University of Manitoba: Description of the NUBA System as Used for MUC-5

Dekang Lin
Department of Computing Science
University of Manitoba
Winnipeg, Manitoba, Canada R3T 2N2
lindek@cs.umanitoba.ca

INTRODUCTION

Abduction is the inference to the best explanation. Many tasks in natural language understanding such as word-sense disambiguity [1], local pragmatics [4], metaphor interpretation [3], and plan recognition [5, 8], can be viewed as abduction.

NUBA (Natural-language Understanding By Abduction) is a natural language understanding system, where syntactic, semantic, and discourse analysis are performed by abductive inference. The task of understanding is viewed as finding the structural relationships between unstructured inputs. That is, to understand is to seek the best explanation of how the inputs are coherently related. From this abductive perspective, the goal of the parser is to explain how the words in a sentence are structured according to syntactic relationships; the goal of the semantic interpreter is to find the semantic relationships among the content words in a sentence; the goal discourse analyzer is to show how the events mentioned in the sentences fit together to form a coherent plan.

Although the abductive formulation of natural language understanding tasks results in significant simplifications [4], the computational complexity of abductive inference presents a serious problem. Our solution to this problem is obvious abduction [6], a model of abductive inference that covers the kinds of abductive inferences people perform without apparent effort, such as parsing, plan recognition, and diagnosis. Obvious abduction uses a network to represent the domain knowledge. Observations correspond to nodes in the network annotated with a set of attribute-value pairs. An explanation of the observations is a generalized subtree of the network that connects all the observations. This connection is a coherent set of relationships between the observations, therefore, explains how they are related.

SUMMARY OF TEST RESULTS

Considering that NUBA is a new participant in MUC and that several important modules of NUBA have not yet been implemented, NUBA's test results are very good. Table 1 shows a summary of the scores.

Table 1: MUC-5 Formal Testing Scores

SLOT	REC	PRE	UND	OVG	ERR	UND	OVG	SUB
ALL OBJECTS	41	50	49	38	69	49	38	19
MATCHED ONLY	61	75	29	13	45	29	13	14
TEXT FILTERING	87	84	13	16	-	-	-	-
F-MEASURES	P&R=44.96			2P&R=48.01		P&2R=42.28		

NUBA achieved this performance with speed. It took NUBA 16 minutes (elapse time) to process the 300 formal testing articles on a SPARCbook[1] with 16M memory. About a quarter of the time was spent on memory swapping due to some known memory leaks.

