GE: DESCRIPTION OF THE NLTOOLSET SYSTEM AS USED FOR MUC-3

George Krupka, Paul Jacobs, Lisa Rau, and Lucja Iwańska
Artificial Intelligence Laboratory
GE Research and Development
Schenectady, NY 12301 USA
krupka@crd.ge.com

Abstract

The GE NLTOOLSET *is a set of text interpretation tools designed to be easily adapted to new domains. This report summarizes the system and its performance on the MUC-3 task.*

INTRODUCTION

The GE NLTOOLSET aims at extracting and deriving useful information from text using a knowledge-based, domain-independent core of text processing tools, and customizing the existing programs to each new task. The program achieves this transportability by using a core knowledge base and lexicon that adapts easily to new applications, along with a flexible text processing strategy that is tolerant of gaps in the program's knowledge base.

The NLTOOLSET's design provides each system component with access to a rich hand-coded knowledge base, but each component applies the knowledge selectively, avoiding the computation that a complete analysis of each text would require. The architecture of the system allows for levels of language analysis, from rough skimming to in-depth conceptual interpretation.

The NLTOOLSET, in its first version, was behind GE's participation in the MUCK-II conference. Since MUCK-II, the Toolset, now in Release 2.1, has expanded to include a number of new capabilities, including a text pre-processor for easier customization and better performance, broader lexical and syntactic coverage, and a domain-independent module for applying word-sense preferences in text. In addition to being tested in several new application areas, the Toolset has achieved about a 10 times speedup in words per minutes over MUCK-II, and can now partially interpret and tag word senses in arbitrary news stories, although it is very difficult to evaluate this task-independent performance. These basic enhancements preceded the other additions, including a discourse processing module, which were made for MUC-3.

The performance of the program on tasks such as MUCK-II and MUC-3 derives mainly from two design characteristics: central knowledge hierarchies and flexible control strategies. A custom-built 10,000 word-root lexicon and 1000-concept hierarchy provides a rich source of lexical information. Entries are separated by their senses, and contain special context clues to help in the sense-disambiguation process. A morphological analyzer contains semantics for about 75 affixes, and can automatically derive the meanings of inflected entries not separately represented in the lexicon. Domain-specific words and phrases are added to the lexicon by connecting them to higher-level concepts and categories present in the system's core lexicon and concept hierarchy. Lexical analysis can also be restricted or biased according to the features of a domain. This is one aspect of the NLTOOLSET that makes it highly portable from one domain to another.

The language analysis strategy in the NLTOOLSET uses fairly detailed, chart-style syntactic parsing guided by conceptual expectations. Domain-driven conceptual structures provide feedback in parsing, contribute to scoring alternative interpretations, help recovery from failed parses, and tie together information across sentence boundaries. The interaction between linguistic and conceptual knowledge sources at the level of linguistic relations, called "relation-driven control" was a key system enhancement before MUC-3.

In addition to flexible control, the design of the NLTOOLSET allows each knowledge source to influence different stages of processing. For example, discourse processing starts before parsing, although many decisions about template merging and splitting are made after parsing. This allows context to guide language analysis, while language analysis still determines context.

The next section briefly describes the major portions of the NLToolset and its control flow; the remainder of the paper will discuss the application of the Toolset to the MUC-3 task.

SYSTEM OVERVIEW

Processing in the NLToolset divides roughly into three stages: (1) *pre-processing*, consisting mainly of a pattern matcher and discourse processing module, (2) *linguistic analysis*, including parsing and semantic interpretation, and (3) *post-processing*, or template filling. Each stage of analysis applies a combination of linguistic, conceptual, and domain knowledge, as shown in Figure 1.

	Pre-processing	***Analysis***	***Post-processing***
Syntax	Tagging, bracketing	Parsing, attachment	Scoring
Semantics	Collocations, cluster analysis	Sense discrimination, role mapping	Role extension
Domain Templates	Template activation, text segmentation	Ambiguity pruning, recovery	Default filling, merging, splitting, etc.

