000-word lexicon. A relevance filter based on *n*-gram statistics and hand-specified keywords was used to screen out sentences unlikely to contain useful information. Implementations of various parse heuristics were key to robustness in parsing the remaining sentences. The syntactic processing stage produced a parse and a logical form, which was passed to the pragmatics component. Pragmatics was done by abductive theorem proving, which attempted to build the lowest cost explanation for the observed set of semantic relations. The process of building this explanation involved resolving reference (by minimizing the number of entities), filling in contextual information, and inferring relationships among events, all driven by a domain-specific collection of axioms. Templates were generated from the output of the pragmatics component using heuristics about the number and kind of templates to generate from an "interesting act."

3.2.3 Semantics-Driven Systems.

GE

The GE system tool set was used to adapt and customize existing modules for the MUC-3 application. Preprocessing was quite extensive in the MUC-3 domain and included initial segmentation and bracketing into discrete units, as well as activation of relevant templates based on the presence of keywords. At the heart of the system was a domain-independent 10,000 word-root lexicon and a 1,000-concept semantic hierarchy. Processing used "relation-driven" control, which combined syntactic and semantic preference scores. This permitted semantic preferences to influence parse priorities, avoiding a combinatorial explosion of possible parses. The grammar, like the lexicon and the concept hierarchy, required only minor adaptation to the MUC-3 task. Discourse processing used text segmentation based on definite and indefinite reference and various cue phrases to identify and segment events. Subsequent processing split and merged events, extracted temporal relations, and filled in missing arguments to generate templates.

GTE

The GTE system used a phrase parser to identify phrases. These phrases served as input to a semantic case frame analyzer that combined expectation-driven analysis with a bottom-up analysis of the information contained in the phrases. There was a special module in semantics for conjunction processing. Reference resolution was handled by merging "similar" concepts at the frame level and by combining events into a single template.

Synch/UMD

The Synch/UMD system was developed by researchers across several sites using a pipelined architecture to facilitate multisite development. The first stage was a phrase parser, which segmented the text into phrases. This was followed by a semantic phrase interpreter, which mapped the phrases into a semantic net representation via a spreading activation process, to make the system robust in the face of missing knowledge. The syntactic component then performed some limited kinds of syntactic regularization, as needed. The template generator worked from the semantic net representation, determining whether and how to generate templates given the semantic concepts.

U Mass

The U Mass system used a semantic case frame approach to drive the processing. There was some initial preprocessing of phrases, dates, and names, followed by lexical look-up, which associated words with concepts. A text that did not trigger any concept nodes was classified as irrelevant. Once a concept node was activated by association with a word, it triggered certain semantic predictions for slot fillers; these might also require some limited syntactic phrase parsing. This stage generated a set of instantiated case frames, which went through a process of "consolidation" to merge frames into template-level events, using a set of domain-specific rules. An optional case-based reasoning component associated a set of concept nodes with the slot fills derived from those nodes. These were grouped into classes by incident type. For a new message, the associated concepts were compared with the case base, which was generated automatically from the training corpus, to find matching configurations of concepts. These were used to hypothesize possible templates and slot fillers.

4. Significance Tests on Measures of Effectiveness

Recall and *precision* measure the absolute ability of systems to perform the data extraction task. Measuring the relative ability of different systems requires not only that we can measure effectiveness but also that we can distinguish meaningful differences in effectiveness from inconsequential ones. Defining a meaningful difference may depend on the particular data extraction application. For instance, suppose two systems behave similarly in producing slot fills corresponding to proper names in the original text, except that system A always includes a few extraneous words along with a proper name, and system B does not. System A would get a substantially lower score according to MUC-3 criteria. However, the two systems might be equally effective operationally if the extracted names were used to index messages for text retrieval, and users always searched on, say, the last names of individuals. On the other hand, system A would in fact be inferior if extracted proper names had to be matched identically against other names in order to join two records.

Differences in effectiveness are also inconsequential when there is a significant probability that the difference resulted from chance. Determining whether differences are uninteresting in this way is the province of statistical hypothesis testing. In this section we review the main concepts in statistical significance testing and describe our approach to significance testing for the MUC-3 data.

4.1 Review of Statistical Significance Testing

A statistical significance test measures the extent to which data disagree with a particular hypothesis.[10] The test begins with the posing of the *null hypothesis,* that is, the hypothesis that a relationship of interest is *not* present. In order to pose a null hypothesis we must determine a relationship of interest that can be tested and state that it is not present. One thing of interest about two systems in MUC-3 is whether one system is better than the other on the data extraction task. To know this would require knowing whether one system is better than the other in both recall and precision.

Recall and precision are examples of *test statistics.* A test statistic is a function that can be applied to a set of sample data to produce a single numerical value. A set of sample data consists of *observations*—instances of values of a set of random variables. In MUC-3, examples of random variables are the number correct, partially correct, possible, and actual. The test statistic of recall is a function of the number correct, partially correct, and possible. The test statistic of precision is a function of the number correct, partially correct, and actual.

A null hypothesis is stated in terms of a single statistic. The relationship of interest is whether one system is better than another. However, in order to state the null hypothesis in terms of a single test statistic, we must consider recall and precision separately. Also, for testing purposes we can simplify the relationship to just look at whether one system differs from another since we know the direction of the difference from the scores themselves.

The form of the null hypotheses we will test for MUC-3 is:

> *System X and system Y do not differ in recall.*

Corresponding null hypotheses will be tested for precision as well. Notice that the single test statistic for the null hypothesis is the difference in recall between the two systems. The null hypotheses are more formally stated as:

> *The absolute value of the difference between system X's overall recall (precision) score for the data extraction task and system Y's overall recall (precision) score for the data extraction task is approximately equal to zero.*

To test a null hypothesis, the test statistic is applied to the observations of the random variables and the value of the test statistic is used to determine a *p-value,* or *significance level*—the probability that a test statistic as extreme or more extreme than the actual value could have arisen by chance, given the null hypothesis. If the value

[10] In this section we draw on descriptions of hypothesis testing appearing in Noreen (1989) and Chatfield (1988).

of the test statistic is unlikely to have arisen by chance, then we have evidence against the null hypothesis.

4.2 A Computer-Intensive Approach to Significance Testing

In significance testing, we need to compute the probability that a test statistic value as extreme as the test statistic for the actual data could have arisen randomly, given the null hypothesis. If observations are drawn from a known probability distribution and the test statistic is of the proper form, then this probability can be found analytically. However, the significance tests that arise from this conventional, analytic approach are inapplicable to the MUC-3 data, given our lack of knowledge of the distribution of the random variables.

The availability of cheap computing power has resulted in an increased interest in computer-intensive methods in statistics (Efron and Tibshirani 1991). These methods avoid some of the more restrictive assumptions of conventional statistical methods by explicitly simulating large numbers of random trials, thus eliminating the need to know the distribution of the random variables. Our analysis of the MUC-3 results uses one of these methods, a significance-testing technique known as approximate randomization (Noreen 1989).

An *exact randomization* test simulates all logically possible sets of observations, applying the test statistic formula to produce a pseudostatistic for each observation. In other words, an exact randomization test would simulate all the logically possible results and calculate the significance level of the actual results by comparison with the simulated results. For a problem like ours, there are too many logically possible results for it to be practical to run an exact randomization test. *Approximate randomization* tests are similar to exact randomization tests, but use a nonexhaustive random sample of the possible sets of observations. An approximate randomization test for the MUC-3 results can be run in less than a day. It is also possible to provide the *confidence level* of the test that measures how indicative the approximate randomization is of the exact randomization.