[1] The same 25-MHz CPU as SPARCstation SLC

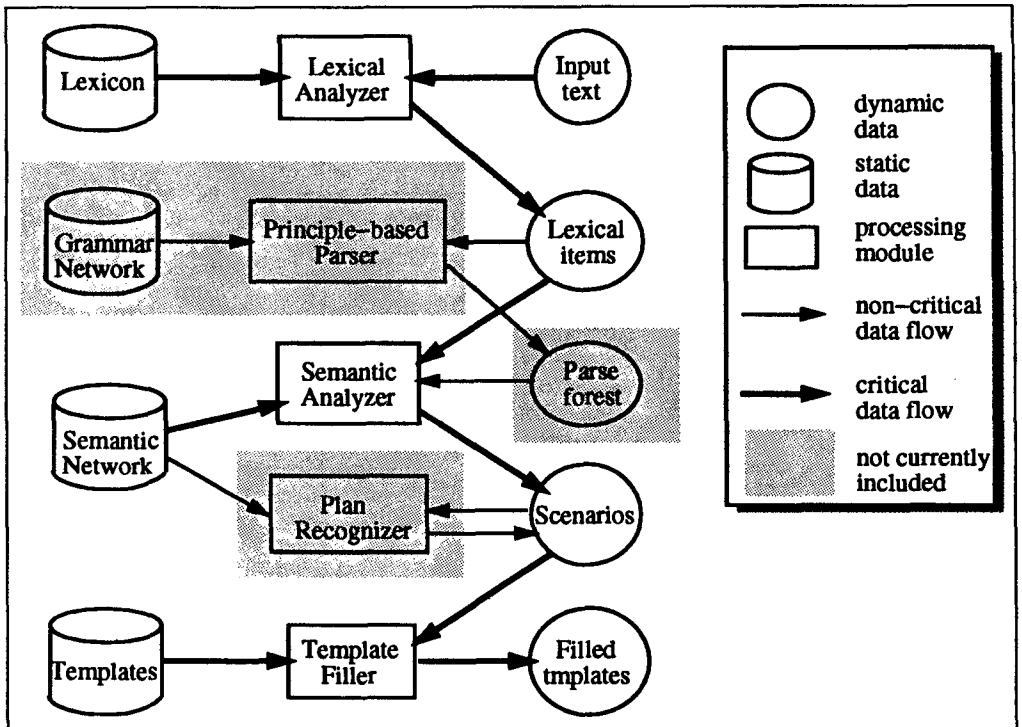

Figure 1: The System Architecture

SYSTEM ARCHITECTURE

Figure 1 shows the NUBA system architecture. NUBA has the following innovative aspects, compared with the generic information extraction system [2].

- The lexical analyzer creates a set, instead of a sequence, of lexical items. This means that the surface strings of the lexical items may overlap.
- The semantic analyzer takes as inputs, the set of lexical items and a shared packed parse forest, rather than a parse tree or fragments of a parse tree of the input sentence. Further more, the parse forest is optional, i.e., the semantic analyzer is able to proceed without syntactic analysis. In fact, our official MUC-5 system does not include a parser.
- Unlike many other systems, semantic interpretation in NUBA is neither rule-based nor pattern-based.
- In the generic system, the algorithms in parsers, semantic interpreters and discourse analyzers have little, if anything at all, in common. In NUBA, they share the same message passing algorithm for obvious abduction. They differ only in the contents of messages and the constraints on message combination and propagation.

LEXICAL ANALYSIS

The lexical analyzer recognizes the sentence boundaries and creates a set of lexical items for each sentence. A lexical item is a pair:

`<surface-string, attribute-vector>`, where

`surface-string` is an interval [i,j] denoting the i'th to j'th word in the sentence;

attribute-vector is a list of attribute-value pairs. The attributes may either be syntactic, e.g., **+plu**, **(per 3)**., or semantic, e.g., **(film silicon)**, **(layering CVD)**.

In case of lexical ambiguity, multiple items are created for the same word or phrase. The surface strings of different items may even overlap.

A LEX program is used for message zoning and sentence boundary recognition. For each sentence, the lexical analyzer then

1. maps the words and phrases in a sentence into a set of lexical items by looking up a lexicon.
2. applies a set of lexical rules to the lexical items.

These two steps are discussed in the next two subsections.

LEXICON

The purpose of lexicons is to map words into their semantic and/or syntactic representations. NUBA's lexicon consists of two files: one holds the entries, the other contains a hash index into the first file. None of these files are loaded into the memory. When changes are made to the lexicon, the hash index file has to be rebuilt. In our experiment, where the lexicon contains 90K entries, the average time to retrieve an entry is 0.002 second.

Lexical entries are written in LISP-like format. It contains a key, which may consist of more than one word, and a list of functions. The functions return either a meaning of the key or a list of phrases for which the key is the head word.

The format of an lexical entry is as follows:

```
(<key words>
  (func <arguments>)
  ... ...
  )
```

Two example entries are shown below:

```
(aluminum
 (meaning MucMeaning ((mucnode "bonding") (bonding ALUMINUM)))
 (meaning MucMeaning ((mucnode "film") (film ALUMINUM)))
 (phrases
   (Aluminum Co of America)
   (aluminum copper)
   (aluminum silicon)
 )
)

(epitaxy, chemical beam
 (meaning MucMeaning ((muctype layering) +equip +layer (mucnode "equipment")
                      (equipment EPITAXIAL_SYSTEM)))
)
```

The function **meaning** returns a **Meaning** object. The first argument is the class of the meaning object, which is a subclass of **Meaning**. The arguments following the class are passed to the initializer for the class. The first entry means that the word **aluminum** can be a type of bonding, or a type of film, or the head word in phrases **Aluminum Co of America**, **aluminum copper**, and **aluminum silicon**.

The second entry is a phrasal entry (**chemical beam epitaxy**). The word **epitaxy** is said to be the head of the phrase. Generally speaking, the head word of a phrase should be the least frequent word in the phrase or one that may undergo morphological changes. When the head word is found to be present in a sentence, the lexicon then check its neighboring words in the sentence to see whether the phrase is present or not. If it is, one or more lexical items for the phrase are created.