Figure 1: Stages of data extraction

The pre-processor uses lexico-semantic patterns to perform some initial segmentation of the text, identifying phrases that are template activators, filtering out irrelevant text, combining and collapsing some linguistic constructs, and marking portions of text that could describe discrete events. This component is described in [1]. Linguistic analysis combines parsing and word sense-based semantic interpretation with domain-driven conceptual processing. The programs for linguistic analysis are largely those explained in [2, 3]—the changes made for MUC-3 involved mainly some additional mechanisms for recovering from failed processing and heavy pruning of spurious parses. Post-processing includes the final selection of templates and mapping semantic categories and roles onto those templates. This component used the basic elements from MUCK-II, adding a number of specialized rules for handling guerrilla warfare, types, and refines the discourse structures to perform the template splitting and merging required for MUC-3.

The control flow of the system is primarily from linguistic analysis to conceptual interpretation to domain interpretation, but there is substantial feedback from conceptual and domain interpretation to linguistic analysis. The MUC-3 version of the Toolset includes our first implementation of a strategy called *relation-driven control*, which helps to mediate between the various knowledge sources involved in interpretation. Basically, relation-driven control gives each linguistic relation in the text (such as subject-verb, verb-complement, or verb-adjunct) a preference score based on its interpretation in context. Because these relations can apply to a great many different surface structures, relation-driven control provides a means of combining preferences without the tremendous combinatorics of scoring many complete parses. Effectively, relation-driven control permits a "beam" strategy for considering multiple interpretations without producing hundreds or thousands of new paths through the linguistic chart.

The knowledge base of the system, consisting of a feature and function (unification-style) grammar with associated linguistic relations, and the lexicon mentioned earlier, still proves transportable and largely generic. The core lexicon contains over 10,000 entries, of which 37 had to be restricted because of specialized usage in the MUC-3 domain (such as *device*, which always means a bomb, and *plant*, which as a verb usually means to place a bomb and as a noun usually means the target of an attack). The core grammar contains about 170 rules, with 50 relations and 80 additional subcategories. There were 23 MUC-specific additions to this grammatical knowledge base, including 8 grammar rules, most of them dealing with unusual noun phrases that describe organizations in the corpus.

The control, pre-processing, and transportable knowledge base were all extremely successful for MUC-3; remarkably, lexical and grammatical coverage, along with the associated problems in controlling search and selecting among interpretations, proved not to be the major stumbling blocks for our system—further distinguishing events and merging or splitting templates proved to be the major obstacle in obtaining a better score.

ANALYSIS OF TST1-0099

The common "walkthrough" example, TST1-0099, is a good example of many of the problems in analysis and template filling, although it is somewhat unrepresentative of the difficulties in parsing because the key content is contained in fairly simple sentences. We will explain briefly what our program did, then provide details of the story-level and sentence-level interpretation with an analysis of the templates produced.

Overview of Example

In many ways, TST1-0099 is representative of how the Toolset performed on MUC-3. The program parsed most of the key sentences, failed to parse some of the less relevant sentences, missed a key relationship between locations—thus failing to split a template into two separate events—and incorrectly included an earlier bombing as part of a main event in the story. One additional program fill was scored incorrect because the answer key had the wrong date. The program thus derived 36 slots out of a possible 43, with 21 correct, 2 partial, 2 incorrect, and 11 spurious, for 51% recall, 61% precision, and 30% overgeneration.

Detail of Message Run

As explained in the previous section, the Toolset uses pattern matching for pre-processing, followed by discourse processing, parsing and semantic interpretation, and finally template-filling.