Both exact and approximate randomization require the production of a large number, *ns* (number of shuffles, as defined in Section 4.3), of *pseudostatistics* (random values of the test statistic). These random values form an empirically generated distribution curve that can be used to calculate the significance level of the actual test statistic. Each random value is compared with the actual value of the test statistic on the original data. The number of times, *nge*, that the pseudostatistic is greater than or equal to the true statistic is recorded. The significance level (p-value) of the test is the ratio of $(nge+1)/(ns+1)$. In general, the lower the p-value, the less probable it is that the null hypothesis holds; that is, in our case, the lower the p-value, the more likely it is that the two systems are significantly different. The confidence level for the significance level can be calculated or looked up in published tables (Noreen 1989) if approximate randomization is used. The higher the confidence level, the more probable it is that the approximate randomization test gave the significance level that an exact randomization test would have given.

4.3 Application of Approximate Randomization to the MUC-3 Results

We use the approximate randomization technique set forth by Noreen (1989) with stratified shuffling to control for categorical variables that are not of primary interest in the hypothesis test. For more details on this method see Chinchor (1992).

The test statistic we have chosen is the absolute value of the difference in recall or precision. The data consist of the four-tuples of number possible, actual, correct, and partially correct for each message for each system. The actual test statistic is calculated for each pair of systems. The desired number of shuffles is set to 9,999 because it was determined that 9,999 shuffles produced slightly higher confidence levels than 999 and were worth the 16-fold increase in computing time.

In the algorithm, once the desired number of shuffles is set, the counters for the number of shuffles, *ns*, and the number of times the pseudostatistic is greater than or equal to the actual statistic, *nge*, are set to 0. A loop then increments *ns* until it has exceeded the desired number of shuffles. The first step in this loop is to shuffle the data. Data is shuffled by exchange of the systems' message scores depending on the outcome of a computer-simulated coin flip. After 100 coin flips, one per message in the MUC-3 test, the absolute value of the difference in the test statistics of the resulting pseudosystems can be compared with the corresponding absolute value of the difference in the test statistics of the actual systems. The value of *nge* is incremented every time a randomized pair of pseudosystems satisfies the inequality:

$$|\mathrm{stat}_{\mathrm{PseudoA}} - \mathrm{stat}_{\mathrm{PseudoB}}| \geq |\mathrm{stat}_{\mathrm{A}} - \mathrm{stat}_{\mathrm{B}}|$$

where *stat* is the test statistic. The significance level is estimated by $(nge + 1)/(ns + 1)$, where the ones are added to ensure that the test is valid. The corresponding confidence level is then found by table lookup.

According to Noreen,

> Randomization is used to test the generic null hypothesis that one variable (or group of variables) is unrelated to another variable (or group of variables). Significance is assessed by shuffling one variable (or set of variables) relative to another variable (or set of variables). Shuffling ensures that there is in fact no relationship between the variables. If the variables are related, then the value of the test statistic for the original unshuffled data should be unusual relative to the values of the test statistic that are obtained after shuffling. (Noreen 1989, p. 9)

In our case, the four-tuple of data associated with each message for each system is the dependent set of variables that is shuffled. The explanatory variable is the system. Shuffling ensures that there is no relationship between the differences in the scores and the systems that produced them, that is, that the differences were achieved by chance. If they were not achieved by chance, then the value of the actual test statistic will be far enough out on the tail of the empirically generated distribution to be significant (as indicated by the significance level).

Informally speaking, shuffling two systems with a large recall score difference is likely to produce more homogeneous pseudosystems whose difference in recall will be less than or equal to the observed difference most of the time. This will lead to a small *nge* value and a low p-value. Conversely, shuffling two systems whose behavior is quite similar may lead to a larger number of times that the recall score difference between the pseudosystems is greater than or equal to that of the real systems; this creates a larger p-value and a lower likelihood that the differences are significant.

4.4 Choice of Observations

In MUC-3 there are several ways of breaking down the output of systems into observations on random variables. The number of correct and partially correct answers generated by a system are recorded at the slot level and then aggregated at the template and message levels before recall is computed. Recall could be viewed as a summary measure computed from the 2,771 or more (system-dependent) observations of system behavior on slots in key templates, from the 163 or more (system-dependent) observations of system behavior on key templates, or from exactly 100 (system-independent) observations of system behavior on messages. The system-dependent counts are a result of optional slots and templates in the key that are counted only when a system has chosen to fill those slots or templates. The number of messages is independent of the system's fills.

Whether data extraction systems succeed or fail, and thus should be observed at the message level, template level, slot level, or at some completely orthogonal level of granularity is an open question. From the standpoint of significance testing, however, the most reasonable approach available to us was to consider recall and precision as test statistics computed on observations at the message level, since this assumes the smallest and only exact number of observations. The two corresponding message four-tuples of possible, actual, correct, and partial scores for the two systems were randomly exchanged. The pseudostatistic was arrived at by summing the possible, actual, correct, and partial for the two pseudosystems constructed by the random exchange and computing based on those sums.

Two hypothetical examples illustrate the method applied at the message level. Suppose we have two systems reporting precision results for a set of 100 messages. System A has a score of 15/20 on each of 50 relevant messages and no spurious templates, for a total of 750/1000 = 75% precision. System B has the identical score on each of the relevant messages except one, for which it has a score of 0/20. System B has a precision of 735/1000 = 73.5%. In the random shuffle, either system A or system B will have the "0 precision" template. The pseudostatistic will always be equal to the measured absolute difference between system A and system B, that is, 1.5%. Since a difference in precision of at least 1.5% has a probability of 1.0 of arising when the systems are randomly shuffled, an observed difference of 1.5% has a p-value of 1.0 and, therefore, is not statistically significant.

Now suppose that we have a third system, system C, which always gets 18/20 on the same set of 50 relevant messages with no spurious templates. Any random shuffle of systems A and C is likely to produce a smaller difference than the absolute value of the difference between A and C. The p-value will be extremely close to zero indicating with virtual certainty that the two systems are significantly different.

These examples correspond with our intuition that two systems that show consistent differences across a range of messages will more likely be statistically significantly different in their scores than two systems that differ on just a few messages.

5. Significance Results

This section presents significance results on the differences between recall and precision for each of the 105 pairs of MUC-3 systems. We draw some tentative conclusions about the behavior of MUC-3 systems with respect to the recall and precision measures and provide some cautions with respect to interpreting our results.

5.1 Results of the Approximate Randomization Test

The results of the approximate randomization test for 9,999 shuffles are presented in Figure 10. Recall and precision scores, as computed for each system using the MATCHED/MISSING method (Section 2.3), are listed along the top and left side (recall first, then precision). For purposes of the statistical testing, more exact values of recall and precision than those listed were calculated from the raw data. The top of each cell in the table presents the significance level (p-value) for the difference in recall between

