CERIAS Tech Report 2001-10

NATURAL LANGUAGE PROCESSING FOR INFORMATION ASSURANCE AND SECURITY: AN OVERVIEW AND IMPLEMENTATIONS

by Mikhail J. Atallah, Craig J. McDonough,
Victor Raskin, and Sergei Nirenburg

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907

Natural Language Processing for Information Assurance and Security: An Overview and Implementations

Mikhail J. Atallah, Craig J. McDonough, Victor Raskin
Center for Education and Research in Information Assurance and Security
(CERIAS, www.cerias.purdue.edu)
Purdue University
W. Lafayette, IN 47907
mja, raskin, mcdonoug@cerias.purdue.edu
Sergei Nirenburg
Computing Research Laboratory, New Mexico State University
Las Cruces, NM 88003
sergei@crl.nmsu.edu

Abstract

This research paper explores a promising interface between natural language processing (NLP) and information assurance and security (IAS). More specifically, it is devoted to possible applications to, and further dedicated development of, the accumulated considerable resources in NLP for, IAS. The expected and partially accomplished result is in harnessing the weird, illogical ways natural languages encode meaning, the very ways that defy all the usual combinatorial approaches to mathematical--and computational--complexity and make NLP so hard, to enhance information security. The paper is of a mixed theoretical and empirical nature. Of the four possible venues of applications, (i) memorizing randomly generated passwords with the help of automatically generated funny jingles, (ii) natural language watermarking, (iii) using the available machine translation (MT) systems for (additional) encryption of text messages, and (iv) downgrading, or sanitizing classified information in networks, two venues, (i) and (iv), have been at least partially implemented and the remaining two (ii) and (iii) are being implemented to the proof-of-concept level. We must make it very clear, however, that we have done very little experimentation or evaluation at this point, though we are moving quickly in that direction. The merits of the paper, if any, are in its venture to make considerable progress achieved recently in NLP, especially in knowledge representation and meaning analysis, useful for IAS needs. The NLP approach adopted here, ontological semantics, has been developed by two of the coauthors; watermarking is based on the pioneering research by another coauthor and his associates; most of the implementation of the password memorization software has been done by the fourth coauthor. All the four of us have agonized whether we should report this research now or wait till we have fully implemented all or at least some of the systems we are developing. At the end of the day, we have reached a consensus that it is important, even at this early stage, to review for the information security community what NLP can do for it and to invite feedback and further efforts and ideas on what seems likely to become a new paradigm in information security. To the body of the paper, we have added two self-contained deliberately reference-free appendices on NLP and ontological semantics, respectively, primarily for the benefit of those IAS readers, who are interested in expanding their understanding of those fields and further exploring their possible fruitful interactions with IAS.

1. Introduction: Natural Language and Code-Breaking

Perhaps the most dramatic instance of the use of natural language in communication security occurred during the World War II, in which both sides were very successful in breaking each other's codes and increasingly aware of it as well as gravely apprehensive about it. At a late critical point, on the advice of a prominent American linguist Edward Sapir, two Navajo speakers, each assigned to the US and British General Headquarters, respectively, transmitted top-secret messages to each other in their native tongue without any encryption. The German famed code-breaking unit could not break the "code." This success was due to the fact that the enemy did not even realize that the Allies were using a natural language, let alone the almost-extinct Navajo language (and even if they had realized it they simply had no one who knew that language). The German code-breaking team was rumored then to have paid with their lives for their failure to breach the security of this particular natural-language-based communication scheme.

Obviously, many people in IAS are aware of that episode, as probably is the other side. The element of surprise is gone, and even if a special effort is needed to recognize the natural language as one of the 5,600 or so extant languages, if such an episode recurred, the decoders would be more successful this time around, perhaps after trying Navajo first! The approach may still be good for shock value, in the short run, possibly just as a one-time trial. Also, it is hard and expensive to organize.

This episode is perhaps ancient history (and/or folklore: other versions mention different Amerindian languages--Shawnee, Choctaw-- and other war theaters) in the rapidly developing field of

IAS, but it served as a kind of inspiration and a source of intuition for us at the start of our joint effort. Our combined experience in NLP and IAS made us believe that natural language could still contribute an important level to information security, perhaps as dramatic as the classic--and, very likely, largely apocryphal--example described above. Accordingly, we describe below some lines of investigation that we are following, for bringing powerful NLP techniques to bear on some important problems in information security.

We are currently exploring four specific venues of applying NLP to IAS:

- <u>Natural Language and Humor Generation for Memorizing Random Strings</u> (e.g., Passwords, PINs, etc.). Let Σ be a random string representing something that a human is supposed to remember, e.g., a password, a PIN, etc. How does one construct a mnemonic that helps the human remember Σ? The ``Sing-a-Password" software tool built by Meunier (1998) is a successful first step in this direction, and uses popular melodies and tunes (suitably modified) to help a human remember randomly generated Σ's. The Meunier software focuses simply on meshing the lyrics with music; it does not worry about where the lyrics come from. It other words, it totally lacks the NLP component, and we have put it in. As a result, we have extended "Sing-a-Password" by making it capable of generating sentences with meaning (eventually, humorous meaning) to help the user to memorize randomly generated and otherwise meaningless passwords. This immediately gets into such NLP issues as automatic humor generation, but with the added constraint that the humor generated must help in remembering the passwords. To convince the reader of the importance of this effort, we recall that the COPS software tool (Farmer and Spafford 1990) has revealed a distressing pattern of poor password choices by users who followed the above process in the wrong direction: they chose a password Σ because it was easy to remember, instead of generating a random Σ and then trying to come up with ways of remembering it. The success of the research we have already partially implemented (see Section 2) could therefore have a substantial impact on the practice of information security. Two important disclaimers seem to be in order here. First, this particular direction in our NLP-IAS research is, of course, premised in a claim that having a randomly generated password is much preferable to using a meaningful, memorable and, therefore, more easily predictable user-selected password. We are under no illusion that randomly generated passwords solve the problem of IAS. We do, however, believe, as many but by no means all IAS experts do, that it is a good first line of defense against intrusion. It is reasonable to assume also that, for a large number of ordinary private users, such as the 70,000 or so students at our universities, this will remain for a long time their only contribution to the safety of their computer accounts. It is also a good education al step towards enhancing public awareness about IAS. Second, we are fully aware that our implementation of the software only for 8-character single-case alphabetic-only passwords is just the first step: limiting randomly generated passwords to this formal would actually narrow the combinatorial possibilities so seriously as to actually damage rather than increase IAS.
- <u>NLP for Watermarking</u>. We are developing software capable of embedding a hidden textual watermark in a textual message without changing the meaning of the text at all and the wording only slightly if necessary. To build this application, we are researching the interface of the theory of quadratic residues and of specially constrained natural language generation. Let *T* be a natural language text, and let *W* be a string that is much shorter than *T*. We wish to generate natural language text *T'* such that:
  - *T'* has essentially the same meaning as *T*;
  - *T'* contains *W* as a secret watermark, and the presence of *W* would hold up in court if revealed (e.g., *W* could say, ``This is the Property of X, and was licensed to Y on date Z");
  - the watermark *W* is not readable from *T'* without knowledge of the secret key that was used to introduce *W*;
  - for someone who knows the secret key, *W* can be obtained from *T'* without knowledge of *T* (so there is no need to permanently store the original, non-watermarked copy of copyrighted material);
  - unless someone knows the secret key, *W* is difficult to remove from *T'* without drastically changing the meaning of *T'*, and we are working on ways to make it even more difficult;
  - the process by which *W* is introduced into *T* to obtain *T'* is not secret, rather, it is the secret key that gives the scheme its security;
  - there is built-in resistance to collusion by two people who have differently watermarked versions of the same text, that is,

suppose watermarked versions of *T* are sold to *A* and to *B*: if buyer *A* has *T_A'*, where *W_A'* is hidden using a key that is not known to *A*, and buyer *B* has *T_B'* where *W_B'* is hidden using a key that is not known to *B*, then even if *A* and *B* were to share all the information they have they would not be able to either read or delete the watermark (from either *T_A'* or *T_B'*).

The solutions we sketch in Section 3 will typically satisfy all but the last of the above requirements, but we are optimistic we can modify our ideas to also satisfy it; in fact we can already satisfy it if we assume W_A' = W_B' (i.e., if the watermark makes no mention of the buyers A or B, and just mentions the seller).

- MT Techniques for Information Security. The method of encoding information in each natural language is highly idiosyncratic, and only customized language resources, such as ontologies, lexicons, analyzers, and generators allow to access the layer of meaning underlying text. We would like to explore how text in a natural language, possibly encoding a secret message (such as the *T'* mentioned above) can be automatically translated, in a way controlled by a secret key, into another natural language (resulting in *T''*), and it is this translated message *T''* that is actually transmitted or made public. This technique can be thought of as either the main security mechanism (e.g., in case *T'* was generated without using a secret key), or as an extra layer of security "on top" of the other methods used. In either case, the recipient will not be able to get *T'* without having the secret key that governs the behavior of the MT software used to create *T''* from *T'*. We see a way to do this starting from English, and automatically translate into another language, as well as the other way around.
- Downgrading, or Sanitizing Information. Increasingly, in interagency exchanges in the government, international coalition communication, and exchanges among business partners, there has been a need to develop an intricate architecture for combining a "high" network and a "low" network. Authorized users, with access to the high network, where sensitive data are stored and exchanged, must have access to the low network, but not the other way around. If this is all there is to it, the communication between the two networks is assured with the help of a variety of switches and one-way filters: the low-network information can propagate up but the high-network information must not leak down. There are enough technical and conceptual problems with such one-way filters, but they are multiplied manifold if there is also a need to share some high-network information with the low-network users but in a way that removes all the sensitive data. In this context already, the essentially semantic ability to recognize a sensitive message comes into play (see, for instance, Nelson 1997). In this paper, we are focusing only on sanitizing textual information. In other words, for each classified text T there must be generated a sanitized, downgraded text T', from which all sensitive data are removed according to a certain list of criteria. We are doing this by utilizing the NLP resources developed by the ontological-semantic approach (Nirenburg and Raskin 2001), which allows deep-meaning penetration and, as a result, much enhanced sensitive information detection and removal.