Pre-processing uses pattern matching to manipulate the input text. It recognizes relevant fillers, tags, collapses constructs, and segments the text into fragments describing different events. For example, the pattern matcher recognizes the phrase describing the bombing event in the first sentence of the text, collapses the conjunctive phrase *the embassies of the PRC and the Soviet Union*, and marks *that* as a complementizer (rather than a relative pronoun, pronoun, or determiner). In later sentences, it also marks locative phrases like *in the Lima residential district of San Isidro* and *located in Orrantia district.* The discourse processing module does an initial text segmentation based on (1) definite and indefinite references like *a car bomb* and *the attack*, (2) the relationship between events (e.g. *bombing* and *arson*), and (3) cue phrases. This identifies six events:

(1) the phrase *bombed* describes a bombing event; the phrase *the bombs* marks its continuation;

(2) the phrase *a car bomb exploded* signifies a new bombing event;

(3) the temporal cue phrase *meanwhile* combined with the phrase *two bombs* in indefinite form signifies another bombing event; the phrases *the bombs* and *the attacks* mark the continuation of this event;

(4) the phrase *a Shining Path bombing* indicates yet another bombing event; the sentence gets deleted because the temporal information in the phrase *some three years ago* violates MUC-3 constraints;

(5) the cue phrase *in another incident* combined with the phrases *killed* and *dynamite* delineates another bombing event; the sentence gets deleted because the temporal information from the phrase *three years ago* violates MUC-3 constraints;

(6) the phrase *burned* indicates a new arson event.

Linguistic analysis parses each sentence and produces (possibly alternative) semantic interpretations at the sentence level. These interpretations select word senses and roles, heavily favoring domain-specific senses.

Post-processing maps the semantic interpretations onto templates, eliminating invalid fills (in this case none), combining certain multiple references (such as to the embassies), and adding certain information (like adding numbers and "guessing" terrorist groups as fillers when it fails to find evidence to the contrary). Post-processing collapses three of the segments (events) produced during pre-processing into one template.