	ADS 12, 56	BBN 45, 52	GE 42, 63	GTE 11, 43	HUGHES 31, 36	ITP 20, 58	LSI 16, 48	MDC 28, 51	NYU 44, 57	PRC 28, 53	SRI 25, 65	SYNCH 7, 55	UMASS 51, 62	UNEB 14, 34	UNISYS 19, 54
ADS 12, 56		0.1894		0.4634 *0.0012	*0.0001		0.0188	0.1961		0.2762		*0.0001 0.9188		*0.0001	0.7478
BBN 45, 52	*0.0001		0.3522	*0.0001 *0.0002	*0.0001 *0.0001	*0.0001	*0.0001 0.0769	*0.0001 0.4929	0.8311	*0.0001	*0.0001	*0.0001		*0.0001 *0.0001	*0.0001
GE 42, 63	*0.0001 0.0484	*0.0001		*0.0001 *0.0001	0.0078 *0.0001	*0.0001 0.2028	*0.0001 *0.0004	*0.0005 *0.0009	0.0677	*0.0002 *0.0016	*0.0001	*0.0001 0.0616	0.8502	*0.0001 *0.0001	*0.0001 0.0272
GTE 11, 43					*0.004							0.0523		0.0161	
HUGHES 31, 36	*0.0001			*0.0001		*0.0002	*0.0001	0.282		0.2102	0.0653	*0.0001		*0.0001 0.6272	*0.0001
ITP 20, 58	*0.0006 0.4912	0.0108		*0.004 *0.0001	*0.0001		0.1444 *0.0004	0.0467	0.7181	0.0399		*0.0001 0.4781		0.0124 *0.0001	0.6109 0.2457
LSI 16, 48	0.0473			0.046 0.0567	*0.0001							*0.0001		0.2427 *0.0001	
MDC 28, 51	*0.0001			*0.0001 0.0267	*0.0001	*0.0041	*0.0002 0.3917				0.2448	*0.0001		*0.0001 *0.0001	*0.0006
NYU 44, 57	*0.0001 0.6907	0.0208	0.5788	*0.0001 *0.0001	*0.0002 *0.0001	*0.0001	*0.0001 *0.0039	*0.0001 0.0362		*0.0001 0.0691	*0.0001	*0.0001 0.655		*0.0001 *0.0001	*0.0001 0.4185
PRC 28, 53	*0.0001	0.9502		*0.0001 *0.0002	*0.0001	*0.001	*0.0001 0.0627	0.9761 0.5504			0.2226	*0.0001		*0.0001 *0.0001	*0.0005
SRI 25, 65	*0.0001 *0.0052	*0.0001	0.517	*0.0001 *0.0001	*0.0001	0.1251 0.0752	0.014 *0.0001	*0.0002	0.0093	*0.0001		*0.0001 0.0204	0.3776	*0.0001 *0.0001	0.0766 0.0079
SYNCH 7, 55		0.3228		*0.002	*0.0001		0.028	0.3294		0.3821				*0.0001	0.8515
UMASS 51, 62	*0.0001 0.0348	0.01 *0.0001	0.0093	*0.0001 *0.0001	*0.0001 *0.0001	*0.0001 0.2426	*0.0001 *0.0001	*0.0001 *0.0003	0.0143 0.0276	*0.0001 *0.0004	*0.0001	*0.0001 0.0672		*0.0001 *0.0001	*0.0001 0.0143
UNEB 14, 34	0.1397			0.1281								*0.0002			
UNISYS 19, 54	*0.0024	0.4656		0.0099 *0.0017	*0.0001		0.4781 0.0429	0.3641		0.5661		*0.0001		0.026 *0.0001	

Legend: System
Recall,Precision

Recall Significance
Precision Significance

* significance level < .01 and confidence level ≥ .99

Figure 10
Significance levels.

the corresponding systems, while the bottom of the cell presents the corresponding figure for the difference in precision.

There are two cells for each pair of systems, since a system appears in both a row and column. We present the p-value only at the intersection of the row for the higher scoring system and the column for the lower scoring system. A row containing many p-values indicates that the system for that row produced relatively high scores. The actual values show the significance of the differences in effectiveness. Systems with skewed effectiveness (high recall and low precision, or vice versa) show up as rows with many values, but with most of those values in either the top or the bottom of cells.

It is common in many scientific fields to reject the null hypothesis (i.e., reject the hypothesis that there are no significant differences) in exactly those cases where the p-value is less than a prespecified threshold. Some standard rejection levels are 0.01, 0.05, and 0.10. Although we prefer the most conservative 0.01 cutoff level for this data, we present the raw p-values to allow readers to draw their own conclusions.

The p-value computed by approximate randomization is an approximation of the value that would be computed by the computationally intractable exact randomization procedure. Noreen (1989) gives tables for computing the confidence level for a given rejection level, that is, what the probability is that the hypothesis would be rejected by exact randomization given that approximate randomization produces a particular estimate of the p-value. Rather than presenting a table of confidence levels, we put an asterisk in front of p-values in Figure 10 that have a confidence level of 0.99 or better for a rejection level of 0.01. The values with asterisks are those for which we can reject the null hypothesis at the most conservative cutoff with high confidence. Overall the confidence levels for the approximate randomization test indicate that these pairwise tests are adequate for our purposes compared with the exact randomization test run for the same data.

5.2 Analysis

The results of the approximate randomization test are intuitive in that the systems whose scores are numerically very different also have a significance level less than 0.01. Likewise, systems whose scores are numerically very close have a significance level higher than 0.10. However, it should be noted that the same difference in scores may not equate to the same significance level when different systems are involved. The test measures the difference from what would have occurred randomly, and each pair of systems provides a different set of conditions for the shuffle. Thus, we cannot conclude from the approximate randomization tests that the evaluation scores are generally significant within a certain range of percentage points.

The approximate randomization tests with 9,999 shuffles show that 59% (62 out of 105) of the pairs differ with significance levels less than 0.10 for both recall and precision. Similarly, 39% (41 out of 105) differ in exactly one score. Of those, 27 differ in recall (26% of the total) and 14 differ in precision (13% of the total). Only 2% (2 out of 105) of the pairs do not differ in either score. At the most conservative significance level of 0.01 with confidence of 1.000 and 9,999 shuffles, 60% (126 out of 210) differ in at least one of their scores, and only 31% (33 out of 105) differ in both of their scores. Under these conservative conditions, 11% of the pairs do not differ in either score (12 out of 105). Although it was impossible to directly test a hypothesis about recall and precision simultaneously, 24% (25 out of 105 pairs of systems) were such that one system had higher values of both recall and precision, and the null hypotheses for both measures could be rejected at the 0.01 level with 0.99 confidence.

The results of the approximate randomization tests can be represented according to the groupings of systems that do not differ significantly at any of the three cutoff

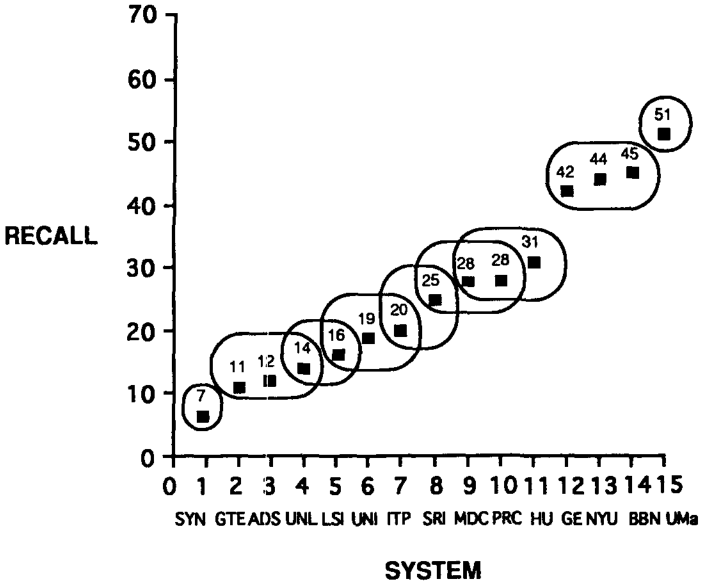


Figure 11
The systems showing no significant difference in their recall scores at the 0.10 level are grouped together.

levels with respect to either recall or precision. In Figures 11 and 12, the systems are ordered with respect to their recall and precision scores, respectively, and grouped according to significant differences. Systems are contained within the same group if they all are not significantly different from each other at the 0.10 level and all have significantly different effectiveness from all systems outside the group at the 0.10 level. If a system is in a group by itself, then it is significantly different from all other systems for that score. If groups overlap, the systems in the intersection are not significantly different from the systems in either group.

