The analysis lexicon and the lexicon management system

Sergei Nirenburg

Center for Machine Translation

Carnegie-Mellon University

1. Lexicon Acquisition

This paper deals with the process of analysis lexicon acquisition for MT and other NLP systems. The approach to MT is interlingua-based as developed for and partially implemented in TRANSLATOR, a knowledge-based system of English - Russian MT (see Nirenburg et al. 1985, 1986, 1987). Analysis in MT is the process of automatic translation from the source language into the interlingua, e.g., in TRANSLATOR, from English into IL. This process is based on two knowledge sources, the analysis (English --> IL) lexicon and the analyzer. This paper deals almost exclusively with the former, though, in the last three sections, there are some glimpses into what the analyzer may look like.

This paper is based on the following premises:

- lexicon acquisition is executed by humans assisted by an interactive aid which enhances productivity and ensures uniformity
- the MT system deals with a constrained realistic domain, i.e., a subworld served by a natural sublanguage
- as any NLP system, an MT system needs three interrelated but distinct lexicons , namely
  - the world concept lexicon which structures our knowledge of the world
  - the analysis lexicon which is indexed by natural language words and phrases connected with concepts from the world concept lexicon, and
  - the generation lexicon, which is indexed by concepts in the world concept lexicon connected with natural language words and phrases.

The interactive aid is a lexicon management system (LMS) whose functionalities with respect to the task of language analysis are discussed here. The subworld/sublanguage used here for illustration purposes is that of computer science, the same domain in which TRANSLATOR operates. The main principle underlying this research and distinguishing it from the work of other groups interested in interactive aids (e.g., Ahlswede 1985; see also Nirenburg and Raskin 1987, Section 1 and references there) is the firm conviction that both the analysis and generation lexicons (AL and GL, respectively) are based on the concept lexicon (CL), whose acquisition must precede that of the former lexicons.

2. Summary of LMS and CL.

In this section, the previous work on the TRANSLATOR LMS and CL (Nirenburg and Raskin 1987) is briefly summarized. An LMS is a collection of programs that help create, augment, modify and test the various lexicons in an NLP application. The particular LMS referred to in this paper is suggested for the TRANSLATOR project. The goal of TRANSLATOR is ultimately to develop a knowledge-based multilingual machine translation system for multiple subject areas. Various modules in the system are designed so as to allow interactive human participation with no pre- or post-editing. The LMS is one such module. LMS maintains all the various types of lexicons in a NLP system. The primary purpose of an LMS at the first stage of the project is to support knowledge acquisition. At later stages in the life of the LMS, testing and modification will become the primary types of work it supports. CL is the first to be acquired.

An ordinary LMS user, i.e., an enterer , will obtain at this stage a list of concepts to enter in the CL and will code the information about them in a specially developed DRL knowledge representation language.

The LMS assists the enterer by providing graphic and other aids for human decision making. In case of doubt the enterer can try to resolve the difficulty or to refer to the lexicon manager, whose responsibility it is to force solutions to problems in lexicon acquisition. The task of the manager has much in common with that of a database administrator in the database system environment. In their respective capacities, they are both responsible for

- maintaining the format and contents of the knowledge representation language ('data dictionary' in database terms)
- defining and executing the security, consistency and integrity checks for the accumulated data
- developing and running statistic analysis routines to monitor access time, etc. (this becomes important in production-size lexicon systems)
- interfacing with regular users (enterers for the lexicon manager).

In addition to the above, the lexicon manager also modifies the knowledge representation language in accordance with the evidence accumulated in the process of knowledge acquisition. The above means that the LMS has two modes of operation: a mode for enterers and a mode that supports the activities of an lexicon manager.

The completed CL is a complex network, with concepts as nodes. The connections in this network place the nodes in various (tangled) hierarchies and classify them on the basis of certain characteristics and constraints. We suggest a frame-based representation in which frames correspond to concepts, and slots convey constraints on the meaning of these concepts. The sets of values that can occupy certain slots, the domains of the latter, can be further classified. Thus, some slots take names of concepts in the world as values (such are hierarchy-related slots or *relations*); some others, take values from specially defined sets; these are *properties*. The slots can be occupied by any number of members of the corresponding domain, and the logical operators *and, or,* and *not* can be used to augment the expressive power. Also, in every case, the semantics of the constraints in the lexicon is that of default knowledge: the contents of a slot are understood as *typically* constraining the meaning of the concept.