Sentence-level Interpretation

The following is the trace of the first two sentences of TST1-0099: The "call" to *Trumpet* represents the end of the first stage of semantic interpretation and the beginning of conceptual analysis, and the output that follows this represents the mapping (or role extension) from semantic roles to templates.

```
;;; Trigger rules filtered 1 tokens (3.45%)
1 *REDUCE1-CONTEXT* Bracketing:
[BRACKET: [TEXT] ]POLICE HAVE REPORTED THAT TERRORISTS [DATE: TONIGHT ]BOMBED THE
EMBASSIES OF THE PRC AND THE SOVIET UNION.

Creating temporal pp for "TONIGHT "
1 *REDUCE2-CONTEXT* Bracketing:
POLICE [AUX: HAVE REPORTED ][CMPLZR: THAT] TERRORISTS TONIGHT BOMBED [EMBASSY: THE
EMBASSIES OF THE PRC AND THE SOVIET UNION].

1 *LIST-CONTEXT* Bracketing:
POLICE HAVE REPORTED THAT TERRORISTS TONIGHT BOMBED [GLIST: THE EMBASSIES PRC
AND EMBASSIES THE SOVIET UNION].

TOKEN DANGER: Creating list NP from "THE EMBASSIES PRC AND EMBASSIES THE SOVIET UNION"

Calling Trumpet with Sexp:
(VERB_REPORT1
  (R-PATIENT
    (VERB_BOMB1
      (R-PATIENT
        (COORDCONJ_AND1
          (R-PART1 (NOUN_EMBASSY1 (R-PART (C-ENTITY))
                                  (R-NATIONALITY (NATION-NAME_CHINA1 (R-NAME CHINA)))
                                  (R-DEFINITE (DET_THE1))))
          (R-PART1 (NOUN_EMBASSY1 (R-PART (C-ENTITY))
                                  (R-NATIONALITY (NATION-NAME_SOVIET-UNION1 (R-NAME SOVIET-UNION))))))
      (R-DATE
        (C-DATE-OF-OCCURRENCE (R-RELATIVE NO) (R-YEAR NIL) (R-DAY |25|) (R-MONTH OCT)))
      (R-AGENT (C-VERB_TERRORIZE1-ER))
      (R-INSTRUMENT (C-BOMB))))
  (R-COMMUNICATOR (NOUN_POLICE1)))

TRUMPET WARN: Breaking out core templates (C-BOMBING-TEMPLATE)

Processing core template:
(C-BOMBING-TEMPLATE (R-INSTRUMENT (C-BOMB))
   (R-DATE (C-DATE-OF-OCCURRENCE (R-RELATIVE NO)
                                 (R-YEAR NIL)
                                 (R-DAY |25|)
                                 (R-MONTH OCT)))
   (R-TARGET
        (COORDCONJ_AND1
          (R-PART1
            (NOUN_EMBASSY1 (R-PART (C-ENTITY))
                           (R-NATIONALITY (NATION-NAME_CHINA1 (R-NAME CHINA)))
                           (R-DEFINITE (DET_THE1))))
          (R-PART1
            (NOUN_EMBASSY1 (R-PART (C-ENTITY))
                           (R-NATIONALITY
                             (NATION-NAME_SOVIET-UNION1 (R-NAME SOVIET-UNION))))))
   (R-PERPETRATOR (C-VERB_TERRORIZE1-ER)))
```

```
Calling Trumpet with Sexp:
(C-CAUSING (R-CAUSE (C-BOMB (R-DEFINITE (DET_THE1))))
           (R-EFFECT
             (COORDCONJ_BUT1 (R-PART1 (NOUN_DAMAGE1))
                             (R-PART2 (C-INJURY (R-POLARITY (DET_NO1)))))))

TRUMPET WARN: Breaking out core templates
(C-INJURY-TEMPLATE C-DAMAGE-TEMPLATE)

Assuming C-BOMBING-TEMPLATE and C-INJURY-TEMPLATE have same R-INSTRUMENT
Assuming C-BOMBING-TEMPLATE and C-DAMAGE-TEMPLATE have same R-INSTRUMENT
Assuming C-BOMBING-TEMPLATE and C-INJURY-TEMPLATE have same R-DATE
Assuming C-BOMBING-TEMPLATE and C-DAMAGE-TEMPLATE have same R-DATE
Assuming C-BOMBING-TEMPLATE and C-INJURY-TEMPLATE have same R-PERPETRATOR
Assuming C-BOMBING-TEMPLATE and C-DAMAGE-TEMPLATE have same R-PERPETRATOR
Assuming C-BOMBING-TEMPLATE and C-INJURY-TEMPLATE have same R-TARGET
Assuming C-BOMBING-TEMPLATE and C-DAMAGE-TEMPLATE have same R-TARGET
```