In summary, then, the use of the approximate randomization test method now allows us to report the significant differences in scores between systems participating in MUC-3. The approximate randomization test does not tell us anything about the representativeness of the sample of messages used in the MUC-3 evaluation. However, for the sample of test messages used, the approximate randomization method indicates that the results of MUC-3 are statistically different enough to distinguish the performance of most of the participating systems.

5.3 Independence and Representativeness of Observations

Any significance test assumes that observations are independent; what appear to be multiple observations of a set of variables are not in fact copies of the same observation. In one sense, the 100 MUC-3 test messages were independent observations, since, in an operational setting, each would be a separate document coming over the newswire and confronting a data extraction system.

On the other hand, there were strong dependencies among the 100 messages. Some messages described the same real world events, occasionally using similar language. There were 65 test messages that had one or more templates in the answer key, for a

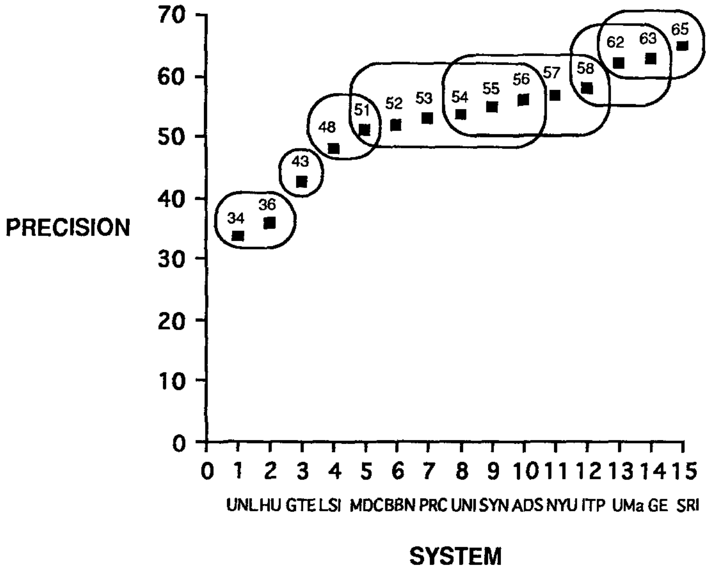


Figure 12
The systems showing no significant difference in their precision scores at the 0.10 level are grouped together.

total of 128 templates describing events. (Another 35 empty templates corresponded to messages with no reportable events.) Among these 128 filled templates, we found at least 30 that were based on the same real world event as some other template, and another 15 or so unclear cases.

Whether these duplications had an impact on our significance results depends on where data extraction systems succeeded or failed. If success or failure depended on the particulars of the language used to describe events, then, except for exactly repeated text passages, duplication did not have a significant impact on results. If success or failure is based on the nature of the event, then the presence of multiple messages about the same event resulted in the effective number of observations being reduced, with the result that insignificant differences in systems may have appeared statistically significant.

In addition to the duplication among test stories, there was duplication of events between training and test stories. This meant that system builders were able to consciously or unconsciously take advantage of prior knowledge of some of the events that would be present in the test set. For instance, on the MUC-3 corpus the proper noun *Jesuits* was statistically a very good predictor of the set fill MURDER, due to numerous descriptions of one particular attack (Lewis 1991). This effect means that the recall and precision scores we observed may be higher than those that would be observed in an operational setting.

Duplication between training and test stories is one instance of the broader question of representativeness of observations. One system might be significantly better than another on the MUC-3 test set, but the second might be better on some other text stream, or even on a different chronological period from the same text stream. (The latter issue is discussed in Section 7.5 with reference to the MUC-4 results.) Experimentation with more and larger data sets will be needed before we can have confidence in the robustness of results such as those reported here.

6. Linguistic Phenomena Test Experiment

The scores discussed above provide a *black-box* measure of system effectiveness because only input/output pairs were examined (Palmer and Finin 1990). The subsystem mechanisms used to create those outputs were not considered as they would have been in a *glass-box* test because these subsystems differed widely from system to system. In an attempt to gain some insight into the strengths and weaknesses of subsystems, we examined the effect of particular linguistic constructions on the ability of systems to extract data correctly. The general method of testing linguistic phenomena was to find all instances of the chosen phenomenon in the test messages, to determine the slots that could only be filled correctly if a system were able to handle the phenomenon, and to configure the scoring program to score just those slot instances.

Three such linguistic phenomena tests were attempted during a MUC-3 dry run in February 1991. The phenomena examined were negation, conjunction, and active versus passive verb forms. However, most systems were poor enough at extracting data from any linguistic construction during this test that differences caused by particular linguistic constructions were unnoticeable. Those differences that were observed mirrored the differences in overall scores.

The linguistic phenomena tests carried out as an experiment during the final MUC-3 run in May 1991 were more successful and showed that differences in handling of linguistic structures could be discerned through black-box testing. This was not only because the preliminary tests taught us to use a relatively frequent linguistic phenomenon but also because many sites had improved system performance in the interim, allowing meaningful scores on subsets of the data to be measured.

6.1 Hypotheses

The phenomenon of apposition was chosen for the experiment because of its frequency, its importance for slot fills, and the variety of appositive structures in the texts. A typical example of apposition from the test messages was "MSGR GREGORIO ROSA CHAVEZ, AUXILIARY BISHOP OF SAN SALVADOR, ..." To determine whether the phenomenon of apposition was being isolated, the phenomenon was tested to see if performance on affected slots was different than the overall performance.

The instances of the phenomenon in the texts were also divided into subsets to see whether the performance on the subsets would behave in a predictable way. Texts were first divided according to the complexity of the appositive construction. For example, "VENEZUELAN PRESIDENT CARLOS ANDRES PEREZ" was a simple case, whereas "UNIVERSITY STUDENTS HUGO MARTINEZ AND RAUL RAMIREZ" was complex because of the conjunction in the head noun phrase. Any complexity within the apposition, even as slight as a missing comma, put the example in the complex category. Higher scores were expected for the simpler appositive constructions.

The texts were also divided according to whether the appositive was postposed or preposed. In the examples given above, "AUXILIARY BISHOP OF SAN SALVADOR" is postposed, while "VENEZUELAN PRESIDENT" and "UNIVERSITY STUDENTS" are both preposed. It was expected that although the systems would score differently on these types of apposition, there would be no clear trend seen. Postposed appositives are more commonly thought of as appositives and are marked by some form of punctuation such as commas or dashes. However, preposed appositives could be processed as adjectival phrases.

A final approach to measuring the effect of apposition involved altering the original text. Systems were scored on modified messages where the relevant information was conveyed by a simple sentence rather than by an appositive construction. An example would be to substitute the sentence "CARLOS ANDRES PEREZ IS THE VENEZUELAN PRESIDENT" in the message and to remove the appositive, "VENEZUELAN PRESIDENT" from the original sentence. It was expected that the systems would score higher on the messages with the appositions removed.

In summary, the following hypotheses were tested:

1. The systems should score differently on the appositive constructions than they did on the overall testing.
2. The systems should score higher on the simpler appositive constructions.
3. The systems should score differently on postposed and preposed appositives.
4. The systems should score higher on their responses to messages where simple sentences were substituted for appositive constructions.