The current version of the CL is presented in Nirenburg, Raskin, and Tucker (1985, 1986, 1987) and in Nirenburg and Raskin (1987). Figure 3 in the latter contains the current version of the *isa* hierarchy underlying the CL. The examples in the following two sections of this paper will, however, be self-sufficient. The work on the AL begins theoretically on the completion of a version of the CL, and the analysis facet of the LMS draws constantly and extensively on the CL.

3. The AL and the analysis facet of the LMS.

The main function of ALs is to connect units of a natural language with the corresponding concepts or property values in a subworld CL. There has been, however, a significant amount of confusion in the AI community about the relation between the analysis and CLs as well as about the role of the AL and the types of information it should contain. It must also be reiterated here that the main direction of research in natural language processing has for a long time been developing schemata for representing world and lexical knowledge, not actually acquiring the knowledge itself. A good example of the relative importance of the knowledge representation work in AI is the influence of the research in semantic network-based knowledge representation schemata such as Quillian's (1968), Bobrow and Winograd's (1977),

Brachman's (1979) or Hirst's (1983). Indeed, one has to devise a format to record knowledge before proceeding to actually acquiring it. Unfortunately, the emphasis on knowledge representation languages has left the work on actual knowledge acquisition in natural language processing with the aura of a secondary task.

We believe that a significant amount of research work must actually be performed to acquire knowledge even in a restricted subworld, even and specifically after a particular knowledge representation format has been chosen. A frequent inaccuracy in building lexicons for NLP is the lack of distinction between the CL and the AL (cf. Ahlswede 1985). As a result, the crucial decisions determining the structure of the subworld are made unconsciously and arbitrarily in the process of postulating certain semantic features and assigning them to specific entries. We agree with the Tacitus (Hobbs 1986:220) approach 'to define rich core theories of various domains... and then to define... English words in terms of predicates provided by these core theories.' The TRANSLATOR CL seeks to become such a rich core theory of the computer science domain.

The sense-frames developed and used in the Collative Semantics project at the Computing Research Laboratory, the New Mexico State University (Fass 1986; Wilks and Fass 1984), are another example of the lack of distinction between the CL and AL. The set of slots used in these frames is a subset of those in the TRANSLATOR concept lexicon. The sense-frames contain just the preference-oriented slots while the TRANSLATOR CL contains a full characterization of each entry. The sense-frames are meant to be used to describe English lexical units but, in fact, they represent conceptual submeanings, with the disambiguation already done in an unspecified way.

Another problem with the existing dictionaries is the confusion of the lexical information proper in the entries with the commands for the analyzers or parsers. While the lexicon-driven analyzers (Birnbaum and Selfridge 1981; Cullingford and Onyshkevych 1985) adhere the principle of inclusion of both types of information, they do not typically make an effort to keep them apart and that significantly complicates the use of such a dictionary in a system of natural language processing. Wilks and Fass' sense-frames contain primarily command-type information in the preference slots used to filter out incompatible word combinations.

Still in other systems, different blocks are introduced to contain information of different degrees of complexity. Thus, for example, the Yale school postulates the existence of a separate 'knowledge' level of conceptual representation (and a separate formalism for it), in addition to the representations in the conceptual dependency formalism (Schank and Abelson 1977; Wilensky 1983; Schank 1982). The knowledge level includes the knowledge about 'memory organization packets,' scripts, plans, or goals. Typically, the knowledge that pertains to the above is kept separate from a 'lexicon.' Such an arrangement may be not justified in terms of the nature of the information contained in these various places. Besides, it creates the unnecessary task of distinguishing among those different types of information and of working out a traffic pattern among them in the process of semantic analysis.