In Appendix 1, we formulate briefly, for the benefit of the IAS community, what exactly is meant by natural language and NLP. The terms are widely used and equally widely abused, both inside and outside of the fields of theoretical and computational linguistics, but especially so on the outside, where, for instance, the myth of non-feasibility of NLP as something which is "10-20 years down the road" can still be encountered. A tight description of the terms will help us to avoid any further misunderstanding. The Appendix can be skipped by those who are familiar with elements of formal linguistics and / or NLP.

2. Natural Language and Humor Generation for Memorizing Random Strings

2.1 Necessity for Mnemonics for Strong Passwords and PINs

Many computer breakins can be traced back to a poorly chosen password (one that is easy to guess, to derive, to find in a dictionary, etc.). Breaking such a weak password, then, allows an intruder access to one account, from which the intruder exploits system flaws and weaknesses to further compromise the system. The extent of this problem is truly frightening, and some innovative approaches have been designed for dealing with it. One approach (Farmer and Spafford 1990, Spafford 1991, 1992a,b) consists of detecting weak passwords and alerting system administrators and users about them. That is, the system administrator can run a software tool like COPS and alert the users whose passwords are found to be vulnerable. One problem with software tools that pinpoint weak passwords is that, while they are useful to system administrators, they can also be used to attack a system (in the obvious way). More recently, Meunier (1998) explored an approach for generating a quality password (a long enough random string, produced using quality pseudo-random number generation), and then providing a software tool to help the user in remembering the password. While Meunier's approach focused more on mixing music with lyrics and contained only some tentative suggestions on text generation, we have made natural language text generation the main point of the effort.

There is a powerful incentive for users to choose weak passwords (or, in other contexts, weak PIN numbers): they have to remember it——in fact they are instructed not to write their choice on a piece of paper or in a file. So they end up choosing a weak password because it is easy for them to remember. A tool that helps users remember random strings therefore considerably weakens (or even removes) the incentive to choose weak passwords. If users had a handy and attractive software tool that helps them remember a random string, then they would be more likely to choose secure passwords. More widespread use of secure passwords would go a long way towards making it harder to illegally—and easily—break into computer systems. The software tool we are working on helps bring this goal closer.

2.2 NLP Generation of Meaningful Humorous Texts as Mnemonic Devices

The approach we are using can be summarized as this: for any random string Σ, to generate a meaningful natural language text *T* that is a good mnemonic for Σ. The requirements for *T* are:

- it should be easy to extract Σ from *T*: there are many ways of achieving this, including the naive way we have adopted for the first release of using the first letter of every word in *T*;
- *T* itself should be easy to remember: we achieve this by automatically constructing from Σ a *T* that has meaning, eventually of the humorous kind, because funny things are particularly easy to remember, and we are using the results of pioneering research in computational humor to achieve this goal (see Raskin 1985, 1996; Raskin and Attardo 1994).

As far as generating humor is concerned, our approach here differs from existing humor-generation efforts and software (both the original LIBJOG system in Raskin and Attardo 1994 and its spin-offs in Binstead and Ritchie 1997, Hulstijn and Nijholt 1996) in at least four ways:

- a factor that tends to make our problem more difficult is the requirement that *T*, in addition to being "memorable," also corresponds to Σ.
- another complicating factor is that our generation has to use a little more intelligence than, for instance, what extremely little of it is necessary to generate a light bulb joke (Raskin and Attardo 1994) or a cross joke (Binstead and Ritchie 1997) from a standard template.
- a factor that tends to make our problem easier is that the humor generated does not have to be particularly good; a particularly bad joke can be easy to remember precisely because it is so bad (not that the cited toy systems could generate particularly good jokes either!);
- speaking of toy systems, this particular system has a gratifyingly meaningful, non-toy goal.

Below humor generation, there lies a specific natural language generation task, for which we have developed abundant resources in NLP, ready for use or for well-defined tweaking if necessary (see Section 4 below on the NLP resources available for IAS).

2.3 Implementation so Far

For the initial stage of the implementation, we limited the problem and the output in the following helpful ways:

- the accepted input is a random-generated password which is only alphabetical (not numerical and consisting of exactly eight Latin characters, e.g., *shbvwwlo*;
- the generated output corresponds to one primitive jingle tune only;
- the generated text follows the same meter;
- the generated text follows the same grammatical template; and, of course,
- the generated text consists of 8 words beginning, respectively, with the letters in the random string.

The tune goes TA-ta-TA-ta-TA-ta-TA / TA-ta-TA-ta-TA-ta-TA. Accordingly, the meter in each of the two identical lines is 4-foot trochaic, with the 4th foot incomplete. The grammatical template in each identical line is Name, Verb+Past, Name+Poss, Noun. In other words, with W_n, where $1 \le n \le 8$, corresponding, obviously, to the *n*th word in the text, $W_1 = W_3 = W_5 = W_7$ = Name (= Noun+Proper), $W_2 = W_6$ = V+Past, $W_4 = W_8$ = N+Common. $W_{1\text{-}3}$ and $W_{5\text{-}7}$ are all bisyllabic and trochaic, i.e., stressed on the first syllable; $W_{4,8}$ are monosyllabic. Thus, the jingle for the random string above will be, for instance:

Sandra handled Byron's vault.

William wasted Lana's ore.