Comparison of Program Answers with Answer Key

The NLToolset results for TST1-0099 were the following templates:

```
0. MESSAGE-ID      TST1-MUC3-0099
1. TEMPLATE-ID     1
2. INCIDT-DATE     25 OCT 89
3. INCIDT-TYPE     BOMBING
4. CATEGORY        TERRORIST ACT
5. INDIV-PERPS     "TERRORISTS"
6. ORG-PERPS       "MAOIST \"SHINING PATH\" GROUP"
                   "GUEVARIST \"TUPAC AMARU REVOLUTIONARY MOVEMENT\""
7. PERP-CONF       POSSIBLE: "MAOIST \"SHINING PATH\" GROUP"
                   POSSIBLE: "GUEVARIST \"TUPAC AMARU REVOLUTIONARY MOVEMENT\""
8. PHYS-TGT-ID     "VEHICLES"
                   "EMBASSIES OF THE PRC"
                   "EMBASSIES OF THE PRC AND THE SOVIET UNION"
9. PHYS-TGT-NUM    4
10. PH-TGT-TYPE    TRANSPORT VEHICLE: "VEHICLES"
                   DIPLOMAT OFFICE OR RESIDENCE: "EMBASSIES OF THE PRC"
                   DIPLOMAT OFFICE OR RESIDENCE: "EMBASSIES OF THE PRC AND THE SOVIET UNION"
11. HUM-TGT-ID     "SOVIET MARINES"
12. HUM-TGT-NUM    15
13. HM-TGT-TYPE    ACTIVE MILITARY: "SOVIET MARINES"
14. TARGET-NAT     PEOPLES REP OF CHINA: "EMBASSIES OF THE PRC"
                   USSR: "EMBASSIES OF THE PRC AND THE SOVIET UNION"
                   USSR: "SOVIET MARINES"
15. INST-TYPE      *
16. LOCATION       PERU: SAN ISIDRO (NEIGHBORHOOD): ORRANTIA (DISTRICT)
17. PHYS-EFFECT    SOME DAMAGE: "VEHICLES"
                   SOME DAMAGE: "EMBASSIES OF THE PRC AND THE SOVIET UNION"
                   SOME DAMAGE: "EMBASSIES OF THE PRC"
18. HUM-EFFECT     NO INJURY: "SOVIET MARINES"
```

```
0. MESSAGE-ID      TST1-MUC3-0099
1. TEMPLATE-ID     2
2. INCIDT-DATE     24 OCT 89
3. INCIDT-TYPE     ARSON
4. CATEGORY        TERRORIST ACT
5. INDIV-PERPS     -
6. ORG-PERPS       "SHINING PATH"
7. PERP-CONF       REPORTED AS FACT: "SHINING PATH"
8. PHYS-TGT-ID     "BUSES"
9. PHYS-TGT-NUM    10
10. PH-TGT-TYPE    TRANSPORT VEHICLE: "BUSES"
11. HUM-TGT-ID     -
12. HUM-TGT-NUM    -
13. HM-TGT-TYPE    -
14. TARGET-NAT     -
15. INST-TYPE      *
16. LOCATION       PERU: LIMA (CITY)
17. PHYS-EFFECT    SOME DAMAGE: "BUSES"
18. HUM-EFFECT     -
```

The failure to recognize that San Isidro and Orrantia are distinct locations caused the program to combine the two bombings into one template (under the mistaken assumption that Orrantia is in San Isidro). We do not know now why the program did not fill in the City name "Lima", although this would not have affected the score. As a result of the location assumption, the Toolset got two extra fills for slots 8 and 10 in the first template (effectively by merging the templates), missed slot 9 entirely (because the number of targets is different), and got one extra fill in slot 14, in addition to partial credit for the location.

The program correctly discarded the bombing of the bus in 1989 but failed to group the 15 wounded Soviet marines correctly with that event (because of a simple bug which caused the deletion of the earlier event before the wounding effect was processed), thus losing points also in Template 1, slots 11, 12, 13, 14, and 18 (even though slot 18 was correct except for the cross-reference).

The program got the correct date from *tonight*, October 25. The answer key has a range, October 24-25.

The second template produced by the Toolset was completely correct, but the score for this message is 51% precision and 61% recall, mainly due to the combining of two possible templates into one.

SUMMARY AND CONCLUSION

MUC-3 is a very difficult task, involving a combination of language interpretation, conceptual and domain knowledge, along with many rules and strategies for template filling. The examples given here show not only how our system performs this, but hopefully some of the limitations of the system and the penalties paid in the scoring for these mistakes. While it is very difficult to attribute effects in the score to particular functions of the programs, there is no question that the task adequately exercises most of the current features of our system. It is equally clear that there is ample room for improvements from promising research areas, such as implicit event reference, discourse processing and representation, and general reference, as well as from task-specific processing and more well-known problems such as general inference.

References

[1] Paul S. Jacobs, George R. Krupka, and Lisa F. Rau. Lexico-semantic pattern matching as a companion to parsing in text understanding. In *Fourth DARPA Speech and Natural Language Workshop*, San Mateo, CA, February 1991. Morgan-Kaufmann.

[2] Paul Jacobs and Lisa Rau. SCISOR: Extracting information from on-line news. *Communications of the Association for Computing Machinery*, 33(11):88–97, November 1990.

[3] P. Jacobs. TRUMP: A transportable language understanding program. *International Journal of Intelligent Systems*, 6(4), 1991.