BBN's PLUM Probabilistic Language Understanding System

*The PLUM System Group**

BBN Systems and Technologies
70 Fawcett Street
Cambridge, MA 02138
weischedel@bbn.com

1. APPROACH

Traditional approaches to the problem of extracting data from texts have emphasized hand-crafted linguistic knowledge. In contrast, BBN's **PLUM** system (**P**robabilistic **L**anguage **U**nderstanding **M**odel) was developed as part of an ARPA-funded research effort on integrating probabilistic language models with more traditional linguistic techniques. Our research and development goals are:

- Achieving high performance in objective evaluations, such as the Tipster evaluations.
- Reducing human effort in porting the natural language algorithms to new domains and to new languages.
- Providing technology that is scalable to realistic applications.

We began this research agenda approximately three years ago. During the past two years, we have ported our data extraction system (PLUM) to a new language (Japanese) and to two new domains.

2. KEY SYSTEM FEATURES

Three key design features distinguish PLUM from other approaches: statistical language modeling, learning algorithms and partial understanding. The first key feature is the use of *statistical modeling to guide processing*. For the version of PLUM used in MUC-5, part of speech information was determined by using well-known Markov modeling techniques embodied in BBN's part-of-speech tagger POST [5]. We also used a correction model, AMED [3], for improving Japanese segmentation and part-of-speech tags assigned by JUMAN. For the microelectronics domain, we used a probabilistic model to help identify the role of a company in a capability (whether it is a developer, user, etc.). Statistical modeling in PLUM contributes to portability, robustness, and trainability.

The second key feature is our use of *learning algorithms* both to obtain the knowledge bases used by PLUM's processing modules and to train the probabilistic algorithms. We feel the key to portability of a data extraction system is automating the acquisition of the knowledge bases that need to change for a particular language or application. For the MUC-5 applications we used learning algorithms to train POST, AMED, and the template-filler model mentioned above. We also used a statistical learning algorithm to learn case frames for verbs from examples (the algorithm and empirical results are in [4]).

A third key feature is *partial understanding*, by which we mean that all components of PLUM are designed to operate on partially interpretable input, taking advantage of information when available, and not failing when information is unavailable. Neither a complete grammatical analysis nor complete semantic interpretation is required. The system finds the parts of the text it can understand and pieces together a model of the whole from those parts and their context.

3. PLUM SYSTEM DESCRIPTION

The PLUM architecture is presented in Figure 3-1. Ovals represent declarative knowledge bases; rectangles represent processing modules. The arrows connecting the processing modules indicate a roughly sequential processing of the sentences of an input document through the modules. After the message reader processes the whole document, each sentence is processed through the morphological analyzer, concept-based pattern matcher, parser, semantic interpreter, and the anaphora resolution portion of the discourse processor. After all the sentences are processed in this fashion, final document-level discourse processing and template-generation take place.

A more detailed description of the system components, their individual outputs, and their knowledge bases is presented in Ayuso et al., [1]. The processing modules are briefly described below.

3.1 Message Reader

This module is like the "text zoner" of Hobbs' description of generic data extraction systems. PLUM's specification of the input format is a declarative component of the message reader, allowing the system to be easily adapted to handle different formats. The input to the PLUM system is a file containing one or more messages. The message reader module determines message boundaries, identifies the message header information, and determines paragraph

*Ralph Weischedel (Principal Investigator), Damaris Ayuso, Sean Boisen, Heidi Fox, Tomoyoshi Matsukawa, Constantine Papageorgiou (BBN), Dawn MacLaughlin, Masaichiro Kitawa, Tsutomu Sakai (Boston University), June Abe, Hiroto Hosihi, Yoichi Miyamoto (University of Connecticut), and Scott Miller (Northeastern University) and sentence boundaries. To date, we have designed format specifications for about half a dozen domains.

3.2 Morphological Analyzer

The first phase of sentence processing is assignment of part-of-speech information to the words, e.g., proper noun, verb, adjective, etc. In BBN's part-of-speech tagger POST [5], a bi-gram probability model, frequency models for known words (derived from large corpora), and probabilities based on word endings for unknown words are employed to assign part of speech to the highly ambiguous known and unknown words of the corpus. POST tags each word with one of 47 possible tags with 97% accuracy for known words. For the Japanese domains, JUMAN is used to propose word segmentation and part-of-speech assignments, which are then corrected by AMED [3] before being handed to POST for final disambiguation. Below are the part-of-speech tags produced by POST for the first sentence of the EJV walkthrough article 0592:

"BRIDGESTONE SPORTS CO. SAID FRIDAY IT HAS SET UP A JOINT VENTURE IN TAIWAN WITH A LOCAL CONCERN AND A JAPANESE TRADING HOUSE TO PRODUCE GOLF CLUBS TO BE SHIPPED TO JAPAN."

(BRIDGESTONE NP) (SPORTS NPS) (CO. NP)
(SAID VBD) (FRIDAY NP) (IT PP) (HAS VBZ)
(SET UP VBN) (A DT) (JOINT VENTURE NN) (IN IN)
(TAIWAN NP) (WITH IN) (A DT) (LOCAL JJ)
(CONCERN NN) (AND CC) (A DT) (JAPANESE JJ)
(TRADING HOUSE NN) (TO TO) (PRODUCE VB)
(GOLF NN) (CLUBS NNS) (TO TO) (BE VB)
(SHIPPED VBN) (TO TO) (JAPAN NP) (. .)

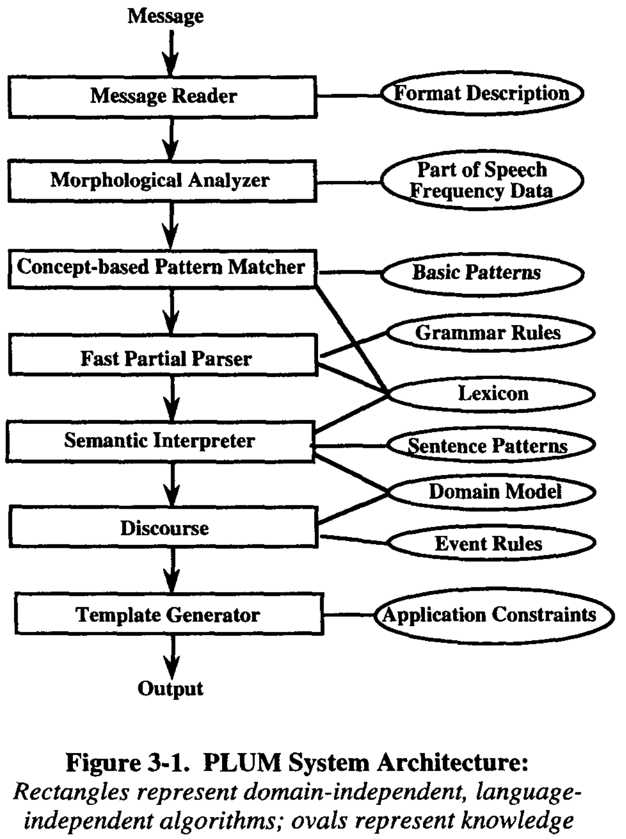


Figure 3-1. PLUM System Architecture:
Rectangles represent domain-independent, language-independent algorithms; ovals represent knowledge bases.