6.2 Experimental Method

All sentences from the test messages containing appositions were extracted and analyzed according to the appositions' effect on slot fills, their position (appositive preposed or postposed), and their complexity (simple or complex). Configuration files were composed that specified which slots in which templates were to be scored for appositioned noun phrases, simple appositive constructions, complex appositive constructions, preposed appositives, and postposed appositives. A set of test messages with simple sentences substituted for the appositive constructions was generated from the official test messages. The configuration files and modified messages were sent to the sites as part of the MUC-3 test package. The sites were required to automatically rescore their templates using their official history file with each of the configuration files. It took two weeks to develop the test package for the linguistic phenomena experiment and it took the sites on the order of two hours to automatically rescore their systems. Sites voluntarily chose to participate in the test that compared their performance on the 'minimal pairs' of appositive constructions and simple sentences. This test required them to run their systems on the modified test messages and to rescore using the apposition configuration file for the new templates. This part of the experiment was voluntary because the process was more time consuming than those tests that only required automatic rescoring.

6.3 Experimental Results[11]

The recall and precision scores on the apposition test are shown in the scatter plot in Figure 13. These results are quite different from the overall results for MUC-3 in Figure 2. These results indicate that the appositive constructions could be isolated by the test procedure at current levels of performance, although the systems' overall performance on the subset of involved slots may have played a role in these results.

Figure 14 shows the combined results of recall multiplied by precision for the simple and complex appositive constructions. The systems scored higher on the simpler

[11] The detailed results of the linguistic phenomena test experiment appear in the MUC-3 proceedings and contain the raw scores and plots of recall and precision (see Chinchor 1991b).

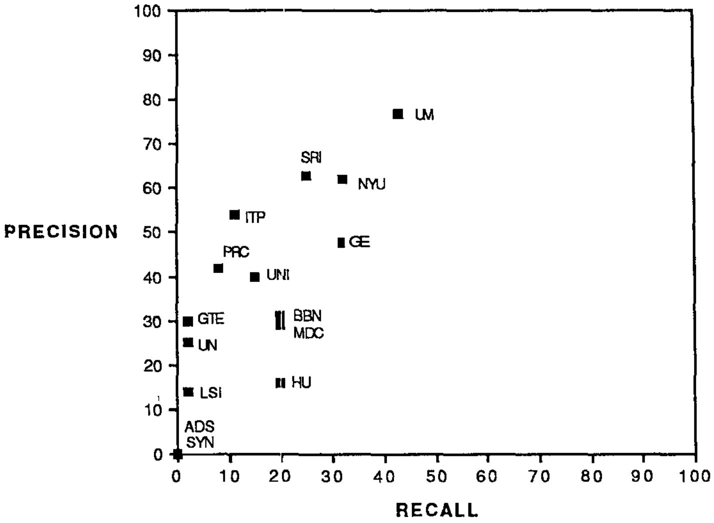


Figure 13
The scatterplot of the recall versus precision scores for appositioned noun phrases shows that the systems scored differently on the appositive constructions than on the overall test.

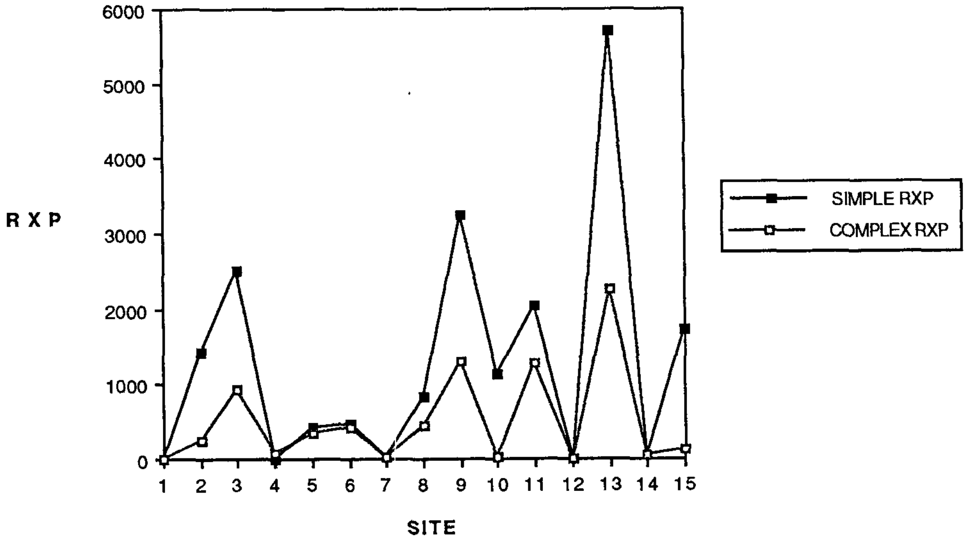


Figure 14
The plot of the product of recall and precision scores for the simple and the complex appositive constructions shows that the systems scored higher for the simpler appositive constructions as predicted.

appositive constructions than they did on the more complex appositive constructions. The results indicate strongly that the tests were isolating the linguistic phenomenon.

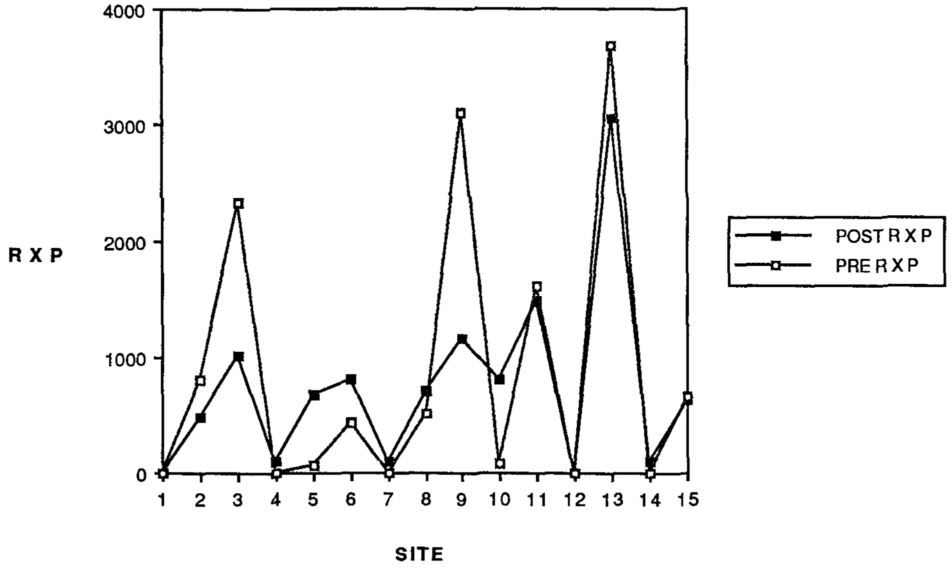


Figure 15
The plot of the product of recall and precision scores for the postposed and preposed appositives show that there is no clear trend in the scores.

Site	Recall No Appositives	Recall Appositives	Precision No Appositives	Precision Appositives
X	28	32	53	62
Y	38	43	68	77

Figure 16
The table of recall and precision scores for the 'minimal pair' type test shows that the phenomenon of apposition was being isolated by the scoring of slots because both sets of scores went down considerably as a result of the substitution of simple sentences for the appositive constructions.

The systems scored differently for the postposed and the preposed appositives, but did not score consistently higher on either. The results in Figure 15 were predicted because although the postposed appositives were more typical and usually indicated by commas or dashes, the systems could interpret the preposed appositives as adjectival.