Unlike Schank's (1973) 'conceptual dependencies,' and similar to Fass and Wilks' 'collative semantics' (Fass 1986), the proposed analysis lexicon is not based on any small set of primitives. This follows immediately and naturally from the proposed format of the CL (Nirenburg et al. 1987). Instead, the entries in the proposed AL are defined in terms of the various elements of the CL. Each entry in the AL is a command to the analysis program. The following types of commands are distinguished:

- instantiate a specified concept available in the CL; the concept to be instantiated can have to be determined by performing a test; the act of instantiation itself can also be conditional (e.g., the entry for DATA in Section 5)
- insert a value into a property slot of one of the frames in the structure that holds the (current) results of analyzing the source language sentence; the value to be inserted can be either listed directly in the AL or can be accessed indirectly through a pointer to an appropriate concept in the CL (e.g., the entry for PERMANENTLY in Section 5)
- serve as test or control knowledge for the analyzer decisions concerning the representation of the various meanings of the original phrase (e.g., the entry for THE in Section 5)

In the following section, a sample sentence from the computer sublanguage is presented as analyzed by the TRANSLATOR analyzer. Section 5 demonstrates the format of the AL entries for the words of the sample sentence. And, finally, Section 6 discusses the analysis facet of the LMS.

4. A Sample Sentence Analyzed in TRANSLATOR

A typical sentence from the computer sublanguage has been selected for the illustrative analysis below. The sentence is:

Data such as the above, that are stored more or less permanently in a computer, we term a database.

What follows is the results of analysis of the example sentence by TRANSLATOR's analyzer. This set of frames constitutes an instance of what we call an interlingua (IL) text.

```
(object
 (id object1)
 (is-token-of data) *
 (subworld computerworld) *
 (quantifier (type all) (scope (and clause1 clause2))))

(object
     (id object2)
     (is-token-of computer)
     (subworld computerworld)
     (quantifier any))

(object
     (id object3)
     (is-token-of database)
     (subworld computerworld))

(state
     (id state1)
     (is-token-of be-equivalent)
     (phase static)
     (patient1 object1)
     (patient2 (antecedent-of above))
     (time always)
     (space none)
     (subworld computerworld))

(state
     (id state2)
     (is-token-of in)
     (phase static)
     (patient1 object1)
     (patient2 object2)
     (time always)
     (space none)
     (subworld computerworld))

(state
     (id state3)
     (is-token-of  be-a-name-of)
     (phase static)
     (patient1 object3)
     (patient2 object1)
     (time always)
     (space none)
     (subworld computerworld))
```

```
(clause
    (id clause1)
    (discourse-structure (+expan clause1 clause3))
    (event state1)
    (focus state1.patient2)
    (modality conditional)
    (subworld computerworld)
    (time always)
    (space none))

(clause
    (id clause2)
    (discourse-structure (+expan clause2 clause3))
    (event state2)
    (focus time)
    (modality conditional)
    (subworld computerworld)
    (time always)
    (space (in object1 object2)))

(clause
    (id clause3)
    (discourse-structure none)
    (event state3)
    (focus object3)
    (modality real)
    (subworld computerworld)
    (time always)
    (space none))

(sentence
    (id sentence1)
    (main-clause clause3)
    (clauses clause1 clause2)
    (subworld computerworld)
    (modality real)
    (focus object3)
    (speech-act (type definition)
                (performative direct)
                (speaker author)
                (hearer reader)))
```

Obviously, this analysis must be based on an AL. The next section presents the entries for all the words of the example sentence in the TRANSLATOR English - IL lexicon of the computer sublanguage. These entries will indeed lead to the analysis results presented above.