3.3 Concept-Based Pattern Matcher

The concept-based pattern matcher was developed after MUC-4 to deal with certain grammatical forms, such as corporation names. In particular, word groups that are important to the domain and that may be detectable with only local lexical information can be treated here. The concept-based pattern matcher applies finite-state patterns to the output of POST, which consists of word tokens with part-of-speech information. The patterns in addition have access to lexical semantic information, which includes a semantic type (or concept) and other semantic information. When a pattern is matched, a semantic form is assigned by the action component of the pattern to the word sequence. This assignment has two effects: the word sequence is temporarily defined in PLUM's lexicon, and the input is re-tokenized to treat the word sequence as a single unit for further processing. Lexical items added by the pattern actions remain active for the duration of the document's processing. In this way, subsequent sentences may recognize, for example, aliases of corporations identified in previous sentences.

The patterns used by the concept-based pattern-matcher may be grouped by "phases", indicating multiple pattern-matching passes. Phases following the first phase operate on the input as modified by the previous phase. We have used at most two phases in our applications so far. In both joint ventures and microelectronics, patterns were used to group proper nouns into company names, organization names, and person names. Continuing with the example sentence discussed above, a pattern recognized the sequence (BRIDGESTONE NP) (SPORTS NPS) (CO. NP) as a company; the pattern's action substituted the single token (BRIDGESTONE SPORTS CO. CORP), with semantics of corporation.

3.4 Fast Partial Parser (FPP)

The FPP is a near-deterministic parser which generates one or more non-overlapping parse fragments spanning the input sentence, deferring any difficult decisions on attachment ambiguities. When cases of permanent, predictable ambiguity arise, the parser finishes the analysis of the current phrase, and begins the analysis of a new phrase. Therefore, the entities mentioned and some relations between them are processed in every sentence, whether syntactically ill-formed, complex, novel, or straightforward. Furthermore, this parsing is done using essentially domain-independent syntactic information.

FPP averages about 6 fragments for sentences as complex as in the EJV corpus; this number is inflated since punctuation usually results in an isolated fragment. Continuing with the same example sentence, Figure 3-2 shows nine parse fragments as generated by FPP. The Japanese grammar produces smaller fragments by design.

3.5 Semantic Interpreter

The semantic interpreter contains two sub-components: a rule-based fragment interpreter and a pattern-based sentence interpreter. The first was used in MUC-3 and MUC-4. The second subcomponent was added before MUC-5.

3.5.1 Rule-based Fragment Interpreter

The rule-based fragment interpreter applies semantic rules to each fragment produced by FPP in a bottom-up, compositional fashion. Semantic rules are matched based on general syntactic patterns, using wildcards and similar mechanisms to provide robustness. A semantic rule creates a semantic representation of the phrase as an annotation on the syntactic parse. A semantic form includes a variable (e.g., ?13), its type, and a collection of predicates pertaining to that variable. There are three basic types of semantic forms: entities in the domain, events, and states of affairs. Each of these can be further categorized as known, unknown, and referential. Entities correspond to the people, places, things, and time intervals of the domain. These are related in various ways, such as through events (who did what to whom) and states of affairs (properties of the entities). Entity descriptions typically arise from noun phrases; events and states of affairs are often described in clauses.

The rule-based fragment interpreter encodes defaults so that missing semantic information does not produce errors, but simply marks elements or relationships as unknown. Partial understanding is critical to text processing systems; missing data is normal. For example, the generic predicate PP-MODIFIER is used to indicate that two entities are connected via a certain preposition when no more specific information is known. In this way, the system has a "placeholder" for the information that a certain structural relation holds, even though it does not know what the actual semantic relation is. Sometimes understanding the relation more fully is of no consequence, since the information does not contribute to the template-filling task. The information is maintained, however, so that later expectation-driven processing can use it if necessary.

In Figure 3-3 we show the semantic representation that is built by the rule-based semantic interpreter for the phrase "THE JOINT VENTURE, BRIDGESTONE SPORTS TAIWAN CO., CAPITALIZED AT 20 MILLION NEW TAIWAN DOLLARS" in EJV walkthrough article 0592 (this phrase is parsed within a single fragment by FPP). Notice that the JOINT-VENTURE is linked to the OWNERSHIP information via an unknown role, because the interpreter was unable to determine a specific relationship between the NP "THE JOINT VENTURE, BRIDGESTONE SPORTS TAIWAN CO.," and the participial modifier "CAPITALIZED AT ..." The discourse component will further refine the relationship between these two semantic objects to the JV-OWNERSHIP-OF relation.

```
F1: "BRIDGESTONE SPORTS CO. SAID FRIDAY IT
HAS SET UP A JOINT VENTURE"
  (S (NP (N (NAME "BRIDGESTONE SPORTS CO.")))
    (VP (AUX)
      (VP (V "SAID")
        (NP (MONTH "FRIDAY"))
        (S
          (S (NP (PRO-DET-SPEC "IT"))
            (VP (AUX (V "HAS"))
              (VP (V "SET UP")
                (NP (DETERMINER "A")
                    (N "JOINT VENTURE")))))))))
F2: "IN TAIWAN"
  (PP (PREP "IN")
      (NP (N (NAME "TAIWAN"))))
F3: "WITH A LOCAL CONCERN"
  (PP (PREP "WITH")
      (NP (DETERMINER "A")
          (ADJP (ADJ "LOCAL"))
          (N "CONCERN")))
F4: "AND"
  (CONJ "AND")
F5: "A JAPANESE TRADING HOUSE"
  (NP (DETERMINER "A")
      (ADJP (ADJ "JAPANESE"))
      (N "TRADING HOUSE"))
F6: "TO PRODUCE GOLF CLUBS"
  (VP (AUX (TO "TO"))
      (VP (V "PRODUCE")
          (NP (N "GOLF") (N "CLUBS"))))
F7: "TO"
  (PREP "TO")
F8: "BE SHIPPED TO JAPAN"
  (VP (AUX (V "BE"))
      (VP (V "SHIPPED")
          (PP (PREP "TO")
              (NP (N (NAME "JAPAN"))))))
F9: "."
  (PUNCT ".")
```

Figure 3-2. Parser Output: *Partial parse found for the example sentence.*

An important consequence of the fragmentation produced by FPP is that top-level constituents are typically more shallow and less varied than full sentence parses. As a result, a fairly high level of semantics coverage can be obtained quite quickly when the system is moved to a new domain. This would not be possible if the semantic rules were required to cover a wider variety of syntactic structures before it could achieve reasonable performance. In this way, semantic coverage can be added gradually, while the rest of the system is progressing in parallel.