LEXICAL RULES

Once the lexical items have been obtained by looking upt the lexicon, a set of lexical rules is applied to them. Each of the following task is performaed by a lexical rule:

Corporate name recognition: An entity name will be recognized if it appears in the lexicon. Otherwise, if it is followed by a corporate designator, it may be recognized by this rule. When a lexical item has `+corpdesig` but the `name` attribute is undefined, the word is a corporate designator. The rule then searches for the sequence of capitalized words before this word and interprets the sequence as the name of an entity.

Irrelevant sentence filtering: When a non-IC word or phrase, such as "printed circuit board," is found in a sentence, the sentence is assumed to be irrelevant and all the lexical items are removed.

Negation handling: Since the current implementation does not include a parser, the scope of negation operator is simply assumed to be from its position to the end of the sentence. When a negation word (e.g., `not`, `no`, `except`) is encountered, all the lexical items following the word are removed.

City name recognition: The lexicon contains all the country names and the provinces for several countries that are most frequently mentioned in the training corpus. When a country or province name is preceded by a sequence of capitalized unknown words, the sequence is assumed to be a city name.

Determination of location of entities: Locations of entities are determined if they appear either before or after the entity in the text.

PRINCIPLE-BASED PARSING

In [7], the author presented an efficient, principle-based parser. The parser encodes the Government and Binding (GB) theory in a network. The nodes in the network represent grammatical categories, such as NP, VP, etc. The links in the network represent dominance relationships between the categories. The GB principles are represented as constraints attached to nodes and links in the network. The lexical items are mapped into nodes in the network, annotated with attribute values. The algorithm for obvious abduction is used to find connections between the words in the network, which are consistent with principles in the GB Theory. The connections explain how the words in the sentence relate to one another in terms of syntactic relationships and can serve as the parse trees of the sentence.

The parse has been implemented and preliminarily tested. However, due to the shortage of time and people, we were not able to integrate the parser with the rest of the system before MUC-5 formal testing.

SYNTAX-CONSTRAINED SEMANTIC INTERPRETATION

In NUBA, the domain knowledge is represented by a semantic network (Figure 2). Semantic interpretation is viewed as the process of finding the best explanation of how the content words in the sentence are related to one another in terms of semantic relationships in the network.

A lexical item corresponds to a node in the semantic network, annotated with a set of attribute values. The goal of semantic interpretation is to find a generalized subtree of the network that connects the lexical items. A tree is a generalized subtree of the network if the nodes in the tree are labeled with the nodes in the network and every directed path in the tree is also a directed path in the network. We call the connection a **scenario**. Such a scenario explains the lexical items because a description of the scenario may mention the surface strings in the lexical items. The best explanation is one that explains the largest number of lexical items with the minimum number of links.

The algorithm for finding the best explanation is a message passing algorithm. A message is a pair that represents an explanation of a subset of the lexical items:

$([b, e], av)$

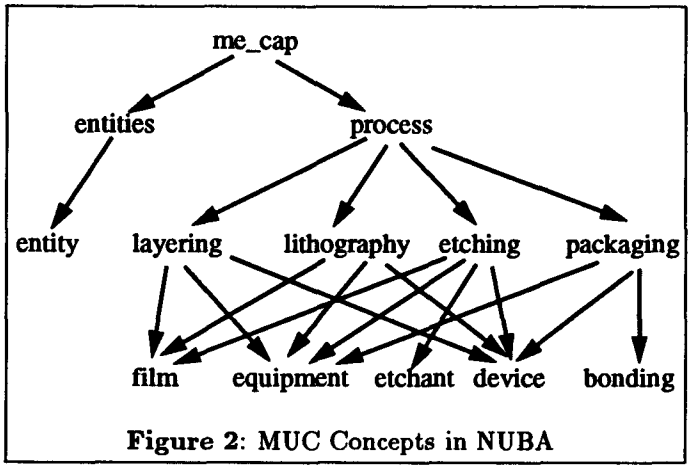

Figure 2: MUC Concepts in NUBA

where $[b, e]$ is an integer interval representing the span of the lexical items it explains and av is an attribute value vector representing the properties of the explanation. Upon receiving a message, a node attempts to combine it with other messages already received by the node to form a new message. That is, the node combines several smaller explanations into a larger one. Two messages $m_1 = ([b_1, e_1], av_1)$ and $m_2 = ([b_2, e_2], av_2)$ can only be combined if their spans do not overlap, i.e., $e_1 < b_2$, and their attribute values are unifiable. The result of the unification is $m = ([b_1, e_2], \text{unify}(av_1, av_2))$.