For all the words, the following constraint is important: their initial sound should be immediately associated with its first letter, i.e., the non-trivial spellings should be excluded, such as, for instance, *ph*, which can be confused with *f*. Specifically for the names, the two additional constraints are that the names should be well-known, wherever possible and not ending in a sibilant (*s, z, sh, zh, ch* or *j*) because those require an extra syllable in the possessive form, e.g., *judges*. Specifically for the verbs, the constraints require that they be unitransitive, i.e., taking one and only one object, that they take both [+Human] subjects and objects, and that they have a regular Past form. Finally, for the monosyllabic nouns, we prefer them concrete, not abstract.

All the three categories are determined by the appropriate word lists with under 10 options for each letter. The lists are also organized in sublists in some cases for the second phase, which we have also partially implemented. At this stage, we want the verbs to be somewhat opposite to each other, e.g., the first one being "positive" and the second one "negative," so that the whole jingle have the effect of somebody doing something good for somebody else in the first line while in the second line something bad is done. This is taken care of by dividing the verb list into two: the good ones and the bad ones, or the first-line verbs and the second-line verbs. We also want the final nouns of the lines to rhyme, and this requires, for each noun on the main word list, 26 rhyming nouns, each beginning with a different letter. Neither of these two tasks has proven to be daunting, and the second-phase version has been easily implemented on a Windows machine in Visual Basic, resulting in 750 lines of code.

The most challenging part of the research is to make the jingle humorous. This part is an implementation of a popular script-based semantic theory of humor (Raskin 1985) which stipulates, basically, that a short verbal joke is ambiguously compatible with two different scripts and that those scripts are opposed to each other in one of the 20 or so prescribed way, e.g., sex / no sex. good / bad, rich / poor, etc. This part of the research is also coordinated with a humorous human-computer interface project, both for entertainment and diversion purposes and for the security-oriented goal of user humor profiling (see Raskin *et al.* 2000, Attardo *et al.* 2000). The verb opposition at the second phase of implementation goes a long way towards the desired humor opposition. We are now working on a method to make the jingles funnier, and part of it is involving the common nouns in the humorous scripts.

We have also conducted a small series of purely illustrative experiments testing the acceptance of the generated lines by users and the extent of the recall. 12 subjects were asked to rate 50 automatically generated lines for acceptance from 1 (unacceptable) to 5 (perfectly acceptable), and the average rank was 3.26. In a different experiment, 7 uninvolved subjects were asked to remember two lines each, and the recall was 100% after 1, 2, and 3 weeks, and it fell down to 93% (one subject forgot one of his two lines) 6 weeks later (and 3 weeks after the subjects had been told that there was no need to remember the lines any longer).

3. NLP for Watermarking

Many techniques have been proposed for watermarking multimedia documents. Many are defective in that they fall prey to attacks that erase the watermark. The most successful operate in the frequency domain, i.e., on the Fourier or Discrete Cosine transform of an image or audio document (see Cox *et al.* 1996, Cox and Miller 1996, and the papers they reference). Of course, such methods do not work on text unless the text is represented as a bitmap image (with, for instance deliberately manipulated kerning and / or spacing to hide the watermark), but in that case the watermark can easily be erased by using OCR (optical character recognition) to change the representation of the text from a bitmap to ASCII or EBCDIC. We would like to come up with a method of watermarking natural language text that is at least as successful as the frequency-domain methods (such as Cox *et al.* 1996 and related work) have been for image and audio.

In one variant of the problem, where the main interest is in hiding a secret message W in text T that looks innocuous and unrelated to the message, T' does not even have to be in the same language as T (see Section 4 below). In the example that follows we assume, however, for the sake of simplicity, that T' is in the same natural language as T.

One idea we want to explore is the use of the theory of quadratic residues (Atallah and Wagstaff 1996). We explain below an adaptation of this method to the problem at hand. We give a simplified (and weaker) version of the scheme we have in mind, to illustrate the main ideas involved.

First we introduce some notation and terminology and review basic facts that will be used later.

Let p be a secret prime at least 20 decimal digits long. The prime p will be used to obtain T', and only by having the prime p can one extract W from T'.

An integer n is a quadratic residue modulo p if p does not divide n and there exists an integer x for which $x\hat{}2 = n \bmod p$. An integer n is a quadratic nonresidue modulo p if p does not divide n and there does not exist an integer x for which $x\hat{}2 = n \bmod p$. Whether n is a quadratic residue or nonresidue modulo p is called the quadratic character of n

modulo *p*. Half of the nonzero (modulo *p*) integers are quadratic residues modulo *p*, and half are quadratic nonresidues modulo *p*.

Using Gauss' quadratic reciprocity law, it is easy to determine the quadratic character of *n* modulo *p* in time *O((log n)(log p))*. The problem is just about as hard as computing the greatest common divisor of two numbers as large as *n* and *p* by using the Euclidean algorithm.

Let the watermark message be the bit string *w_0, w_1, . . . , w_{k-1}.* Let *n_0 , n_1 , . . . , n_{m-1}*, be the integers corresponding to the, e.g., ASCII representations of the words in *T*, i.e., *n_i* corresponds to the *(i-1)*th word in *T*. (Assume for now that *m* is much larger than *k*.)