3.5.2 Pattern-based Sentence Interpreter

The second sub-component of the semantic interpreter module is a pattern-based sentence interpreter which applies semantic pattern-action rules to the semantics of each fragment of the sentence. This replaced the fragment combining component used in MUC-4. The semantic pattern-matching component employs the same core engine as the concept-based pattern matcher. These semantic rules can add additional long-distance relations between semantic entities in different fragments within a sentence. The patterns used by the sentence-level interpreter may be in terms of individual words, semantic structures, and/or syntactic structures. Unlike the fragment-combination module used in MUC-4, the rules used in this module need not be tailored to expected fragmentation by the parser. The rules may simply look for parsed phrases with certain semantic characteristics, ignoring whether such a phrase is alone in a fragment or is a part of a larger fragment. For example, in the English joint-venture domain, we have defined a rule which looks for possibly multiple instances of [<PERCENTAGE> "by" <ENTITY>]. This rule's action creates an OWNERSHIP semantic form, where <ENTITY> is related via the OWNERSHIP-ENTITY role and <PERCENTAGE> via the OWNERSHIP-% role. In the walkthrough, this rule matches in the sentence "THE NEW COMPANY, ..., IS OWNED 75 PCT BY BRIDGESTONE SPORTS, 15 PCT BY UNION PRECISION ...".

3.5.3 Lexical Semantics

The semantic lexicon is separate from the parser's lexicon and has much less coverage. Lexical semantic entries indicate the word's semantic type (a domain model concept), as well as predicates pertaining to it. For example, here is the lexical semantics for the noun collocation "joint venture".

```
(defnoun "joint venture"
  (JOINT-VENTURE
    (:CASE
      (("with" "between") ENTITY PARENT-OF)
       ("for" ACTIVITY ACTIVITY-OF))))
```

This entry indicates that the semantic type is JOINT-VENTURE, and that a "with" or "between" PP argument whose type is ENTITY should be given the role PARENT-OF, and a "for" PP argument of type ACTIVITY should be given the role ACTIVITY-OF.

We used an automatic case frame induction procedure to construct an initial version of the lexicon [4]. Word senses in the semantic lexicon have probability assignments. For MUC-5 probabilities were (automatically) assigned so that each word sense is more probable than the next sense, as entered in the lexicon.

3.6 Discourse Processing

PLUM's discourse component [2] performs the operations necessary to create a meaning for the whole message from the meaning of each sentence. The message level representation is a list of discourse domain objects (DDOs) for the events of interest in the message (e.g., JOINT-VENTURE events in the joint-venture domain, CAPABILITY events in the microelectronics domain or ENTITIES in both domains). The semantic representation of a sentence only includes information contained within it; in creating a DDO, the discourse module must infer other long-distance or indirect relations not explicitly found by the semantic interpreter, and resolve any references in the text.

The discourse component creates and maintains two primary structures: a discourse predicate database and the DDOs. The database is a propositional database supporting

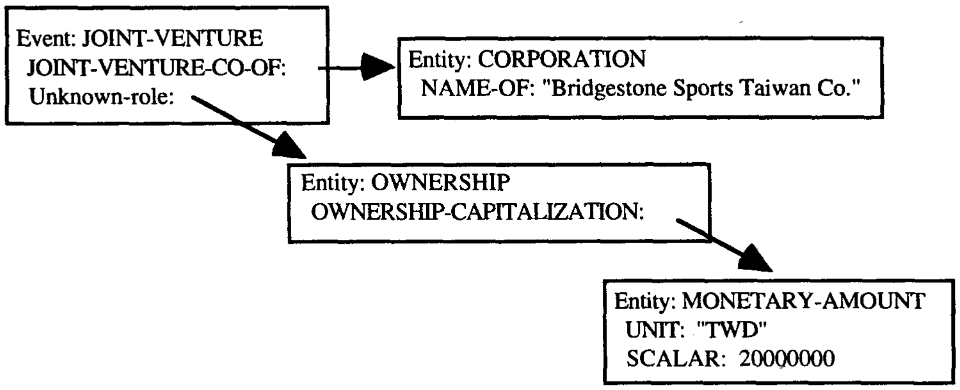


Figure 3-3. Semantic Structure: *The semantic representation for the first fragment in Figure 3-2.*

unification, and contains all the predicates mentioned in the semantic representation of the message. When references are resolved, corresponding semantic variables are unified. Other inferences may also be added to the database.

For each sentence, the discourse component processes each semantic form produced by the interpreter, adding its information to the predicate database and performing reference resolution for pronouns and anaphoric definite NPs. Set- and member-type references may be treated. When a semantic form for an event of interest is encountered, an initial DDO is generated, and any slots already found by the interpreter are filled in. The discourse processor then tries to merge the new DDO with a previous DDO, in order to account for the possibility that the new DDO might be a repeated reference to an earlier one.

This merging of discourse domain objects is itself a form of reference resolution. To check compatibility of objects before allowing merging to take place, this procedure uses the same tests used for reference resolution. In addition, other parameters are also used which may, for example, limit the distance allowed between objects considered for merging .

Once all the sentences have been processed through DDO creation and merging, heuristic rules are applied to fill any empty DDO slots by looking at the text surrounding the forms that triggered a given DDO. Each slot filler found in the text is assigned a confidence score based on distance from its trigger. Fillers found nearby are of high confidence, while those farther away receive worse scores. Low numbers represent high confidence; high numbers low confidence; thus 0 is the "highest" confidence score, used mainly for fillers obtained from predicates asserted in the semantic representation.

Following is the DDO for the first JOINT-VENTURE in EJV walkthrough article 0592:

```
DDO: JOINT-VENTURE
Trigger fragments:
  "BRIDGESTONE SPORTS CO. SAID FRIDAY IT HAS
SET UP A JOINT VENTURE"
  "THE JOINT VENTURE, BRIDGESTONE SPORTS
TAIWAN CO., CAPITALIZED AT 20 MILLION NEW
TAIWAN DOLLARS, WILL START PRODUCTION IN
JANUARY 1990"
"THE NEW COMPANY, BASED IN KAOHSIUNG,
SOUTHERN TAIWAN, IS OWNED" "CASTING CO. OF
TAIWAN"
_______________________________________________

JOINT-VENTURE-CO-OF:
  "BRIDGESTONE SPORTS TAIWAN CO." (score = 0)
JV-PARENT-OF:
  "BRIDGESTONE SPORTS CO." (score = 0)
  "A LOCAL CONCERN" (score = 2)
  "A JAPANESE TRADING HOUSE" (score = 2)
  "GOLF CLUBS" (score = 2)
  "TAGA CO" (score = 2)
JV-ACTIVITY-OF:
  "START PRODUCTION" (score = 1)
  "PRODUCE GOLF CLUBS" (score = 2)
  "BE SHIPPED TO JAPAN" (score = 2)
  "WITH PRODUCTION OF 20,000 IRON" (score = 2)
JV-OWNERSHIP-OF:
  "CAPITALIZED AT 20 MILLION NEW TAIWAN
DOLLARS" (score =0)
```

Each trigger fragment contains one or more semantic components which triggered this DDO. When a DDO has multiple trigger fragments it indicates coreference or merging took place. In this example, a "joint venture" in the first fragment co-refers with "the joint venture" in the second fragment. A score of 0 indicates the filler was found directly by the semantics; 1 that it was found in the same fragment as a trigger form; and 2 in the same sentence.