Each node has a **completion predicate**. If the attribute values of a message satisfy the completion predicate, the message is sent further to other messages. Otherwise, the message waits to be combined with other messages at the node.

Filters can be attached to the links in the semantic network. A filter is an attribute value vector. A message can only pass through the filter if its attribute values are unifiable with those of the filter. For example, the link from **lithography** to **equipment** has a filter (**muctype lithography**). This means that if an equipment has a **muctype** attribute but its value is not **lithography**, then the equipment cannot be involved in a lithography process.

When the message passing process stops, we can find the best explanation by tracing the origins of the messages that explain the largest number of lexical items. The number of lexical items that are explained by a message is the value of the **count** attribute in the message.

Semantic disambiguity is achieved as a side effect of the search for the best explanation. The explanation tree connects at most one of the senses of a word or phrase with the other words in the sentence. Since we search the tree with minimum total length, the selection of the sense is globally optimal as opposed to locally optimal in many other methods, such as marker passing in semantic networks.

Consider an example sentence:

```
Applied Materials , Inc.  today announced new aluminum etch capability with its
single-wafer , multi-chamber Precision 5000 Etch system
```

The following lexical items are created by the lexical analyzer. The semantic interpreter then found a generalized subtree of the semantic network that connects the lexical items:

```
1.  [0,3] Applied Materials , Inc.  ((muctype name) +cap +corpdesig (name "Applied Materials")
    (desig "INC") (mucnode "entity"))

2.  [7,7] aluminum ((mucnode "film") (film ALUMINUM))

3.  [7,7] aluminum ((mucnode "bonding") (bonding ALUMINUM))

4.  [8,8] etch (+etch (mucnode "etching"))

5.  [8,8] etch ((muctype etching) +etch (mucnode "equipment") (equipment ETCHING-SYSTEM))
```

```
6.  [15,16] Precision 5000 (+equip (model "Precision 5000") (mucnode "equipment"))

7.  [17,17] Etch (+etch (mucnode "etching"))

8.  [17,17] Etch ((muctype etching) +etch (mucnode "equipment") (equipment ETCHING-SYSTEM))

9.  [18,18] system (-cap +equip +postmod (mucnode "equipment"))
```

The six of the lexical items can be connected by the following tree (the structure of the tree is indicated by indentation):

```
me-cap ((count 6))
  entities ((count 1))
    entity Applied Materials , Inc. ((muctype name) +corpdesig
                                     (name "Applied Materials") (desig "INC")
                                     (mucnode "entity") (count 1))
  process ((count 5))
    etching ((muctype etching) +etch (mucnode "etching") (count 5)
             (equipment ETCHING-SYSTEM))
      etching etch (+etch (mucnode "etching") (count 1))
      film aluminum ((mucnode "film") (film ALUMINUM) (count 1))
      equipment ((muctype etching) +equip +postmod +etch (model "Precision 5000")
                 (mucnode "equipment") (count 3) (equipment ETCHING-SYSTEM))
        equipment Precision 5000 (+equip (model "Precision 5000")
                                  (mucnode "equipment") (count 1))
        equipment Etch ((muctype etching) +etch (mucnode "equipment") (count 1)
                        (equipment ETCHING-SYSTEM))
        equipment system (+equip +postmod (mucnode "equipment") (count 1))
```

This tree identifies `Applied Materials` to be the entity with an aluminum etching process, where `Precision 5000` is used as equipment.

INTEGRATION WITH SYNTAX

Previous approaches to semantic interpretation can be classified as either one of the following:

Syntax-guided: Semantic structures are derived from a parse tree or parse tree fragments. The disadvantage of this is that the semantic analysis is critically dependent upon the output of syntactic analysis and syntactic ambiguities have to be resolved before semantic analysis.

Frame-guided: Semantic interpretation is driven by instantiation of frames that are triggered by keywords. The problem with this approach is that there is no principled method for controlling the interaction of multiple frames that may be triggered by the same word or the same set of words. Further more, this approach often results in complex frame definitions that are difficult to port to another domain.

Semantic interpretation in NUBA is **syntax-constrained** in the sense that the semantic structure must be consistent with syntactic structure. The notion of structural-consistency between semantic and syntactic structures is similar to the structural-consistency between parse trees [9].