We use a pseudo-random number generator with *p* as seed to generate random numbers r_0 , *r_1 , . . . , r_{k-1}* (i.e., as many as there are bits in the watermark message). We repeat the following for *i* = 0 , 1 , . . . , *m*-1: If *n_i* + *r_{i mod k}* is a quadratic residue (resp., nonresidue) modulo *p* and *w_{i mod k}* is 1 (resp., 0), then the *(i-1)*th word in *T'* is the same as in *T*. Otherwise we keep modifying the remaining (yet unprocessed) portion of *T* until the *(i-1)*th word satisfies the above requirement. This modification is simple if the *(i-1)*th word has many synonymous words, but otherwise it could be quite elaborate because we must not change the meaning of the text. That not too many modifications are needed follows from the observation, made earlier, that half of the nonzero (modulo *p*) numbers are quadratic residues modulo *p* and half are quadratic nonresidues: this implies that there is a *2^{-t}* probability that *t* modifications will be needed.

The above is necessarily an oversimplification of what we have in mind, but it nevertheless gives the flavor of the ideas involved. Its main drawback is that the watermark can be damaged by an attack that consists of repeatedly performing on *T'* a large number of meaning-preserving modifications like those used to obtain *T'* from *T*, in the first place (such as replacing a word by a synonym); of course the attacker does not know *p* and so each such modification has a 50% probability of actually not damaging the watermark bit hidden at that position. There is no easy way around this: any meaning-preserving modifications we do to obtain *T'* from *T* can be "undone'" by an attacker who is randomly applying many such transformations to *T'*.

The idea we propose for foiling this kind of attack is the following. Instead of doing many meaning-preserving transformations to put the watermark in the text, we make more substantial local changes at far fewer positions, randomly determined by using *p*. While these modifications are not quite meaning-preserving, there are few enough of them that the overall meaning of the original *T* should be essentially preserved. The key idea is that an attacker who wants to remove the watermark must make such changes everywhere (because he does not know where we made them), and by doing so he would essentially destroy *T'*. This is precisely our goal: to make it impossible--or at least as hard and as prohibitively expensive as possible--to remove the watermark from *T'* without essentially destroying the meaning of *T'*. At this time, we do not know how to implement this idea without making use of the original (non-watermarked) text *T* for retrieving *W* from *T'*, and we are not sure that it is even feasible--or necessary.

The whole idea is still vulnerable, however: as long as the quadratic characters of words are used to encode the watermark, it can still be removed by an attacker doing systematic synonym-substitutions--unlikely and costly as this kind of attack may be to stage, it is still not enough of a deterrent. Somewhat more resilient would be an encoding that ties the watermark to a hidden element of the text, such as the tree that represents the sentence structure, rather than to the actual words in the sentence: for example, if the watermark is in the quadratic characters of the items in a postorder listing of the preorder numbers (+ *p*, the secret large prime number, to make these numbers large for statistical purposes) of the above-mentioned tree, it would resist word-substitutions and any transformation that preserves the structure of the tree - even if it changes the labels of the tree nodes.

Even more resilient would be a scheme that, instead of using the above-mentioned tree, uses some graphical representation of meaning (this is parallel to Palsberg's 2000 approach to watermarking the data structure and not the actual code in software watermarking). The text-meaning representation (TMR) of the text (see Section 4 and Appendix 2) is, of course, the obvious candidate. The possibility of representing the entire text to be watermarked as its TMR and watermarking the TMR instead seems to have a promising venue that needs to be explored. The TMR of a sentence is a much longer string of words containing an implicit tree (actually, tree of trees); it is completely invisible to the attacker; a larger variety of watermarking techniques may hence be deployed. It is, however, more difficult to make replacements in the TMR.

We are also interested in how to get rid of storing the original (non-watermarked) text. Extending our overall approach so that it does not require the original text is certainly an attractive venue of research. What it means is that we will then need the watermarked text to "self-synchronize" by indicating the positions at which the watermark is placed (assuming we place the watermark in relatively few places). This can be done by using a peculiarly shaped tree (that is secret just like the prime *p*) to effectively say, “Here starts another watermark.”

A number of problems we are aware of and some, we suppose, we are not yet will arise here. One is that of “false positives”: if the property of a sentence signalling that the next sentence contributes to the watermark may recur elsewhere in the text the process of prosucing the watermark, for instance, in court, will be compromised. Generally, speaking, linguistic entities at and above the level of words (e.g., sentences) have an extremely low probabilities, so the occurrence of a false positive there is highly unlikely. The possibility should be somewhat more alarming, however, with regard to syntactic trees because their inventory, while still technically infinite, consists of a small number (< 20) of standard node junctures that recur throught the trees. The semantic trees in the TMRs (see also Appendix 2) do regain their uniqueness and extremely low probabilities of recurring, especially in the same text.

Yet another problem is excerpting: will the watermark hold if a part of the text is excerpted? We address this problem by inserting the watermark in a very small number of sentences throughout the text and repeating it as many times as the length of the sentence allows, making sure that the probability of a hostile action removing all the multiple occurrences of every bit of the watermark string be extremely low.

From the NLP perspective, these watermarking techniques will involve mostly rather a simple natural-language generator of pretty short utterances based either on a substitution of a word by its synonym or, in rare cases, of a phrase by a synonymous phrase. One unusual and almost completely unexplored aspect of it is this partial generation of a substitute word or phrase not on the semantic basis but rather on the basis of the necessary quadratic residue or non-residue value. Here, the system will generally have several choices to make among words which will contribute the same way to the code, and this makes the problem very feasible (again, see Section 4 on the NLP resources). A desirable side effect is that partial natural language generation, in which we are making rapid progress within this research will contribute to many non-IAS-related NLP tasks.