5. AL Entries for the Example Sentence

Since the AL is preceded and largely determined by the CL, we will first present a fragment of the TRANSLATOR CL containing the entries for the concept nodes used in the example sentence:

```
(data
 (isa information)
 (subworld computerworld officeworld world)
 (object-of computer-mental-action)
 (instrument-of mental-action)
 (belongs-to user)
 (consists-of file record byte)
 (part-of database))

(store
  (isa operate)
  (subworld computerworld)
  (consists-of (locate agent destination)
               (send agent object destination))
  (part-of computer-mental-action)
  (precondition (thereexists object destination)
                (controls agent object))
  (effect (in object destination))
  (tempor computer-mental-action)
  (agent user)
  (object data)
  (instrument operating-system DBMS)
  (destination computer-memory database))

(computer          ;the physical object computer
  (isa device)
  (subworld computerworld)
  (consists-of (box board cable peripherals)
               (in board box)
               (connect cable box peripheral))
  (belongs-to organization person)
  (object-of use)
  (size size-set)
  (shape shape-set)
  (color color-set)
  (mass integer))
```

```
(define
  (isa mental-action)
  (subworld computerworld scienceworld)
  (precondition (thereexists patient1)) ;patient1 = definiendum
  (effect (be-a-name-of patient2 patient1))
  (agent author)
  (patient1 mental-object)
  (patient2 mental-object)
  (source author))

(program
  (isa information)
  (subworld computerworld)
  (part-of system)
  (consists-of code)
  (object-of computer-mental-action)
  (instrument-of computer-mental-action))

(database
 (isa data)
 (subworld computerworld)
 (consists-of data)
 (belongs-to user)
 (object-of manage-database))

(to-be-a-subset-of
 (isa mental-state)
 (subworld computerworld world)
 (patient1 all)
 (patient2 all)
 (precondition  ;patient1 is a member or a subset of patient1;
                ;there is a certain defining property for all
                ;members of patient2 (cf. all people such as Peter)
)

(author
 (isa person)
 (subworld computerworld scienceworld cultureworld world)
 (source text))
```

What follows now is a fragment of the English - IL dictionary for the example sentence. The marker # stands for an empty string, which in this case means that no CL concept has been found to correspond to the SL lexical unit in question. The lexical units in parentheses show that there are additional meanings (not given in the sample dictionary) for the lexical units involved.

DATA	data
SUCH	to-be-a-subset-of; the task of looking for fillers of patient1 and patient2 is triggered by the unfilled slots in the instantiated frame for this state
AS	#; test whether SUCH precedes; if so, AS precedes patient1 of 'to-be-a-subset-of'
THE	#; an NP follows; this NP is coreferential with an object already instantiated
[THE	#; an NP follows; set the value of the slot 'quantifier' of this NP to 'every']
ABOVE	#; if a noun, then look for the appropriate instance of NP to which ABOVE refers (deixis resolution)

[ABOVE	#; if a preposition, insert the value (above actant1 actant2) in the instances of both actant1 and actant2]
THAT	#; if a relative conjunction, then instantiate a clause and insert the proper NP into an appropriate actant slot of the clause event
[THAT]	
BE	#; if an auxiliary in passive then signal that the clause event is the state which is the effect of the IL correlate of the main verb
[BE]	
STORE	store
[STORE]	
MORE OR LESS	#; a value of quantifier2; makes the concept or property value it modifies fuzzy; belongs to the same class as VERY, ALMOST, APPROXIMATELY...
PERMANENTLY	#; insert the value 'always' in the time property slot of the event which this word modifies
IN	#; insert the meaning of the modified NP in the 'space' of the clause event
[IN]	
A	#; an NP follows; it should be represented with a newly instantiated object frame, with 'any' as the value of the quantifier slot
COMPUTER	program
[COMPUTER]	
WE	author
TERM	define
DATABASE	database

The final section discusses the elements of the LMS for the compilation of the entries for the example sentence AL.

6. Elements of LMS-AL

The LMS for the AL aids both the manager and the lexicon enterers. Just as the already implemented TRANSLATOR LMS-CL, LMS-AL will provide a variety of ways to direct the thought processes of the lexicon writer by offering graphics-oriented displays and editing facilities, intelligent suggestions with respect to the contents of the entries, fast access to reference sources, reliable bookkeeping, efficient storage and retrieval of available lexical data, extensive help facilities, including tutorials, etc.

The first task for the manager of LMS-AL is to determine the list of entries for the sublanguage. The manager starts with the following resources:

- a corpus of texts in the sublanguage
- the current version of the CL
- a program which creates frequency lists for all word combinations in the corpus, in any of their grammatical forms, of required (variable) length with frequencies above a prescribed threshold value.

The last resource helps the manager to determine the phrasality of certain word combinations in the

sublanguage. The manager selects good candidates for phrasal entries and the frequency program of the LMS can provide the data about the percentage of the occurrence of any component of a potential phrasal entry in or out of the phrase. The higher the percentage of the coocurrence of all the components of the phrase, the higher the desirability of listing the combination as a (phrasal) entry in the AL. The statistics aids the manager primarily in cases of semantic doubt. He does not need to use this facility if there is no doubt in his mind about the phrasality of a word combination in the sublanguage.