3.7 Template Generation

The template generator takes the DDOs produced by discourse processing and fills out the application-specific templates. Clearly, much of this process is governed by the specific requirements of the application, considerations which have little to do with linguistic processing. The template generator must address any arbitrary constraints, as well as deal with the basic details of formatting.

The template generator uses a combination of data-driven and expectation-driven strategies. First the DDOs found by the discourse module are used to produce template objects. Next, the slots in those objects are filled using information in the DDO, the discourse predicate database, or other sources of information such as the message header (e.g., document number, document source, and date information), statistical models of slot filling (e.g., as in the microelectronics domain to choose among the slots: purchaser/user, developer, distributor, and manufacturer), or from heuristics (e.g., the status of an equipment object is most likely to be IN_USE, or the status of a joint venture object is most likely to be EXISTING).

3.8 Parameters in PLUM

Many aspects of PLUM's behavior can be controlled by simply varying the values of system parameters. For example, PLUM has parameters to control aspects of tagging, parsing, pattern matching, event merging and slot filling by discourse, and template filling. An important goal has been to make our system as "parameterizable" as possible, so that the same software can meet different demands for recall, precision, and overgeneration.

3.9 Hardware/Software requirements

The PLUM system is implemented in CommonLisp. We have been developing the system on Sun Sparc stations as well as on SGI machines. By running the PLUM system on the TIPSTER 24-month data set on a SunSparc10 with

128M memory, we gathered the following timing information.

Application	Minutes/ message	Bytes/ minute
EJV	0.49	3,766
EME	0.72	3,416
JJV	2.35	420
JME	2.43	564

Table 3-1. Runtime Performance: *Current speed on a desktop workstation is given here; considerable speedup is feasible, since little optimization has been performed to date.*

4. ORIGINAL PROJECT/SYSTEM GOALS

Our original goal was to apply a new, unproven approach to the text understanding problem. Our new approach was to employ probabilistic models and learning algorithms with traditional sources of linguistic knowledge (e.g., part of speech for words, grammars, semantic preferences, and discourse constraints) to data extraction from text. We postulated that that would result in

- Achieving high performance in objective evaluations, such as the Tipster evaluations.
- Reducing human effort in porting the natural language algorithms to new domains and to new languages.
- Providing technology that is scalable to realistic applications.

We believe that our results, as summarized in Sections 6 and 7, represent achieving a significant milestone toward these goals, and that further research and development will provide further advances in the state of the art.

5. EVOLUTION OF SYSTEM OVER 2 YEARS

We began our research agenda approximately three years ago. During the past two years, we have focused much of our effort on techniques to facilitate porting our data extraction system (PLUM) to new languages (Japanese) and to two new domains (joint ventures and microelectronics). We have also concentrated on infrastructure development, including the addition of the two pattern matching components as well as an object-oriented template generator.

First, consider the evolution in the runtime version of PLUM, as illustrated in Figure 3-1 can be summarized as follows:

- Message reader -- Prior to this contract, the message reader was code that had to be modified for each application domain. The change was to make the code domain independent and language independent, driven by a declarative knowledge base which is modifiable for the peculiarities of a given class of text.
- Concept-based pattern matcher -- This is a new module, which processes finite state grammars of expressions that can be reliably recognized upfront. It covers organization names, person names, and other items which could be defined in the context-free grammar, but which are so simple in structure that a finite state grammar is sufficient.
- Semantic interpreter -- Coverage of the semantic interpreter was extended in two ways
  - The number of domain-independent rules was increased greatly, particularly for English
  - A phrase-based pattern matcher replaced a "fragment combining" component to robustly capture the semantics of phrases that are not parsed in a single fragment.
- Discourse – Discourse processing was improved and generalized to more accurately determine whether two text descriptions actually discussed the same thing or same event.
- Template generator -- A new template generation component was written to separate domain-independent code from domain-dependent details, to support output of object-oriented data.

Several new algorithms were added to automate the porting process, rather than being part of the runtime version of PLUM. These included:

- A component to automatically learn to classify text (paragraphs) as relevant or irrelevant, given examples of both types of text.
- A simple technique for rapidly defining jargon words.
- A technique for hypothesizing semantically related words from relatively small volumes of text, e.g., from as few as 1,000 articles.

6. ACCOMPLISHMENTS

6.1 Dealing With Multiple Languages And Multiple Domains

Any system that participated in more than one domain in MUC-5 and/or in more than one language has demonstrated domain independence and language independence. In PLUM, the text zoner, morphological processing, parsing, and semantic interpretation employ language-independent and domain-independent algorithms driven by data (knowledge) bases. Similarly, the discourse algorithms and template generation algorithms are domain- and language-independent, and are driven by knowledge that is predominantly declarative.

The issue (or the goal) that all systems must address further is greater automation of the porting process. Our approach has been to rely on probabilistic learning algorithms. Here we focus on some surprises in dealing with multiple languages and multiple domains.

6.1.1 Dealing With Multiple Domains

Porting PLUM to a new domain, even in multiple languages, takes much less effort as a result of the last two years of work. Table 1 shows the labor expended in porting PLUM to the microelectronics domain. In 52 person-days, PLUM was processing microelectronics articles in both English and Japanese, obtaining reasonable performance. Had we run PLUM at that time on the TIPS3 test sets, scores would already have been impressive in English (an ERR of 74). For Japanese, performance was 73 on test set TIPS2. (We quote the score for TIPS2, because it covered only the capabilities for which there was data at the time of the TIPS2 version of PLUM.)

When the proposal for this effort was submitted, we made a conservative assumption that all probability models would need to be re-estimated for each domain. During this effort, it became clear that, though one could re-estimate probabilities for each domain, that it was not necessary. Performance of the overall data extraction system could be adequate without domain-specific training. In fact, in moving to microelectronics, no domain-specific linguistic training was used; however, the performance in microelectronics was quite close to that in joint ventures, where domain-specific training was used. See Figure 7-1

Tasks	Person-Days
Language-independent	14
English	19
Japanese	19
TOTAL	52

Table 6-1: Effort to Port to Microelectronics

Porting the PLUM system to a new domain involves developing the domain-dependent knowledge bases, primarily the domain model, event rules, and application constraints of Figure 1. To the degree that jargon is used in the domain, the lexicon and patterns will be updated. We have generally proceeded by first building a text-to-response system with small domain coverage. In our experience, this base system can be built in about a week. Once this base system exists, development of knowledge bases can proceed in parallel.

In order to build the basic text-to-response system, the following tasks must be performed (not necessarily sequentially).

- The format description for the input texts must be specified; this enables the system to digest the text that is to be processed.
- The domain model (concepts and roles) is defined. The domain model can be defined based on the output template specification.
- An initial lexicon of important words is created. Here, we utilize a tool that assists the domain developer in classifying words (collected from a text corpus and sorted by frequency) into a small number of semantic classes. This tool allows the domain developer to quickly create a lexicon with semantic type information. The lexicon is subsequently reviewed by hand, and case frame information is added.
- An initial set of event rules is defined.
- The output template objects are defined.