4. Machine Translation (MT) Techniques for Information Security

To capture the difficulty of breaking a natural language based scheme, we would like to take advantage of the serious progress in meaning-based machine translation achieved by the ontological-semantic approach, where a powerful system of language resources has been created by two of the coauthors and their associates to enable Level 3 (top level, with full meaning access) MT for a growing number of natural languages (Nirenburg and Raskin 1987, 1996, 1998; Onyshkevych and Nirenburg 1995; Mahesh 1996, Viegas and Raskin 1998). These resources include:

- semi-automatically acquired ontology, both general and domain-specific, for over 60,000 nodes and properties;
- semi-automatically acquired lexicons for a growing number of natural languages (already over a dozen at this writing) for over 40,000 word senses;
- an analyzer which translates a text in a natural language into an text-meaning representation (TMR, a language-independent interlingua which represents the meaning of the text);
- a generator which translates a statement in TMR into a text in a given natural language.

In MT, the analyzer goes first and the generator follows. In IAS, the order is reversed. Otherwise, the processes are identical, and the same resources are usable for both purposes.

To combine an additional MT layer with another scheme (perhaps the one described in the previous section), we are developing a compact package of the above resources, trimmed and simplified for the purpose, to be distributed to the system users. In a new and original twist from normal NLP analysis and generation, we are exploring a specially biased MT, distorted by a secret key, which adds another protective layer. The MT layer will be impenetrable without an exact copy of the key (and, of course, MT software), due to the idiosyncratic rules of each of the participating languages.

At present, we have the following capabilities:

- we have Level 2 MT systems for a number of uncommonly known languages along with a semi-automatic system for rapid developments of such systems for other low-density (i.e., not widely used) natural languages, and we can ensure Web access to such systems;
- we can automatically translate the text *T* of a message in English that needs to be transmitted into text *T*’ in a low-density language before encrypting it in any other way; we can complicate it further by translating T’ into T” in yet another low-density language, and so on; and we can vary those languages within our inventory from one transmission to another;

- we can automatically translate messages in a deliberately distorted way while still preserving the appearance of a meaningful text; the distortion may range from the primitive substitution of (selected) words with antonyms to much more sophisticated manipulations on the lexicon;
- we can cause even more complex distortions of texts, still keeping them meaningful and cohesive, by manipulating the ontological nodes evoked by the words in *T*, and only access to the specific ontology will help figure out what *T* is;
- we can also manipulate the analyzer and generator for the same purpose.

The main thrust of this venue in our research is to evaluate the effectiveness, economy, and reliability of these capabilities for IAS, in other words, how simple the system will be for the authorized users to deploy and how hard it will be for the adversaries to break the code. This approach comes the closest to the Navajo antecedent but it takes that ancient method to the contemporary computational-linguistic level by using combinations of the best NLP resources available.

5. Downgrading, or Sanitizing Information

5.1 Downgrading vs. Declassification

We are applying state-of-the-art methods of ontological semantics to the problem of sanitizing classified and sensitive documents as per pre-defined specifications, ranging from banning the use of a specific term to more complex, contextual restrictions. The problem is essentially the same as that of declassification of classified government documents, a task required by the Executive Order 12958, Classified National Security Information, and pretty much swept under the rug so far by the government agency because of the presumed non-implementability. The rules of downgrading are complex and the human implementation of the procedure prohibitively costly and slow. The need to automate the procedure is obvious but simple, keyword-plus-Boolean-logic-based approaches result in an unacceptable level of accuracy.

We are developing a much more accurate method of automatic downgrading based on the ontological semantic resources accumulated at the Purdue NLP Lab and at CRL for a number of DoD-funded projects in machine translation, information extraction and retrieval, summarization, and other related tasks. The method allows the system to accommodate complex contextual rules of downgrading (or declassification) from the human manuals and to model the activity of a competent human declassifier at many times the speed and with no compromise in accuracy. EO12958, signed by President Clinton on April 17, 1995, and obligating all government agencies to declassify documents older than 1976 by April 2000, requires that all government documents be declassified and made available to the public unless they continue to contain sensitive information, in which case parts of the documents should be deleted or blanched or the entire documents barred from declassification. The various affected departments, facing the necessity to declassify virtually billions of pages, have developed pretty complex sets of rules for human declassifiers (the DOE book of rules, for instance, is reported to contain over 700 different, often overlapping instructions). The work has been going very slowly, with each document requiring weeks of processing, checking, and double checking, and the departments are agencies are falling behind in the implementation of the Act.

A computational solution of the backlog is simple in those rare cases when the declassification rule states that no document with a certain word occurring in it (or two words occurring together) can be declassified at this stage. Ingenious techniques have been proposed for pretty complex Boolean algebra formulae of keywords to capture the “contextual” use of the significant words. The level of accuracy reached in such systems (see, for instance, ULTRASTRUCTURE developed by George Washington University’s Declassification Productivity Research Center or SRI International’s Keyword Spotting methodology) remains unacceptably and apparently unimprovably low.