The results for the 'minimal pair' type test were unexpectedly higher for the messages containing appositive constructions than they were for the messages containing simple substituted sentences. Two systems volunteered to run the test and, for both, the results in Figure 16 showed that the exact opposite of the hypothesis was true. The explanation was that both systems filtered out sentences containing the copula because this sentence type did not contain information relevant to the task most of the time. However, the drop of about 5% in the recall scores and 9% in the precision scores indicates that we were isolating the phenomenon of apposition even though the change is in the opposite direction of what was originally expected. The scores did not drop more when the appositive constructions were transformed because part of the information still remained in the original sentence, such as the full name of the perpetrator or target.

6.4 Conclusions Concerning the Linguistic Phenomena Test Experiment

The linguistic phenomena test experiment gives several strong indications that linguistic phenomena can be isolated by scoring the affected slots in the system responses. The development of a range of linguistic phenomena tests spanning the levels of linguistic structure from morphology to discourse would focus attention on the generality of the underlying linguistic mechanisms.

7. Lessons Learned

What lessons have we learned from studying the systems and the MUC-3 application? There are several questions to consider:

- What have we learned about the state of the art in message understanding?
- Can we identify specific system techniques that provided a significant performance gain?
- What have we learned about testing isolated linguistic phenomena?
- What have we learned about evaluation itself?

7.1 State of the Art in Message Understanding

MUC-3 focused on assessing the progress of the research in message understanding in the context of a realistic application domain. The fact that there were a number of systems that were able to handle a task of this size and complexity is encouraging. Overall, the cluster of top-performing systems reported 40–50% recall, 55–65% precision, and 10–20% overgeneration.[12] This performance is certainly a milestone marking significant progress in message understanding technology.

This level of progress raises the question of how soon such systems may be ready for use in real applications. This question cannot be answered without concrete application requirements—for example, what levels of recall, precision, and overgeneration are required for a specific application? How redundant are the messages? Does the system have to match human performance? Will humans post-edit the system output before use? We cannot answer these questions; however, looking at the MUC-3 results, we can make a few observations. First, the current systems do not allow any significant trade off between precision and recall. Several sites (BBN, NYU, U Mass) reported alternative systems that produced small changes in the precision–recall tradeoff. However, most systems, particularly the linguistically based systems, did not have adjustable parameters that could be tightened or relaxed to produce different tradeoffs. Also, recall appeared to be more of a limiting factor than precision for the MUC-3 systems. No system achieved more than 51% recall, and every system had higher precision than recall. Nonetheless, it is possible to imagine applications where this level of recall would prove sufficient.

[12] The figures quoted here all refer to the "official" measure of MATCHED/MISSING scores; see Section 2.3 for an explanation of the various scoring measures.

Precision, recall, and overgeneration are not the only useful measures of system performance. For real applications, a critical issue is the length of time it takes to make the application run in a new task domain. The median preparation time reported for the systems participating in MUC-3 was 10–11 person months. Of course, these figures included basic system development and tool development, which can be looked at as a one-time cost, shortening development times for future applications. Nonetheless, reduction of cost to port to a new application is an area that needs considerably more work.

Although a number of systems reported effort in tool-building, few system developers spent time on automated knowledge acquisition or learning techniques. This is of some concern, because many systems reported that missing domain knowledge was a significant source of error. However, there were some exceptions. The Hughes system (3 person months of effort) drew heavily on automated learning and pattern classification techniques; however, its performance was substantially below that of the best systems (especially in precision). The BBN system also used some automated and semi-automated learning in several places. It is hard to quantify how important this was, but their precision and recall scores were competitive and their level of effort (6 person months) was lower than the other comparably performing systems. Finally, some systems did address the knowledge acquisition bottleneck for lexical knowledge, but not by learning techniques. NYU, for example, used a machine-readable dictionary; both GE and ITP reported using sizable domain-independent lexicons that required only minimal adaptation for new applications.

The issue of processing speed, a problem for some systems during MUCK-II, appeared to be resolved for MUC-3. The time to process 100 messages ranged from over 300 minutes to only around 30 minutes. The median processing time for all the systems was about one minute per message. Some of this speed-up was obtained by using faster hardware. Some was obtained by faster algorithms, and some was obtained by the use of preprocessing techniques, including relevance filtering. In general, it is clear that processing speed is not a major obstacle for current message understanding systems, especially as the cost of machine cycles continues to drop.

In conclusion, the progress of the field is encouraging. Systems were able to handle a large volume of text in a realistic domain and achieve reasonable precision with fairly low overgeneration. For certain applications, it may be necessary to achieve higher recall, even at the expense of precision. However, the major obstacle for message understanding systems is the cost of porting the system to new applications. This, more than limited recall, may hold back the deployment of message processing technology in "real" applications.

7.2 What We Learned about Specific Techniques

To build better systems, we would like to know which techniques worked well and which ones did not. A black-box evaluation makes it difficult to tease apart the contributing factors that made a system perform well or poorly. However, several things emerge quite clearly. First, there is a healthy diversity of approaches among the top-performing systems. These included a semantics-driven case frame system, several syntax-driven systems, and a system that used a hybrid control approach drawing heavily on semantic information.

For the syntax-driven systems, only those systems that provided a robust parsing capability performed well. Among the top-performing systems, there were two successful approaches to robust parsing: use of partial parsing and use of heuristics added to a standard full-sentence parser that allowed recovery of parsed substrings.

As always, acquisition of domain knowledge (lexical semantics and an adequate domain model) proved a major task. For many systems, the incompleteness of this domain knowledge was a significant source of error; extensions to the domain knowledge were likely to produce significant performance improvement. There was a range of approaches to the lexicon development and lexical semantics. At one end was the highly domain-specific knowledge engineering approach of the U Mass system. In the middle were systems like NYU, which used a machine-readable dictionary for syntactic information augmented by hand-coded semantics, or the BBN system, which did automatic part-of-speech tagging but required hand-coding of semantic rules (with some computer aids). An approach that seemed to work well was the use of a handcrafted but domain-independent, semantically rich lexicon (GE, ITP). The other end of the lexicon/semantic knowledge spectrum was represented by the Hughes system, which "learned" its lexicon and semantics from the training corpus. This is appealing because it makes the approach highly portable. However, the approach was not among the higher-scoring systems.

The proper treatment of discourse turned out to be important for the MUC-3 task, and many system developers identified this as an area needing substantially more work. Most systems had trouble distinguishing elaboration of previously mentioned events from the introduction of new events. Many systems lacked a systematic approach to the problem and relied on some simple heuristics ("collapse two events if they are of the same type, with the same target, and occur on the same day"), sometimes coupled with the use of keywords ("meanwhile," "in sum") to help detect a change of topic.

Preprocessing was a key feature in many of the MUC-3 systems. Relevance filtering, either statistically trained or knowledge-based, was successfully used by several systems to reduce processing load. The use of text categorization techniques from information retrieval was more successful in this role than when used as the sole method for filling templates. Systems also used other kinds of preprocessing, including special handling of Spanish names (BBN, SRI), regularization of special forms such as dates, organization names, and place names (many sites), reduction of syntactic complexity by bracketing out certain structures (GE), and flagging of discourse segment markers (GE, ITP). It is clear that, in an application involving large volumes of text, such preprocessing techniques provide an effective way to trade off a requirement for speed with a requirement for accuracy.