Optionally, the POST part-of-speech tagger may be retrained on the new corpus. Note that we did not retrain POST in porting to the microelectronics domain, as we found the frequency models that were derived from domain-independent training were adequate.

It is also helpful to identify a key set of example sentences that the system should cover. These sentences can be used to drive initial development, as well as to track progress.

Once these tasks have been completed, it should be possible to process a text in the new domain and get some limited amount of output. The key to this achievement is the default reasoning that is part of the PLUM system. For example, default semantic, discourse, and template filling rules will be utilized where no domain-specific information is available.

After the initial system is created, the knowledge bases can be refined in parallel to achieve greater coverage. Analysis tools, such as a scoring program, can be used to identify and focus on areas in need of development.

6.1.2 Porting to Multiple Languages

Annotating data for PLUM's probabilistic model of a new language, even with little language-specific resources, proved easier than anticipated. The only resource available to us at the start was the JUMAN system from Kyoto University, which hypothesizes word segmentation and part of speech for Japanese text. Our Japanese speakers were able to annotate part of speech and word boundaries at about 1,000 words per hour, and were able to annotate syntactic structure at about 750 words per hour. Initial annotation and testing were performed using only 16,000 words plus the JUMAN lexicon; therefore, the initial port to Japanese required only about a person-week of annotation effort.

One algorithm change was made in the port to Japanese. The change was not required; the algorithms are language independent; however, by introducing an existing Japanese algorithm much labor was saved in processing Japanese text, where neither spaces nor any other delimiters mark the beginning and end of words. We had at our disposal the following:

- A rule-based Japanese morphological processor (JUMAN) from Kyoto University.

- A context-free grammar of Japanese based on part of speech labels distinct from those produced by JUMAN.

- A probabilistic part-of-speech tagger (POST) [Meteer, et al., 1991] which assumed a single sequence of words as input.

- Limited human resources for creating training data.

The architecture in Figure 6-1 was chosen to minimize labor and to maximize use of existing software. It employs JUMAN first to provide initial word segmentation of the text, an annotation-based algorithm second to correct both segmentation errors and part of speech errors in JUMAN output, and POST third both to select among ambiguous alternative segmentations/part-of-speech assignments and also to predict the part of speech of unknown words.

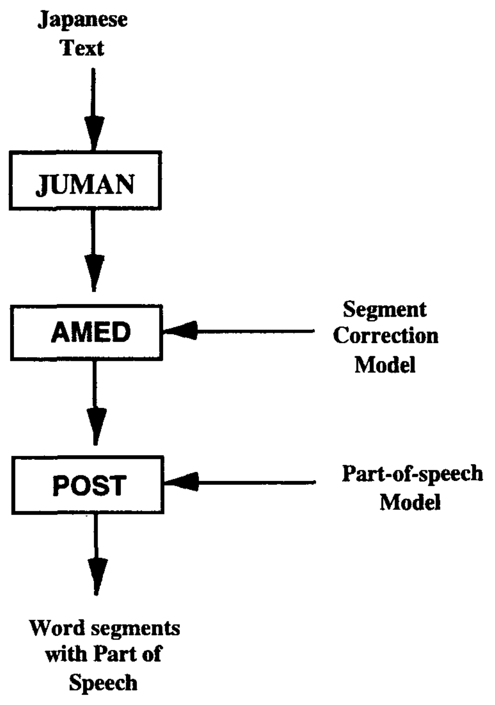


Figure 6-1. Japanese Morphological Processing: *Though POST could be used to find word boundaries in Japanese, an existing component (JUMAN) was used to save effort.*

Building lexical resources for a new language or a new domain took only a few person days using heuristics. In Japanese, a three step process for hypothesizing proper names reduced the labor involved. First, we ran JUMAN + POST over the training corpus to find the sequence of words and their most likely part of speech in context. Then, a finite-state process with a handful of language-specific patterns was run on the result to hypothesize (previously unknown) proper nouns in the corpus. The patterns were designed for high recall of names, at the expense of low precision; we measured the effectiveness of the technique as 90% recall at 20% precision. Lastly, a person ran through the hypothesized proper names using KWIC as a resource to quickly eliminate bad hypotheses. The resulting list of names was made available to all the participants in JJV.

A simple manual technique also enabled fast semantic categorization of the nouns and verbs of each domain in both languages. Using a KWIC index and the frequency of each noun and each verb in the corpus, we could define about 125 words per hour, placing each word into categories such as HUMAN, CORPORATION, OFFICER, GOVERNMENT-ORGANIZATION, etc. The process could go so quickly by organizing the categories into small menus of at most 12 items, so that a person need only make simple discriminations in any pass through a list of words.

6.2 Demonstration Prototype

Since data extraction from text is a new capability, the value of demonstration prototypes became clear, not only as a medium to communicate what the capability offers potential users and also as a trigger to ignite thoughts of how the technology could become a usable aid to users. We developed a single, MOTIF-based, graphical user interface to both domains (joint ventures and microelectronics) and both languages, (English and Japanese), for a total of 4 applications English joint ventures, Japanese joint ventures, English microelectronics, and Japanese microelectronics. The demonstration emphasized the perspective of a potential user of the data extraction technology. A message was selected and processed by the PLUM system, and the output was displayed in the form of a table summarizing the information extracted by the system. In the amount of time that it would take a user to quickly read through an article, the PLUM system can process it and present its results. The user can then browse through the specific template objects and make any changes or corrections that are necessary.

The system also serves as an aide for the analyst's correction task. PLUM graphically displays its confidence in particular pieces of information in the output so the analyst can choose to focus on verifying information that PLUM is not highly confident in. Furthermore, by clicking on a piece of information in the template, the analyst can see where in the text the information came from. Figure 6-2 shows the contents of the screen after the user has clicked on the slot fill "a company in Japan": the fragment containing the phrase which gave rise to the filler, "the Japanese group", is highlighted in the text.

When editing template fill values, the analyst can mark and copy text in the article. The system automatically "normalizes" the new value. For example, the analyst might add or correct a company's location by highlighting a string in the text. The system then normalizes the text into the gazetteer-style format required by the domain. The system also displays alternate values that it had considered but rejected for some reason. Figure 6-3 shows the pop-up window which presents the user with the entities that were considered for a slot. The boxes to the left of each entity description are used as "push-buttons": a raised button indicates the entity is not selected for the slot (as in the first entry, "A company in Japan", which the user had corrected). By providing this type of support, the analyst can concentrate more on what information should be in the template, not what the particular format of the template fills needs to be.

6.3 Results in Probabilistic/Learning Algorithms

6.3.1 Morphology

Our POST part-of-speech tagger (Meteer, Schwartz, and Weischedel 1991) uses a stochastic bi-gram model to assign syntactic categories to words in unrestricted text, even if they are unknown. BBN was the first to successfully use several new techniques in part-of-speech analysis:

In addition to its use in several PLUM applications, POST has now been distributed to over ten research and development sites.

In the last two years, it was modified to deal with ambiguous cases of word boundaries. Also, a more optimal version was written, achieving speedup by a factor of 80.