Definition 6.1 (Span). *An integer interval $[i, j]$ is said to be a span of a sentence if there exists a parse tree and a node n in the parse tree such that the i'th to j'th word in a sentence is dominated exactly by a consecutive subtrees of n.*

The difference between the notion of span here and [9] is that the latter requires that a span consists of all the words that are dominated by a single non-terminal symbol. Figure 3 shows the spans in a parse tree. The spans in a parse forest is the unions of the spans in the trees in the forest.

Semantic interpretation in NUBA is based on:

Definition 6.2 (Structural-consistency Hypothesis). *The spans in semantic dependency structure are a subset of the spans in the parse forest.*

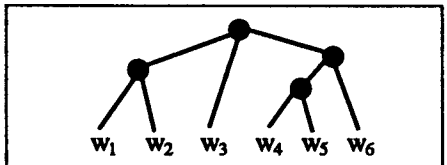

Figure 3: The spans in a sentence

From a packed shared parse forest, we can derive all the allowable spans in a sentence. During the message passing process, the messages are combined by a node only if the span of their combination is an allowable span according to the parse forest.

If the parse forest is not available, the parse tree is assumed to be a flat structure, where the category S (sentence) immediately dominates all the word (Figure 4). This means that any interval $[i, j]$ is an alloable span. The parse tree impose no constraints on the semantic structure. This is what we did in our official system.

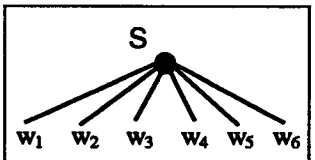

Figure 4: A flat tree.

Note that even though our official system do not make use of the parse forest, the connection trees obtained by the semantic analyzer are not arbitrary. They must be convex with respect to the sequence of words in the sentence.

Definition 6.3 (Convexity). *A tree connecting a sequence of elements is convex with respect to the sequence iff for any two elements w_i, w_j $(i < j)$ in the sequence, any node in the tree that dominates both w_i and w_j must also dominate all the element between w_i and w_j.*

Figure 5 shows examples of a convex and a non-convex tree. Although the syntax of different languages may be very different, they all seem to satisfy the convexity constraint. Therefore, without any help of a parser, the semantic analyzer in NUBA is still able to take advantage of a common denominator of the syntactic constraints in different languages.

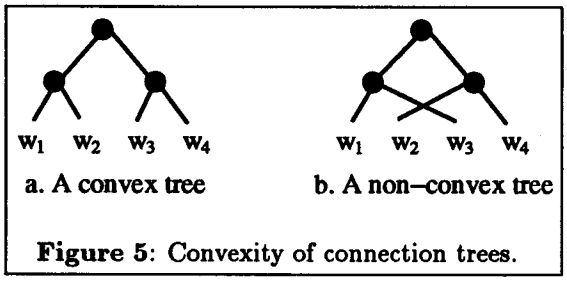

Figure 5: Convexity of connection trees.

DISCOURSE ANALYSIS

The discourse module in our official system is a striped-down-to-the-bare-bone version. The only function implemented is the unification of scenarios. The system maintains a list of scenarios. Whenever possible, a

newly generated scenario is merged with the scenarios in the list. If it is not unifiable with any of them, the new scenario is inserted into the list.

TEMPLATE GENERATION

The structure of the semantic network, hence the structure of the scenarios, is very similar to template structures. Once the scenarios have been identified, template generation is quite straightfowrard. One or more MICROELECTRONICS-CAPABILITY templates will be generated for each scenario. The roles of the entities involved in the scenario are determined by the attribute values of the **me-cap** node in the scenario. For example, if **+distribute** is present, then the entity fills the distributor role. When there are multiple entities, whether they are fillers of the same role or separate roles is determined by the **joint** attribute in the **me-cap** node.

ANALYSIS OF THE WALKTHROUGH EXAMPLE

We now show how the the walkthrough example is processed by our official system. The first sentence of the walkthrough example is:

> **In the second quarter of 1991 , Nikon Corp. <left> 7731 <right> plans to market the <q> NSR-1755EX8A , </q> a new stepper intended for use in the production of 64-Mbit DRAMs**

The following lexical items have been identified by the lexical analyzer:

1. **[7,8] Nikon Corp. ((muctype name) +cap +corpdesig (name "Nikon") (desig "CORP") (mucnode "entity"))**
2. **[14,14] market (+manufacture +distribute (mucnode "me-cap") (count 0))**
3. **[17,17] NSR-1755EX8A (+equip (model "NSR-1755EX8A") (mucnode "equipment"))**
4. **[22,22] stepper ((muctype lithography) +equip +lith (mucnode "equipment") (equipment STEPPER))**
5. **[28,28] production (+manufacture +distribute (mucnode "me-cap") (count 0))**
6. **[31,31] DRAMs ((muctype device) (device DRAM))**

The only relevant lexical item that is not identified by NUBA is the size of the device **64-Mbit**, this is because our official system does not attempt to fill the **device-size** slot.