7.3 Testing Linguistic Phenomena

The linguistic phenomena testing was an attempt to gain insight into the inner workings of the systems by performing tests that had some of the characteristics of glass-box tests but were still fundamentally black-box in nature. The phenomena tests were designed to be system-independent and to give an indication of how well systems were solving various aspects of the text processing problem. Although the phenomena test experiment reported here indicates that it was possible to isolate linguistic phenomena using the scoring mechanisms of MUC-3, there were a number of confounding factors that degraded the capability to determine exact performance on linguistic phenomena. The use of alternate and optional slot fillers in the answer key affected the clarity of the results. Some of the scoring guidelines for giving partial credit also affected the clarity of the results. However, the largest noticeable effects were due to system-dependent operations. Some systems did not attempt to fill certain slots at all, due to limitations on development time. Their scores on the phenomena tests really do not indicate their capability with respect to those phenomena tested. The more robust the systems were in terms of the task, the more indicative the phenomena tests were of their capabilities in certain linguistic areas of text processing. However, these tests must be designed and analyzed with awareness of the strategies used by the task designers and the system designers before solid conclusions can be drawn as to performance on particular linguistic phenomena.

7.4 What We Learned about Evaluation

Perhaps the thing that we learned the most about was evaluation itself: the process, the costs, the payoffs, and the pitfalls. On the whole, the participants—the system developers, the people responsible for the evaluation process, and the observers—all felt strongly that this evaluation was successful and that it was worth the (very large) amount of time and effort. The format of the conference required that participants describe their system in detail and provide an analysis of what did and did not work. This enabled all participants to learn from each other's experiences, which greatly increased the value of the exercise. The fact that the participants were involved in the planning of the conference and defining the methods used for evaluation also contributed to its success. Overall, our conclusions are that evaluation is costly; it requires the investment of substantial resources, both to create the evaluation infrastructure and to port the systems to the chosen application. This kind of evaluation can only be successful if system developers feel that they benefit from their participation by gaining new insights into their own systems in relation to alternatives represented by other systems.

Another lesson we learned is that black-box evaluation is good for getting a snapshot of the field, but it is not necessarily a good predictor of future system performance. Several MUC-3 developers showed interesting statistics indicating that their performance was increasing steadily for each week of continued development, with no significant fall-off as MUC-3 approached. This is strong evidence that these systems will show improved performance if given more time for development. Other systems suffered from slow start-up, limited resources, or missing components that prevented them from achieving peak performance.

Successive black-box evaluations may be required to show whether a system has "topped out" or whether routine bug fixes, new modules, and additional knowledge produce further performance improvements. This observation has led to a new proposal to measure system "convergence" (Hirschman 1991a). The proposal is to perform an initial evaluation using two test sets, S1 and S2. Following this, each site would be allowed to use test set S1 for development for a short period of time (perhaps a few days), but would not be allowed to look at test set S2. At the end of this period, the system would again be scored on both test sets. The *improvement* on test set S2 relative to the *improvement* on test set S1 would provide some measure of how well the system was converging—whether changes made to fix problems in one test set actually helped in another test set.

Another lesson was that black-box evaluation is not effective for determining which techniques are responsible for good performance across systems. Performance trade-offs are very system-specific, and insights depend on a careful analysis of how the particular system failed. SRI provided such an analysis of errors for a subset of the test messages, and this gave some interesting insights into that particular system. However, it is difficult to draw any cross-system comparisons. If we wish to have more consistent insights into the strengths and weaknesses of components within individual systems, we will have to incorporate glass-box measures or rely on more sophisticated tests such as the linguistic phenomena tests described in Section 6. In addition, we need effectiveness measures that go beyond recall and precision to take into account the structured nature of the information being extracted. Such measures might require the use of a structured database to evaluate the effectiveness of the retrieval of certain kinds of information.

7.5 MUC-4

Although this paper is about MUC-3, MUC-4 has already taken place, so it seems appropriate to discuss it briefly here. For more on the results and system details, the reader is referred to the conference proceedings, *Proceedings of the Fourth Message Understanding Conference (MUC-4)*. MUC-4 evaluated the performance of 17 systems on the same domain as MUC-3. Six of the participants were new entrants and eleven were veterans of MUC-3. There were some good first-time systems among the new entrants, and some of the veteran systems were dramatically revised.

The major changes to the evaluation consisted of altering the template to better reflect text processing capabilities and altering the scoring to be more consistent and more demanding. The scoring changes included:

- Greater automation of the scoring;
- Enforcement of mapping templates to the answer key based on content of the templates instead of allowing mapping purely by optimization of the scores;
- A focus on the stricter ALL TEMPLATES row as the official score instead of the MATCHED/MISSING row;
- The calculation of a single score known as the F-measure (van Rijsbergen 1979) combining recall and precision with variable weights;
- The calculation of a *region of performance* on the precision–recall graph for each system using the MATCHED/MISSING, MATCHED ONLY, MATCHED/SPURIOUS, and ALL TEMPLATES scores as the four corners; and
- The measurement of text-filtering capabilities, to give an indication of how well systems judged the relevance of messages.

In addition, results of three adjunct tests were reported at MUC-4. The first was an analysis of the text-filtering capabilities of the MUC-3 and MUC-4 systems (Lewis and Tong 1992). The second test was an analysis of the effect of discourse complexity on the performance of the MUC-4 systems (Hirschman 1992). This test looked at message subsets based on how many relevant sentences occurred in the message and how many templates were generated. The third adjunct test showed that the "flat" template design of MUC-4 with cross references closely approximates an object-oriented template design, but that the object-oriented design allows the collection of additional performance data for diagnosis (Krupka and Rau 1992).

There were two blind test sets in MUC-4 labeled TST3 and TST4. TST3 was a set of 100 messages taken from the same chronological segment of the FBIS corpus as the test and the development sets of MUC-3. TST4, on the other hand, consisted of 100 messages that had been produced up to two years earlier than TST3 and the MUC-3 data sets. Testing of systems on TST4 was intended to detect whether systems had been tuned too heavily to texts produced during a particular period of time. System scores for TST4 were largely in agreement with those for TST3, suggesting that systems were not in fact overtuned to the events that occurred during a particular period.

The progress of the veteran participants between MUC-3 and MUC-4 was measured by forward-converting their MUC-3 TST2 templates to the MUC-4 format and scoring TST2 and TST3 using the MUC-4 scoring methods. The results for TST2 and TST3 could then be meaningfully compared for the systems. All but one system scored higher on TST3. A typical improvement in the ALL TEMPLATES F-measure for systems was ten points with two systems increasing at least twice that much.

The TST3 and TST4 tests and the TST2/TST3 progress test show that the MUC-4 results are an improvement over MUC-3. However, we are cautious about stating exactly how much improvement there has been in the state of the art of data extraction since MUC-3. The degree of improvement is hard to quantify, since we lack good data on the kinds of variability found in FBIS and other message streams and how these factors affect data extraction systems. MUC-5, which will feature new domains and sources of text, will increase our understanding of these issues. However, the data sets developed for MUC-3 and MUC-4 will continue to be valuable resources for future experimentation.

8. Conclusion

In this paper, we have sketched the evaluation techniques that were applied to 15 text processing systems during MUC-3. In addition to the raw results, we have introduced a method of computing significance for the results across systems using approximate randomization. The results showed that the systems fell into a number of distinct clusters for both precision and recall. In many cases, the systems performing well in precision also performed well in recall. We can conclude that the evaluation methodology used in MUC-3 can indeed discriminate among systems and that there were significant differences in effectiveness among the systems fielded for MUC-3.

We have also been able to draw several other conclusions. The first is that all systems performed worse in recall than in precision; furthermore, none of the linguistically based systems had adjustable parameters to increase recall at the expense of precision (although several systems could be run in several configurations to produce slight changes in overall precision and recall). Achievement of high recall scores (over 60%) is a problem for the current text-understanding systems. However, several systems have shown steady improvement with time, and their performance may show further improvement with continued development. Even some of the high-performing systems may not yet have reached peak performance.