It is clear, at this point, that the Executive Order will not be implemented on time or at all. The problem of downgrading, while essentially the same, is seen by the government agencies and industrial entities as not an act of coercion but rather an absolute necessity, which it is. The purpose is not to serve the anonymous public out of some good intentions but rather to get very important work done. It is downgrading, therefore, and not declassification, that our research specifically addresses and targets. We do hope, however, that our success in downgrading will change the agencies’ attitude to declassification as well.

5.2 The Ontological Approach: An Example

We assume that the example in this section is reasonably self-explanatory. We do, however, provide a short sketch of ontological semantics, with further references, in Appendix 2.

Let us consider then a hypothetical example, whose purpose is to illustrate a typical downgrading instruction and how the proposed system will handle it. Let us assume that the system is instructed to allow mentions of nuclear submarines but not their specific deployment, reactor capacity, or mode of refuelling.

Focusing just on the first of these for the moment, instructing the computer to look for *nuclear submarine* and *deploy* in the text will fail the instruction in many different situations, e.g., when both occur in the text but not in the same sentence or adjacent sentences, and *deploy* does not pertain to the vessel. At the same time, the classified information may be given without using any form of *deploy* but rather with such words as *location*, *is*, or even simply *at*. It is very hard to anticipate all the synonymic substitutions and paraphrases as well as permitted uses with just keyword combinations.

One would think that simple syntactic parsing will solve at least the question of whether *deploy* pertains to *nuclear submarine* in a sentence, But, besides the fact that syntactic parsing needs then to be used globally and rather expensively, it will misinterpret such a simple sentence as *Nuclear submarines will be deployed nearby to support ground forces in case of military emergencies.*

What we propose is the limited use of meaning-based text analysis developed at the CRL for machine translation, information retrieval and extraction, summarization, intelligent Web searches, and other related NLP tasks (see Section 4 above). In our experience, information retrieval requires only a partial use of these resources and processes for two different reasons: first, the system is only interested in a few expression or concepts; second, no generation is necessary

In our hypothetical example involving the occurrence of nuclear submarine in three narrowly defined context, this is how the system utilizes the resources. When the analyzer spots *nuclear submarine* or just *submarine* in a sentence, it immediately evokes the appropriate lexical entry, which, in turn, produces the corresponding ontological concept, which will look, in much simplified form, as follows:

submarine	
(isa	warship)
(theme-of	build, commission, decommission, deploy, destroy, attack)
(instrument-of	attack, support, transport, threaten)
(manned-by	naval crew)
(propel-mode	surface, sub-surface)
(engine-type	nuclear-engine)
(range	N < x < M)
(speed	K < y < L)
(current-location	body-of-water and / or geographic point and / or coordinates and / or rela tive, time-range)
(prior-location	body-of-water and / or geographic point and / or coordinates and / or rela tive, time-range)
(next-location	body-of-water and / or geographic point and / or coordinates and / or rela tive, time-range)
(current-mission	Z)

Most of the slot fillers (as well as some of the slot names) are ontological concepts as well. In MT, the analyzer attempts to utilize the syntactic structure of the sentence and the lexical items in it to fill all the slots it can. In the proposed system, it is interested only in the filler for the location slots and perhaps, more narrowly, only the current location (as well in the TYPE-OF-ENGINE slot if its filler is NUCLEAR-ENGINE - this will be established immediately if the triggering string is *nuclear submarine*; if it is just *submarine* the analyzer will look around for *nuclear* or equivalents). If the slot is empty, the red flag will not come up and the text will be passed down unmodified--unless any one of the other two restricting contexts is present.

The system cannot access the other two contexts from the SUBMARINE concept alone but it can from the concept of NUCLEAR-ENGINE used as the filler for TYPE-OF-ENGINE property slot. The property slots for NUCLEAR-ENGINE will include reactor-capacity and refuel-mode, and the analyzer succeeds in filling either of these slots, the sentence (or, depending on the downgrading instructions, the entire document) will be barred from the downgraded version.

This general approach is then further refined in conjunction with the exact nature of each set of the current and future human-to-human instructions for downgrading, or sanitizing sensitive information in a mixed network. This adjustment is implemented for each system with the help of a Web-accessed semi-automatic system for knowledge engineering, similar to the environment we have developed for the rapid deployment of MT systems for low-density natural languages (see Section 4 above and Nirenburg and Raskin 1998).

This approach has turned out to be highly effective in accurate knowledge representation of even the most arcane simulated human-to-human instructions on declassification, resulting in high accuracy in automatic declassification and downgrading in our preliminary runs. It is clear, however, that the real efficacy of the ontological semantic approach to automatic sanitizing of text will have to be measured more carefully in full-fledged testing and evaluation procedures, and we are working on subjecting the system to several such procedures, common to IAS, at the earliest stage of completion.

It is noteworthy here also that other, non-NLP-related emerging paradigms in IAS are beginning to develop a need for ontologies, for instance, for a consistent and meaningful formal representation and classification of attacks and responses (see Templeton and Levitt, this volume). Such paradigms can gain from the substantial body of knowledge on the acquisition, mathematical properties, and effective interchange of ontologies (see, for instance, Nirenburg and Raskin 2001, Ch. 3.3).