These lexical items are sent to the nodes named in their **mucnode** attribute as initial messages. These messages will initiate a message passing process. When all the messages have been sent, NUBA finds the message at **me-cap** node with maximum **count** value. It then traces the origins of the message. The paths traversed during the trace form the following subtree of the semantic network, that connects the lexical items:

```
me-cap (+manufacture +distribute (mucnode "me-cap") (count 4))
  me-cap market (+manufacture +distribute (mucnode "me-cap") (count 0))
  entities ((count 1))
    entity Nikon Corp. ((muctype name) +corpdesig (name "Nikon")
                        (desig "CORP") (mucnode "entity") (count 1))
  process ((count 3))
    lithography ((muctype lithography) +lith (count 3) (equipment STEPPER))
      equipment ((muctype lithography) +equip +lith (model "NSR-1755EX8A")
                 (mucnode "equipment") (count 2) (equipment STEPPER))
        equipment NSR-1755EX8A (+equip (model "NSR-1755EX8A")
                                (mucnode "equipment") (count 1))
```

```
        equipment stepper ((muctype lithography) +equip (mucnode "equipment")
                           +lith (count 1) (equipment STEPPER))
    device DRAMs ((muctype device) (count 1) (device DRAM))
```

This tree represents a scenario with one microelectronic capability. The entity involved is **Nikon Corp.** The process is lithography for making **DRAM**, where a stepper with model name **NSR-1755EX8A** is used as the equipment. Other than missing the **device-size**, all the information contained in this scenario is correct. The scenario is then inserted into the list of scenarios. NUBA proceeds to analyses the next sentence:

> **The stepper will use an 248-nm excimer laser as a light source and will have a resolution of 0.45 micron , compared to the 0.5 micron of the company <g> latest stepper**

The word **compared** is treated similarly as negation words, all the lexical entries after the word are deleted by a lexical rule. The phrase that begins with **compared** is actually a reference to the second microelectronic capability. The slot **granularity** is also ignored in our official system. Therefore, the lexical item corresponding to **0.45 micro** is not identified by the lexical analyzer. The remaining lexical items are:

1. `[1,1] stepper ((muctype lithography) +equip +lith (mucnode "equipment") (equipment STEPPER))`
2. `[6,7] excimer laser ((muctype lithography) +equip (mucnode "equipment") (lithography LASER) (equipment RADIATION-SOURCE))`

The fact that one type of equipment can be part of another equipment is not represented in the semantic network. Therefore, the program thinks that the **stepper** and the **excimer laser** are two separate equipments. As a result, NUBA fails to infer that the type of the previous lithography process is **LASER** and generates a spurious microelectronic capability template. Since no entity is mentioned in the sentence, NUBA assumes it to be the latest reference to an entity, **Nikon Corp.**, and generates the following scenario:

```
me-cap ((count 2))
  process ((count 2))
    lithography ((muctype lithography) +lith (count 1) (equipment STEPPER))
      equipment stepper ((muctype lithography) +equip +lith
                         (mucnode "equipment") (count 1) (equipment STEPPER))
    lithography ((muctype lithography) (count 1) (lithography LASER)
                 (equipment RADIATION-SOURCE))
      equipment excimer laser ((muctype lithography) (mucnode "equipment")
                               +equip (count 1) (lithography LASER)
                               (equipment RADIATION-SOURCE))
  entities ()
    entity ((muctype name) +cap +corpdesig (name "Nikon") (desig "CORP")
            (mucnode "entity"))
```