6.3.2 Acquiring Word Association Information

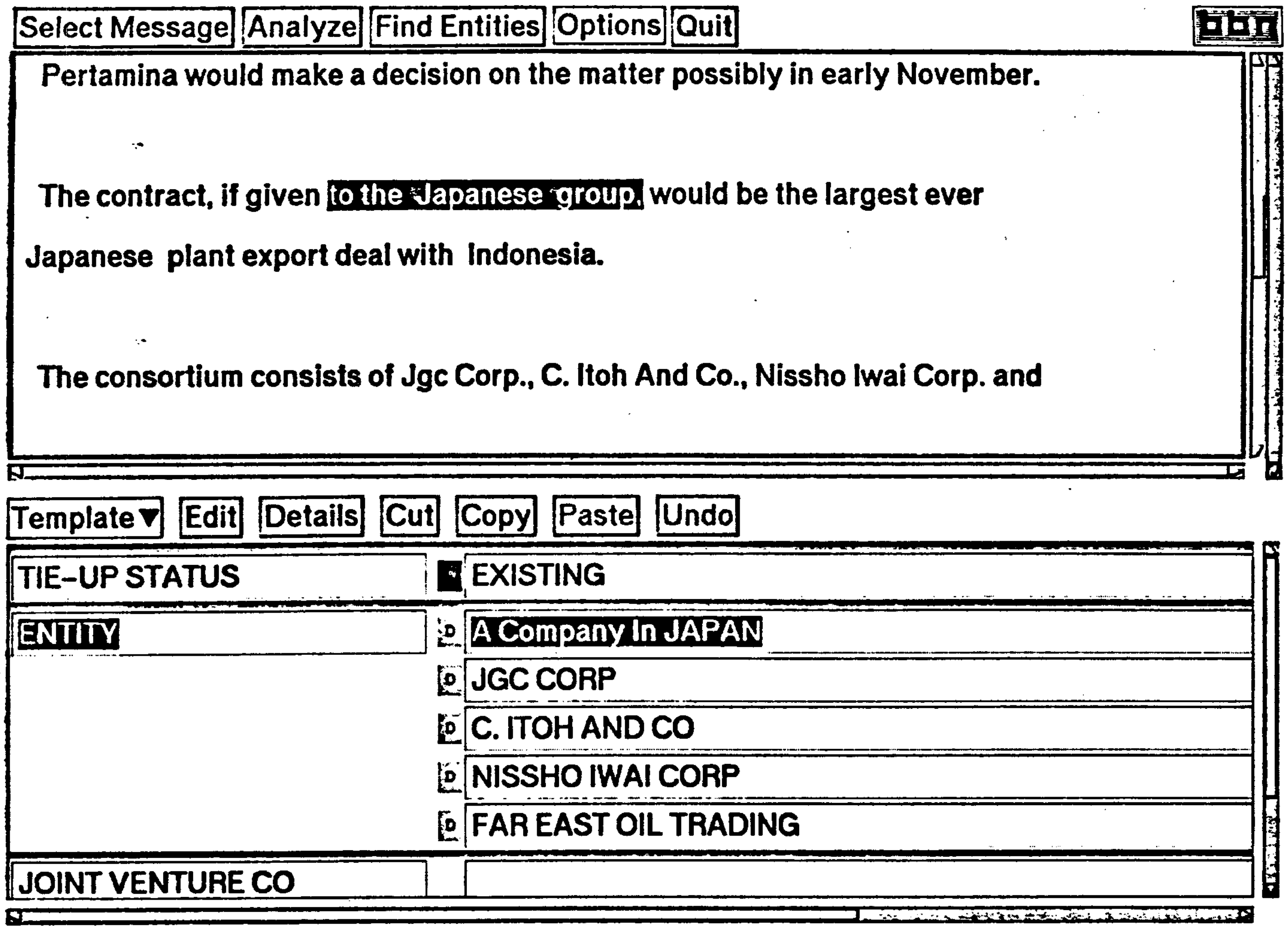
BBN has recently developed a technique for semi-automatically learning word and word group associations [Matsukawa, 1993]. Unlike some other approaches based on mutual information between words, this technique estimates the likely association between *groups* of words, a kind of conceptual clustering. These word groups can either be automatically generated by a hill-climbing algorithm, or produced by manual classification (in the case of nouns, a simple and rapid classification at the rate of 500 words per hour proved to be adequate). By using words grouped into concept classes, even low-frequency data can be considered.

6.3.3 Automatic Error Correction

The traditional approach to word segmentation and part-of-speech labelling in Japanese is the use of rule-based morphological analyzers employing a hand-crafted lexicon and a hand-crafted connectivity matrix. BBN's AMED module [Matsukawa, Miller, and Weischedel, 1993] is an example-based correction technique for segmentation and part-of-speech labelling, which uses annotated text as supervised training. AMED is able to cover cases that occur infrequently by generalizing during training. Using AMED together with POST improved the accuracy of part-of-speech labelling without requiring revision to the existing morphological analyzer (JUMAN).

6.3.4 First Application of Statistical Text Classifier to Data Extraction from Text

For MUC-4, we demonstrated the first technique for significantly varying the trade-off between recall and precision (Weischedel *et al*, 1992). This used an automatically-trained text classifier that predicted whether paragraphs of text were relevant to the data extraction task, based on a probabilistic model using words as features. This gives the user the capability of tuning system performance to favor either recall or precision, by varying the threshold on the classifier.



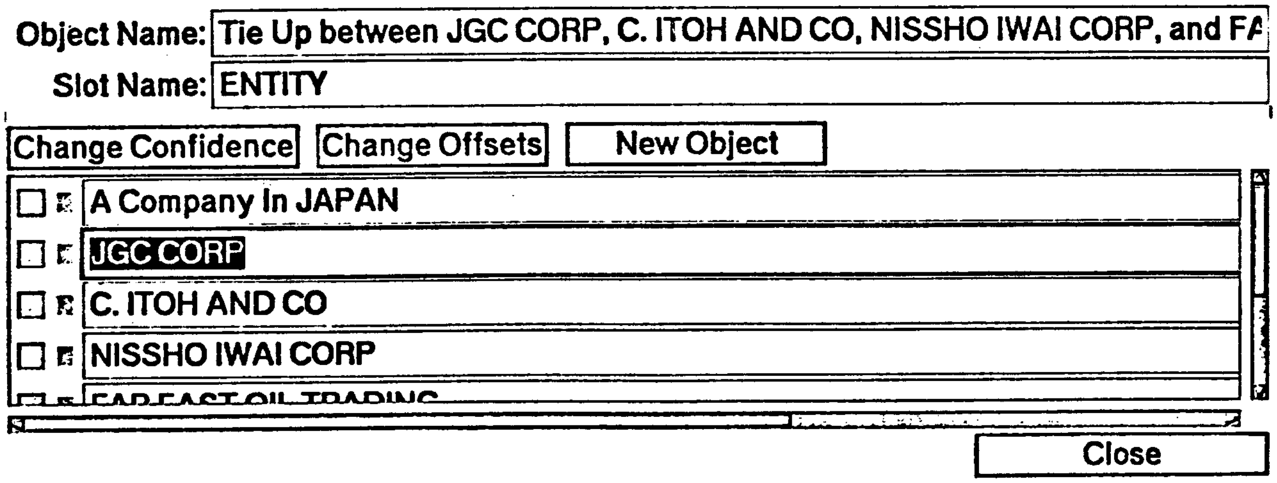
Pertamina would make a decision on the matter possibly in early November.