A more complicated task is to obtain a frequency list of candidates for discontinuous phrasal entries (e.g., "to give *<NP>* a raincheck"). The question here is whether one can write an efficient search program for this purpose. No such facility is yet available for the TRANSLATOR LMS.

The manager's next task is to determine the polysemy in the sublanguage. He needs the following resources:

- on-line dictionary of the language
- a look-up program which checks every word in the sublanguage corpus for polysemy in the on-line dictionary; the polysemous items are extracted from the dictionary and collected in a
- special file of candidates for polysemy
- a grep-like program which lists every clause (not line) in which a word of the sublanguage occurs in the corpus.

Having obtained the list of potentially polysemous words, the manager goes over the special file with their dictionary definitions and rejects most of them out of hand because, typically for any specialized sublanguage (see Raskin 1987 and references there), most of the meanings (usually all but one) in the dictionary do not belong to the sublanguage. In other words, the sublanguage polysemy is much more limited - thus, the analysis of the English word *operator* in Nirenburg and Raskin (1987) demonstrated that the computer sublanguage realizes a part of one of the 7 meanings the word has in English as a whole. .LP If, however, the manager is in doubt about the applicability of more than one meaning of the word to the sublanguage, he uses the search program, such as, for instance, the Unix[1] *grep* command, to review the uses of the word in the corpus. If a decision has to be made about, say, the 2-way polysemy of the word *computer* as 'hardware' or 'software,' the manager splits into 2 entries, *computer1* ('hardware') and *computer2* ('software'), with the material in the parantheses serving as a guide for the enterer. If the number of the polysemous words is very small, which may indeed be the case, it might be beneficial to have the manager himself enter them.

After the list of entries, including phrasal and polysemous ones, is established, it is distributed among the enterers. Typically, the lexicon enterer gets a word and has to come up with an entry in the subworld AL for it. The resources he has at his disposal include:

- corpus of texts in the sublanguage;
- on-line SL dictionary (for humans);
- CL of the sublanguage;
- AL dictionary in its present state;
- all the graphical and interactive (including help) facilities of the LMS.

[1]Unix is a trademark of Bell Laboratories

As established in Section 3, there are three types of entries. First, an SL word meaning may correspond to a concept or a property value in CL. Secondly, it may correspond to a value to a slot in an IL text frame, such that this slot is not in CL. Thirdly, an entry may combine these two types of information.

The enterer then picks up the first entry head in the list. The task now is to decide what type of entry this entry head calls for. For the 'open' parts of speech, i.e., verbs, nouns, adjectives, and adverbs, the enterer scans the hierarchy in the CL. For verbs and nouns, two outcomes are possible. First, he may find the corresponding concept (obviously, not necessarily marked by the same word or expression). The concept may

- coincide entirely with the meaning of the entry head or
- be more general than the meaning of the entry head or
- be more specific than the meaning of the entry head or
- partially overlap with the meaning of the entry head

If it coincides entirely, the concept constitutes the entry (see Section 5).

If the concept is more general, another leaf probably needs to be added to the *isa* hierarchy (e.g., the entry head is 'barn' and the corresponding concept is 'building'). The enterer makes a suggestion to this effect to the manager.

If the concept is more specific than the entry head meaning then the enterer looks at the concept's ancestors in the *isa* hierarchy until he discovers a more general concept. At this point either the previous solution (adding a descendent to the more general concept) is possible or an intermediate concept should be added between a parent and a child. The enterer makes a suggestion to the manager.

If the entry head meaning and the concept overlap the easiest solution (which will not always be possible) is to rearrange the entry head meaning so that part of it fit the concept exactly and the residue is treated as a different meaning within this procedure. Unless both the new submeaning and the residue correspond exactly to the available concepts, the enterer's decision is referred to the manager for approval.

For adjectives and adverbs, the enterer scans the CL as well. However, in these cases, he looks not for concepts but rather for property values in object frames for adjectives and in process frames for adverbs.