This scenario means that the entity **Nikon Corp.** has two lithography processes. A stepper is used as equipment in one and a excimer laser is used as radiation source in the other. This scenario is unified with the previous scenario. The entity and the first lithography process is identical to that of the previous scenario. Therefore, only the second lithography process is added. The result of the unification is new scenario which replaces the previous one in the list of scenarios:

```
me-cap (+manufacture +distribute (mucnode "me-cap") (count 6))
  me-cap market  (+manufacture +distribute (mucnode "me-cap") (count 0))
  entities ((count 1))
    entity Nikon Corp. ((muctype name) +cap +corpdesig (name "Nikon")
                        (desig "CORP") (mucnode "entity") (count 1))
  process ((count 3))
    lithography ((muctype lithography) +lith (count 4) (equipment STEPPER))
      equipment ((muctype lithography) +equip +lith (model "NSR-1755EX8A")
```

```
            (mucnode "equipment") (count 3) (equipment STEPPER))
      equipment NSR-1755EX8A (+equip (model "NSR-1755EX8A")
                              (mucnode "equipment") (count 1))
      equipment stepper ((muctype lithography) +equip +lith
                         (mucnode "equipment") (count 1) (equipment STEPPER))
    device DRAMs ((muctype device) (count 1) (device DRAM))
  lithography ((muctype lithography) (count 1) (lithography LASER)
               (equipment RADIATION-SOURCE))
    equipment excimer laser ((muctype lithography) +equip (mucnode "equipment")
                             (count 1) (lithography LASER)
                             (equipment RADIATION-SOURCE))
```

The third sentence is:

```
Nikon will price the excimer laser stepper at 300-350 million yen , and the company
expects to sell 50 systems during the initial year of marketing
```

The lexical items are:

1. `[0,0] Nikon ((muctype name) +cap (name "Nikon") (mucnode "entity"))`
2. `[0,0] Nikon ((muctype name) +cap +corpdesig (name "Nikon") (desig "CORP") (mucnode "entity"))`
3. `[4,5] excimer laser ((muctype lithography) +equip (mucnode "equipment") (lithography LASER) (equipment RADIATION-SOURCE))`
4. `[6,6] stepper ((muctype lithography) +equip +lith (mucnode "equipment") (equipment STEPPER))`
5. `[14,14] company ((muctype name) -cap (name "Nikon") (mucnode "entity"))`
6. `[17,17] sell (+manufacture +distribute (mucnode "me-cap") (count 0))`
7. `[19,19] systems (-cap +equip +postmod (mucnode "equipment"))`
8. `[25,25] marketing (+manufacture +distribute (mucnode "me-cap") (count 0))`

There are two lexical items for the word **Nikon** due to a bug in a program. However, this bug does not make any difference in the analysis result of this sentence. The scenario generated for the third sentence is:

```
me-cap (+manufacture +distribute (mucnode "me-cap") (count 4))
  me-cap marketing (+manufacture +distribute (mucnode "me-cap") (count 0))
  entities ((count 1))
    entity Nikon ((muctype name) +corpdesig (name "Nikon") (desig "CORP")
                  (mucnode "entity") (count 1))
  process ((count 3))
    lithography ((muctype lithography) (count 1) (lithography LASER)
                 (equipment RADIATION-SOURCE))
      equipment excimer laser ((muctype lithography) +equip (mucnode "equipment")
                               (count 1) (lithography LASER)
                               (equipment RADIATION-SOURCE))
    lithography ((muctype lithography) +lith (count 2) (equipment STEPPER))
      equipment ((muctype lithography) +equip +postmod +lith
                 (mucnode "equipment") (count 2) (equipment STEPPER))
        equipment stepper ((muctype lithography) +equip +lith
                           (mucnode "equipment") (count 1) (equipment STEPPER))
        equipment systems (+equip +postmod (mucnode "equipment") (count 1))
```

The information in this scenario is properly contained in the previous scenario obtained from the first two sentences. Therefore, there is no change in the list of scenarios.

After analysesing all the sentences, there is one scenario in the list:

```
me_cap (+manufacture +distribute (mucnode "me_cap") (count 10))
  me_cap market ((count 2))
    entity  Nikon Corp. ((muctype name) +cap +corpdesig (name "Nikon")
                         (desig "CORP") (mucnode "entity") (count 2))
  process ((count 3))
    lithography ((muctype lithography) +lith (count 6) (equipment STEPPER))
      equipment ((muctype lithography) +equip +postmod +lith (model "NSR-1755EX8A")
                 (mucnode "equipment") (count 5) (equipment STEPPER))
        equipment NSR-1755EX8A ((muctype lithography) +equip +postmod +lith
                                (model "NSR-1755EX8A") (mucnode "equipment")
                                (count 3) (equipment STEPPER))
        equipment stepper ((muctype lithography) +equip +lith (mucnode "equipment")
                           (count 1) (equipment STEPPER))
      device DRAMs ((muctype device) (count 1) (device DRAM))
    lithography ((muctype lithography) (count 2) (lithography LASER)
                 (equipment RADIATION_SOURCE))
      equipment excimer laser ((muctype lithography) +equip (mucnode "equipment")
                               (count 2) (lithography LASER)
                               (equipment RADIATION_SOURCE))
```