This raises the issue of portability—most systems spent approximately one person year preparing their systems to run for MUC-3. If porting takes 12 months of time for highly trained system developers, portability will be a serious stumbling block both to building real systems and to changing the evaluation paradigm. At the end of MUC-3, the participating system developers did not want to spend another year porting their system to yet another evaluation application. This underscores the need for serious research on portable systems.

Generality of linguistic coverage is an important part both of system portability and overall system performance. In order to evaluate the linguistic coverage, we devised a successful method for isolating specific linguistic phenomena and measuring system performance on them within the black-box framework, even though specifics of the performance of systems varied and made these tests more difficult to interpret. Development of a suite of such tests with adequate linguistic coverage would provide insight into how the handling of certain common linguistic phenomena relates to overall system performance. This insight would be similar to that obtainable through glass-box testing, but the method of testing is still technically a black-box method because it looks only at the output given certain input.

We had hoped to draw conclusions concerning the relative effectiveness of the various language processing techniques used by the participating systems. However, the results of MUC-3, even with their statistical significance now known, do not support the recommendation of one approach over another. We did notice, however, that pattern-matching and information retrieval techniques proved successful only when combined with linguistic techniques. Used in isolation, these techniques were not powerful enough to extract the wide range of information needed in the MUC-3 task. We also noted that robust processing techniques (filtering, skimming, robust or partial parsing) were important for good performance.

Overall, we believe that the MUC conferences have made a major contribution to evaluation methodology. They have both benefited from and inspired other work on evaluation, as evidenced by the growing body of research on evaluation techniques for natural language understanding including two workshops on the subject: Palmer and Finin, 1990 and Neal and Walter, 1991. In glass-box evaluation, researchers have developed a parse evaluation methodology (Black et al. 1991) that is now in use. There is active research in the evaluation of spoken language understanding: Hirschman et al., 1992 and Price et al., 1992. In addition, there is a new effort in evaluation of machine translation systems based on quality assessment measures and use of a multiple choice reading comprehension test. This work is beginning to provide the evaluation techniques that will enable the natural language community to assess its progress, to understand its results, and to focus future research towards robust, high-performance systems capable of handling real-world applications.

Acknowledgments

We wish to acknowledge the contributions of many individuals without whom we would not have results to discuss here. First, there are the individual system developers listed by sites in Figure 1; second, there is the Program Committee, which was responsible for the general framework of the conference and many of the detailed decisions about evaluation. Next, the government attendees at MUC-3 provided insight into possible requirements and applications of such systems; their enthusiasm was contagious and made us feel that years of research in natural language processing was ready to bear fruit. And finally, we wish to acknowledge the key people responsible for making the conference happen: Beth Sundheim of NRaD, who, as organizer of all of the Message Understanding Conferences, has made a major contribution to the state of evaluation for natural language systems; and Charles Wayne and Thomas Crystal of the Advanced Research Projects Agency/Software and Intelligent Systems Technology Office, whose financial and moral support for MUC-3 and MUC-4 were invaluable.

References

Black, E.; Abney, S.; Flickinger, D.; Gdaniec, C.; Grishman, R.; Harrison, P.; Hindle, D.; Ingria, R.; Jelinek, F.; Klavans, J.; Liberman, M.; Marcus, M.; Roukos, S.; Santorini, B.; and Strzalkowski, T. (1991). "A procedure for quantitatively comparing the syntactic coverage of English grammars." In *Proceedings, Speech and Natural Language Workshop*. Pacific Grove, CA. February 1991, 306–311.

Chatfield, C. (1988). *Problem Solving: A Statistician's Guide*. Chapman and Hall.

Chinchor, N. (1991a). "Evaluation metrics." In *Proceedings, Third Message Understanding Conference (MUC-3)*. Morgan Kaufmann. San Mateo, CA.

Chinchor, N. (1991b). "Linguistic phenomena test experiment." In *Proceedings, Third Message Understanding Conference (MUC-3)*. Morgan Kaufmann. San Mateo, CA.

Chinchor, N. (1992). "The statistical significance of the MUC-4 results." In *Proceedings, Fourth Message Understanding Conference (MUC-4)*. Morgan Kaufmann. San Mateo, CA.

Cohen, P., ed. (1991). *Proceedings, Workshop on AI Methodology.* Amherst, MA, June.

Efron, B., and Tibshirani, R. (1991). "Statistical data analysis in the computer age." *Science,* 253, 390–395.

Hirschman, L. (1991a). "Evaluation for language understanding systems." In *Proceedings, Workshop on AI Methodology,* edited by P. Cohen, Amherst, MA, June 1991.

Hirschman, L. (1991b). "Comparing MUCK-II and MUC-3: Assessing the difficulty of different tasks." In *Proceedings, Third Message Understanding Conference (MUC-3).* Morgan Kaufmann. San Mateo, CA.

Hirschman, L. (1992). "An adjunct test for discourse processing in MUC-4." In *Proceedings, Fourth Message Understanding Conference (MUC-4).* Morgan Kaufmann. San Mateo, CA.

Hirschman, L.; Bates, M.; Dahl, D.; Fisher, W.; Garofolo, J.; Hunicke-Smith, K.; Pallett, D.; Pao, C.; Price, P.; and Rudnicky, A. (1992). "Multi-site data collection for a spoken language corpus." In *Proceedings, International Conference on Spoken Language Processing (ICSLP-92).* Banff, Canada.

Krupka, G., and Rau, L. (1992). "GE adjunct test report: Object-oriented design and scoring for MUC-4." In *Proceedings, Fourth Message Understanding Conference (MUC-4).* Morgan Kaufmann. San Mateo, CA.

Lehnert, W., and Sundheim, B. (1991). "An evaluation of text analysis technologies." *AI Magazine,* 12(3), 81–94.

Lewis, D. D. (1991). "Data extraction as text categorization: An experiment with the MUC-3 corpus." In *Proceedings, Third Message Understanding Conference (MUC-3).* Morgan Kaufmann. San Mateo, CA.

Lewis, D. D., and Tong, R. M. (1992). "Text filtering in MUC-3 and MUC-4." In *Proceedings, Fourth Message Understanding Conference (MUC-4).* Morgan Kaufmann. San Mateo, CA.

Neal, J. G., and Walter, S. M., eds. (1991). *Natural Language Processing Systems Evaluation Workshop.* Berkeley, CA. June 1991.

Noreen, E. W. (1989). *Computer Intensive Methods for Testing Hypotheses: An Introduction.* John Wiley & Sons.

Palmer, M., and Finin, T. (1990). "Workshop on the Evaluation of Natural Language Processing Systems." *Computational Linguistics,* 16(3), 175–181.

Price, P.; Hirschman, L.; Shriberg, E.; and Wade, E. (1992). "Subject-based evaluation measures for interactive spoken language systems." In *Proceedings, Speech and Natural Language Workshop.* Arden House, NY, February 1992.

Proceedings, Third Message Understanding Conference (MUC-3). (1991). Morgan Kaufmann. San Mateo, CA.

Proceedings, Fourth Message Understanding Conference (MUC-4). (1992). Morgan Kaufmann. San Mateo, CA.

van Rijsbergen, C. J. (1979). *Information Retrieval.* Butterworth.

Santorini, B. (1990). "Part-of-speech tagging guidelines for the Penn TREEBANK project (3rd revision)." MS-CIS-90-47. CIS Department, University of Pennsylvania. Philadelphia, PA.

Swets, J. A., ed. (1964). *Signal Detection and Recognition by Human Observers.* John Wiley & Sons.

Swets, J. A. (1969). "Effectiveness of information retrieval methods." *American Documentation,* 72–89.