For the closed-class parts of speech, there is not much room for generalization. The list of each such category is short and the treatment may be different even among the members of the same class, e.g., pronouns, let alone among the different classes. It is advisable, therefore, to do the work prior to the stage at which the lexicon enterers begin their work. The enterers will be advised of that by scanning the AL accumulated so far.

The enormous advantage of the LMS-AL is that it simplifies the compilation of the AL and reduces it to a number of routine and uniform operations executable by low-level employees and yielding a high-quality knowledge resource. What makes the procedure this way is, of course, the pre-existence of the CL for the corresponding subworld and of the LMS. Once, one implements an LMS, one cannot figure out how one (or anybody else) could possibly do without it.

References

Ahlswede, T. E. 1985. A Tool Kit for Lexicon Building. Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics. University of Chicago, Chicago, July, pp. 268-276.

Birnbaum, L. and M. Selfridge 1981. Conceptual analysis of natural language. In: R.C. Schank and C. Riesbeck (eds.), **Inside Computer Understanding**. Hillsdale, NJ: Lawrence Erlbaum.

Bobrow D. and T. Winograd 1977. An Overview of KRL, a knowledge representation language. *Cognitive Science*, vol. 1, pp. 3-46.

Brachman, R. J. 1979. On the Epistemological Status of Semantic Networks. In: N. V. Findler (ed.), **Associative Networks: Representation and Use of Knowledge by Computers**. New York: Academic Press, pp. 3-50.

Cullingford, R. E. and B. A. Onyshkevych 1985. Lexicon-Driven Machine Translation. In: Nirenburg (ed.), pp. 75-115.

Fass, D. 1986. Collative Semantics: A Description of the Meta5 Program. Technical Report MCCS-86-23, Computer Research Laboratory, New Mexico State University.

Hirst G. 1983. Semantic Interpretation Against Ambiguity. CS-83-25, Brown University.

Hobbs J. R. 1986. Overview of the Tacitus Project. *Computational Linguistics*, Vol 12, No. 3, pp. 220-2.

Nirenburg, S. (ed.) 1985. Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages, Colgate University, Hamilton, NY, August.

Nirenburg, S. (ed.) 1987. **Machine Translation: Theoretical and Methodological Issues**, ACL Series 'Studies in Natural Language Processing,' Cambridge University Press.

Nirenburg, S. and V. Raskin 1987. The Subworld Concept Lexicon and the Lexicon Management System. *Computational Linguistics* (in print).

Nirenburg, S., V. Raskin and A. B. Tucker 1985. Interlingua Design for TRANSLATOR. In: Nirenburg (ed.), pp. 224 - 244.

Nirenburg, S., V. Raskin and A. B. Tucker 1986. On Knowledge-Based Machine Translation. Proceedings of COLING-86, Bonn, Germany, August, pp. 627-632.

Nirenburg, S., V. Raskin and A. B. Tucker 1987. The Structure of Interlingua in TRANSLATOR. In: Nirenburg (ed.), pp. 90 - 113.

Nirenburg, S., V. Raskin and A. B. Tucker (eds.) 1985. The TRANSLATOR Project. Colgate University, Hamilton NY.

Quillian, M. R. 1968. Semantic Memory. M. Minsky (ed.), **Semantic Information Processing**. Cambridge, MA: MIT Press, pp. 216-70.

Raskin, V. 1987. Linguistics and Natural Language Processing. In: Nirenburg (ed.), pp. 42 - 58.

Schank, R. C. 1973. Identification of Conceptualizations Underlying Natural Language. In: R. C. Schank and K. M. Colby (eds.), **Computer Models of Thought and Language**. San Francisco: Freeman, pp. 187-247.

Schank, R. C. 1982. Reminding and Memory Organization: An Introduction to MOPs. In: W. Lehnert and M. Ringle (eds.), **Strategies for Natural Language Processing**. Hillsdale, NJ: Lawrence Erlbaum.

Schank, R. C. and R. Abelson 1977. **Scripts, Plans, Goals and Understanding**. Hillsdale, NJ: Erlbaum.

Wilensky, R. 1983. **Planning and Understanding.** Reading, MA: Addison-Wesley.

Wilks, Y. A. and D. Fass 1984. Preference Semantics, Ill-Formedness and Metaphor. *American Journal of Computational Linguistics*, vol. 9, pp. 178-187.