The template generator then fills in a set of templates according to the scenario. The scores of the template filling is:

POS	ACT	COR	PAR	INC	SPU	MIS	ERR	UND	OVG	SUB
29	21	16	0	4	1	9	47	31	5	20

DEVELOPMENT EFFORTS

NUBA was implemented in C++. The software consists of three parts:

Utilities: such as generic list, LISP-like command interpreter, bit vectors, attribute value vectors, lexicons, graphs, hash tables, binary searched files *etc.* (16,000 lines).

Abduction: A message passing algorithm for obvious abduction and the procedures for retrieving the explanations (4000 lines).

MUC specific programs: (3200 lines), which can be further divided into:

- Message zoning and sentence recognition (600 lines).
- MUC specific message structures and attribute value constraints (1000 lines).
- Lexical rules for entity name and location recognition, *etc.* (600 lines).
- Template generation (700 lines)
- Others, such as gazetteer search, (300 lines).

Most of the utilities and the algorithm for obvious abduction were implemented before MUC-5.

TRAINING

We did not use any automated training. The bulk of training time was spent on adding and modifying the lexicon entries. We used all of the 1000 training texts. However, only the articles with high error rate were examined, which means that we spent a lot of time improving the text-filtering scores. This turned out to be quite futile. Although we managed to improve the text-filtering score for the training articles, the modifications we made did not generalize well to the testing articles.

SOURCES OF KNOWLEDGE

The semantic network is hand crafted, as it is fairly small. The lexical entries came from several sources. The model names and the company names were obtained from the key templates. NUBA can only recognize model names that appeared in the training data. However, the company names that do not appear in the training data can also be recognized if they are followed by a corporate designator. The country names and province names are from the English gazetteer. Other entries in the lexicon are created manually. Most of them are based on the document for template filling rules.

CONCLUSIONS

Our long term goal is to develop a theory and a system for abduction-based natural language understanding. Our participation in MUC-5 took an important step towards this goal by demonstrating the power of a abduction-based semantic interpreter. Our official system is still incomplete: the parser is not integrated and the discourse module is only a crude first approximation. Nevertheless, NUBA's performance in formal testing is quite impressive. Our approach provides an interesting and perhaps better alternative to the rule-based or pattern-based semantic interpretation.

We are certainly not the first to realize the relevance and importance of abductive reasoning in natural language understanding. However, computational complexity plagued many previous abductive understanding systems. We have shown that with proper restriction on knowledge and explanation structure, abductive inference can be made very efficiently.

ACKNOWLEDGEMENTS

This work was supported in part by the Defense Advance Research Projects Agency, monitored by Science Applications International Corporation under subcontract #19-940066-31, and by Natural Sciences and Engineering Research Council of Canada grant OGP121338. The author wishes to thank Jun Shih, Chunshen Yi and Demeng Chen for developing GUI and various utility programs.

References

[1] Eugene Charniak and Drew McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley Publishing Company, 1985.

[2] Jerry Hobbs. The generic information extraction system. In *Proceedings of the Fifth Message Understanding Conference*, 1993.

[3] Jerry R. Hobbs. Metaphor and abduction. Technical Note 508, SRI International, 1991.

[4] Jerry R. Hobbs, Mark Stickel, Paul Martin, and Douglas Edwards. Interpretation as abduction. In *Proceedings of 26th Annual Meeting of the Association for Computational Linguistics*, pages 95–103, Buffalo, 1988. ACL.

[5] Henry A. Kautz. A formal theory of plan recognition. Technical Report 215, Department of Computer Science, University of Rochester, May 1987.

[6] Dekang Lin. *Obvious Abduction.* PhD thesis, Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, 1992.

[7] Dekang Lin. Principle-based parsing without overgeneration. In *Proceedings of ACL-93*, pages 112–120, Columbus, Ohio, 1993.

[8] Dekang Lin and Randy Goebel. A message passing algorithm for plan recognition. In *Proceedings of IJCAI-91*, pages 280–285, 1991.

[9] Ezra Black John Lafferty Salim Roukos. Development and evaluation of a broad-coverage probabilistic grammar of english-language computer manuals. In *Proceedings of ACL-92*, pages 185–192, Newark, Dalaware, 1992.