The contract, if given to the Japanese group, would be the largest ever Japanese plant export deal with Indonesia.

The consortium consists of Jgc Corp., C. Itoh And Co., Nissho Iwai Corp. and

TIE-UP STATUS	EXISTING
ENTITY	A Company In JAPAN
	JGC CORP
	C. ITOH AND CO
	NISSHO IWAI CORP
	FAR EAST OIL TRADING
JOINT VENTURE CO	

Figure 6-2. PLUM highlights the source of the information in the text: *A user can see justification of the output information before editing.*



Object Name: Tie Up between JGC CORP, C. ITOH AND CO, NISSHO IWAI CORP, and FA

Slot Name: ENTITY

- A Company In JAPAN
- JGC CORP
- C. ITOH AND CO
- NISSHO IWAI CORP
- FAR EAST OIL TRADING

Figure 6-3. The entities that have been considered for a slot fill are presented to the user: *A new filler can be quickly selected from this list, or a new one pasted in from the text.*

7. EVALUATION SUMMARY

The official scores achieved by the PLUM system in each of the four application domains are summarized in the table below.

	EME	EJV	JME	JJV
ERR	66	68	70	72
minERR	0.7485	0.7971	0.8671	0.7990
maxERR	0.7822	0.8191	0.9114	0.8196
UND	46	54	53	63
OVG	33	28	41	27
SUB	19	18	15	14
REC	43	38	40	32
PRE	54	59	50	63
F	48.09	45.95	44.61	42.49

Table 7-1. Summary of PLUM scores on TIPS3 data.

Figure 7-1 graphically illustrates that the PLUM system showed remarkable consistency on the official measure ERR across both languages and both domains. Of course, this is meaningless unless one factors in the human effort involved. Figure 7-2 shows the labor invested in the four language-domain pairs: EVJ, JJV, EME, and JME. The effort in each language was largely balanced.

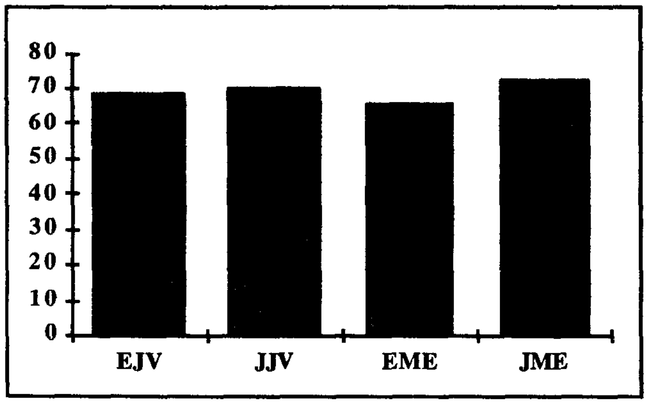


Figure 7-1. Performance Based on ERR: *Across language-domain pairs, there was remarkable consistency in PLUM's performance.*

	Person-Months
English Joint Venture	**4-5**
Japanese Joint Venture	**2**
English Microelectronics	**3**
Japanese Microelectronics	**3.5**

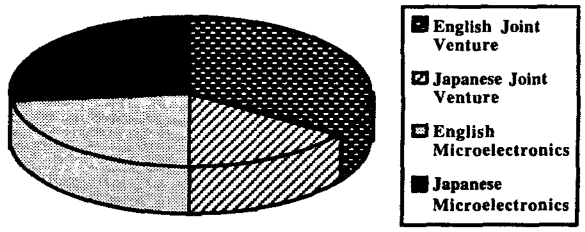


Figure 7-2. Distribution of Effort across Domains: *Effort across languages was about equal.*

The previous evaluation was held eighteen months into the contract and was reported in February, 1993. Much of the effort in this last quarter focused on improving PLUM's performance in extracting data in the microelectronics domain. Figures 7-3 and 7-4 graph our progress in English and in Japanese respectively.

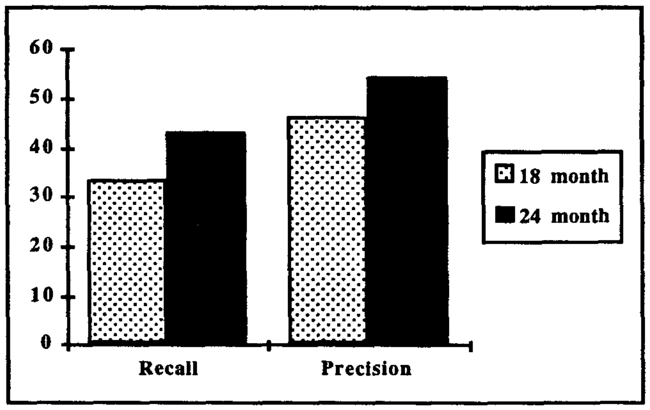


Figure 7-3. Progress in English Microelectronics: *Improvement in the English microelectronics domain from February, 1993 to August, 1993.*

In English, PLUM is extracting about 25% more data than six months ago, and its overall accuracy has improved about 15-20%. Performance in Japanese shows a threefold improvement in the amount of data extracted with about a 10% improvement in accuracy. The improvement in Japanese took that form since our focus was on covering the additional two microelectronics capabilities that had not been attempted in the February evaluation.

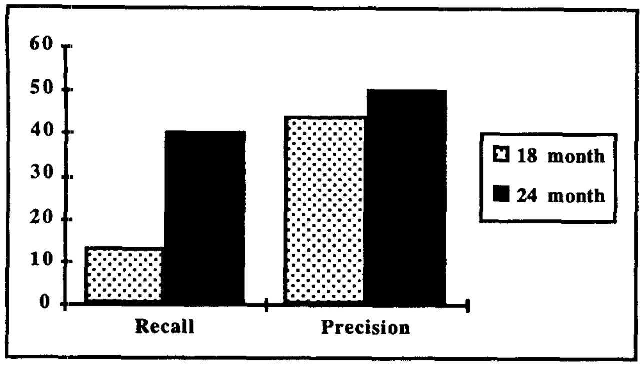


Figure 7-4. Progress in Japanese Microelectronics: *Substantial improvement in the quantity of data extracted (recall) and the quality of data extracted (precision) was achieved from the evaluation at 18 months to 24 months.*

7.1 Discussion of Japanese Performance

Training new staff to use PLUM effectively proved easier than anticipated. Our team faced training new staff two months before the MUC-5 test, as our single Japanese programmer needed to reduce his involvement substantially. Starting at the beginning of June, two Japanese computer science majors, who had just completed their junior year at college came to BBN. They had no training in computational linguistics, but had one course in artificial intelligence and one in LISP. In June, they learned about data extraction, the joint venture and microelectronics tasks, and how to use PLUM. Since the Japanese articles on packaging and lithography had arrived much later than the other data, and since we had not touched that data, they focussed on those two capabilities starting July 1. Initially, of course, PLUM had near 100 as an ERR on sets composed primarily of those microelectronics capabilities.

As evident in Figure 7-5, the progress was rapid and dramatic, as the error rate dropped by 25% in all cases and by almost 50% in some cases.

One consequence in the change of personnel was that performance in both Japanese domains had not peaked. Also, very little effort was devoted to Japanese joint ventures. Many slotw received no effort, accounting for the relatively low recall in JJV.

7.2 Discussion of Metrics

In the last few months before the final evaluation, we noticed an anomaly in the official measure ERR. We noticed that some changes to the runtime version of PLUM could reduce ERR ("error rate") by 2 points (e.g., by 5%), even though the actual number of errors made by the system increased.

The anomaly arises because ERR is computed as

$$\frac{\#\ \text{missed data} + \#\ \text{incorrect answers} + \#\ \text{spurious answers}}{\#\ \text{system answers}}$$

Therefore, it is theoretically possible to generate more questionable output, producing just enough correct output to reduce ERR, even though more than 50% of the additional output is spurious.

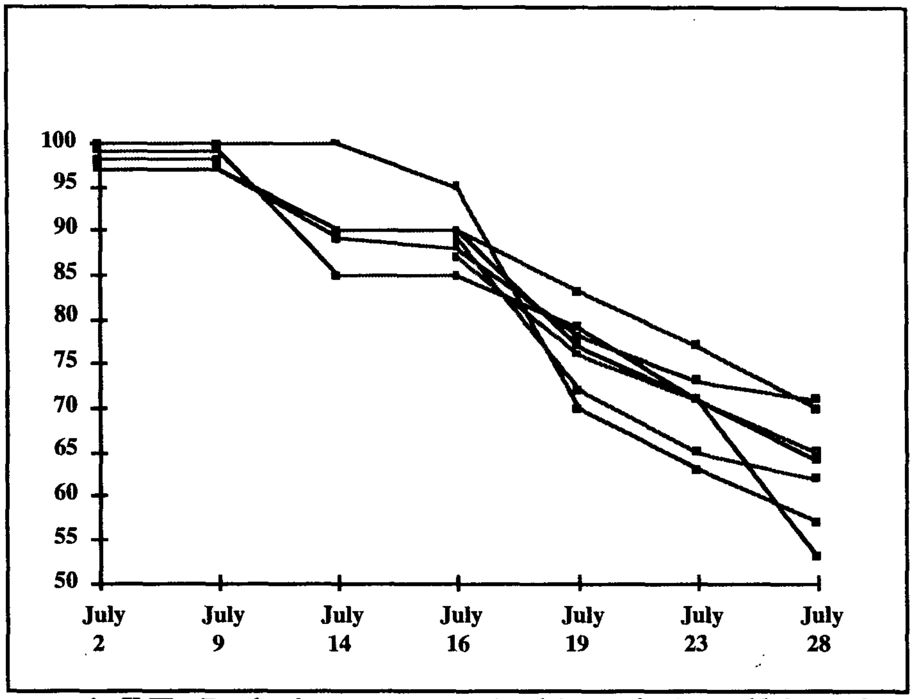


Figure 7-5. Progress in JME: *For development messages involving packaging and lithography, progress of new staff with minimal training was rapid and dramatic.*

As a consequence of the fact that the denominator of ERR is system-dependent, ERR is not ideal for cross-system comparisons. Rather, min-err/max-err are better for cross-system comparison, since they normalize the number of errors by a system-independent measure, the number of answers in the answer key. Min-err/max-err are defined as

$$\frac{\text{\# missed data} + \text{\# incorrect answers} + \text{\# spurious answers}}{\text{\# answers in answer key}}$$

Since some answers are optional, min-err and max-err differ in whether the optional answers are counted in the denominator.

If one uses this unofficial, system-independent measure, systems perform quite differently in at least some cases.

Since we tuned PLUM, trying to jointly optimize both ERR and min-err, PLUM's performance for both English domains was outstanding on both measures.

8. CONCLUSIONS

Some of the lessons we learned during our work in TIPSTER include the following:

- Automatic training and acquisition of knowledge bases can yield relatively good performance at reduced labor, as evidenced, for example, by a quick port to the microelectronics domain (in 2 languages) in 2 person-months (after which further refinements were made). For the TIPS3 test, our total effort spent on each of the 4 domains (in person-months) was as follows: EJV 4.5, EME 3, JJV 2, JME 3.
- Domains dominated by jargon (sub-language) may be easier than domains of normal vocabulary because there is less ambiguity and more predictability. For TIPSTER this means that the microelectronics domain was easier than joint ventures.
- Japanese was easier to process than English because of strong clues provided by case-markers, and a less varied linguistic structure in the articles.
- Availability of a large text corpus was invaluable for quick knowledge acquisition. Less filled templates should still be adequate.
- Our algorithms are already largely language- and domain-independent; an important goal remains to further automate the porting process.
- Finite-state pattern matching is a useful complement to linguistic processing, offering a good fall-back strategy for addressing language constructions that are hard to treat via general linguistically-based approaches.
- Continued work on discourse processing is important to improving performance. Reliably determining when different descriptions of events or objects in fact refer to the same thing remains one of the hardest problems in data extraction.
- Improving syntactic coverage is a priority. Increased coverage normally leads to greater perceived ambiguity in the system; we hope to counter this through the use of probabilistic models.

We plan to continue our research agenda emphasizing the use of probabilistic modeling and learning algorithms for data extraction in order to continue improving robustness and portability.

ACKNOWLEDGMENTS

The work reported here was supported in part by the Defense Advanced Research Projects Agency and was monitored by the Rome Air Development Center under Contract No. F30602-91-C-0051. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.

REFERENCES

[1] Ayuso, D.M., Boisen, S., Fox, H., Ingria, R., and Weischedel, R. "BBN: Description of the PLUM System as Used for MUC-4", *MUC-4 Proceedings,* 1992.

[2] Iwanska, et.al., "Computational Aspects of Discourse in the Context of MUC-3", *Proceedings of the Third Message Understanding Conference (MUC-3),* 1991.

[3] Matsukawa, T., Miller, S., and Weischedel, R. "Example-Based Correction of Word Segmentation and Part of Speech Labelling", to appear in *Proceedings of the ARPA Workshop on Human Language Technology*, 1993.

[4] Weischedel, R., Ayuso, D.M., Bobrow, R., Boisen, S., Ingria, R., and Palmucci, J., "Partial Parsing, A Report on Work in Progress", *Proceedings of the Fourth ARPA Workshop on Speech and Natural Language,* 1991.

[5] Weischedel, R., Meteer, M., Schwartz, R., Ramshaw, L., and Palmucci, J. "Coping with Ambiguity and Unknown Words through Probabilistic Models", *Computational Linguistics (Special Issue on Using Large Corpora: II)* 19, 359-382, 1993.

[6] Matsukawa, T., Hypothesizing Word Association from Untagged Text", to appear in *Proceedings of the ARPA Workshop on Human Lnaguage Technology*, 1993.