6. Conclusion

We have discussed just four possible venues for utilizing the available NLP expertise and resources for IAS. While all these directions were addressed in the exploratory mode, the techniques of simplified NLP text generation have already been adapted for and implemented for the development of mnemonic devices for strong, random-generated system-access passwords; the innovative and promising techniques of severely constrained NLP text generation are being developed for watermarking; and the techniques of ontological semantics are being applied to and modified for downgrading sensitive textual information in mixed networks. We fully expect other applications of NLP for IAS to develop in the process of cooperation between these two areas and to contribute to innovative twists in enhancing computer information assurance and security.

Acknowledgments

This work has been supported in part by the Department of Defense under contract number MDA-904-92-C-5189 and by an Eli Lilly grant to CERIAS, Purdue University. Raskin is grateful to the Trustees of Purdue University for having permitted him to serve as a PI-level consultant for CRL/NMSU since 1994. He is also indebted to Carl Landwehr of Mitretek and Cathy Meadows of Naval Research Laboratory, both visiting speakers at the CERIAS Information Security Seminar in the Fall semester of 1999-2000, for private conferences, from which he formed the idea of downgrading as the major application of NLP to IAS at this time. The coauthors have profited significantly from the questions and comments made at the discussion time by the NSPW 2000 participants which enriched and improved this final draft and influenced subsequent research, especially in the watermarking strand. Finally, the coauthors greatly appreciate the CERIAS (http://www.cerias.purdue.edu/) Founder and Director Gene Spafford's "ball of energy" encouragement of their work on this project and his and his staff's competent administrative and logistic support.

References

Attardo, S., V. Raskin, and O. Stock 2000. Computing Humor for a Human-Computer Interface. IRST-CERIAS Technical Report (forthcoming).

Atallah, M. J., and S. S. Wagstaff 1996. Watermarking Data Using Quadratic Residues. Working Paper, Department of Computer Science, Purdue University.

Binstead, K., and G. Ritchie 1997. Computational rules for generating punning riddles. *Humor: International Journal for Humor Research*, 10:4, pp. 25-76.

Cox, I. J., J. Kilian, F.T. Leighton, T. Shamoon 1996. Secure spread spectrum watermarking for images, audio and video. **International Conference on Image Processing, Vol. 3**, pp. 243--246.

Cox, I. J., and M. L. Miller 1996. A review of watermarking and the importance of perceptual modeling. **Proc. SPIE - Int. Soc. Opt. Eng., Vol. 3016**, pp. 92--99.

Framer, D., and E. H. Spafford 1990. The COPS security checker system. **Proceedings of the Summer Conference**, Berkeley, CA: Usenix, June.

Hulstijn. J., and A. Nijholt (eds.) 1996. **TWLT 12. Automatic Interpretation and Generation of Verbal Humor: IWCH '96**. The Hague: CIP Gegevens Koninklijke.

Mahesh, K. 1996. Ontology Development for Machine Translation: Ideology and Methodology. Memoranda in Computer and Cognitive Science, MCCS-96-292. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.

McDonough, Craig J. 1999. Jingle Software for "Sing-A-Password" for Windows, Version Beta 2, http://omni.cc.purdue.edu/~raskin/jingle.exe.

Meunier, Pascal C. 1998. Sing-a-Password: Quality Random Password Generation with Mnemonics. Description by the author at http://home.sprynet.com/sprynet/meunierp/singit.html}. Latest (Macintosh) version available for download at ftp://ftp.realsoftware.com/developer_examples/applications/.

Nelson, R. 1997. What is a secret and what does it have to do with computer security? **Proceedings. 1994 SIGSAG New Security Paradigms Workshop. Little Compton, RI**. Los Alamitos, CA: IEEE Computer Society Press, pp. 74-79.

Nirenburg, S., and V. Raskin 1987. The subworld concept lexicon and the lexicon management system. *Computational Linguistics*, 13:3-4, pp. 276-289.

Nirenburg, S., and V. Raskin 1996. Ten Choices for Lexical Semantics. Memoranda in Computer and Cognitive Science, MCCS-96-304. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.

Nirenburg, S., and V. Raskin 1998. Universal grammar and lexis for quick ramp-up of MT systems. **Proceedings of ACL/COLING '98, Vol. 2**. Montreal: University of Montreal, pp. 975-979.

Nirenburg, S., and V. Raskin 2001. **Principles of Ontological Semantics**. Cambridge, MA: MIT Press (forthcoming).

Onyshkevych, B., and S. Nirenburg 1995. A lexicon for knowledge-based MT. *Machine Translation*, 10:1-2, pp. 5-57.

Palsberg, J. 2000. Software watermarking with Secret keys. Paper at the CERIAS Annual Research Symposium on “Advancing the State and Practice of Information Assurance and Security,” Purdue University, W. Lafayette, IN, April 21.

Raskin, V. 1985. **Semantic Mechanisms of Humor**. Dordrecht-Boston, D. Reidel.

Raskin, V. 1996. Computer implementation of the general verbal theory of humor. In Hulstijn and Nijholt, pp. 9-19.

Raskin, V., and S. Attardo 1994. Non-literalness and non-bona-fide in language: Approaches to formal and computational treatments of humor. *Cognition and Pragmatics*, 2:1, pp. 31-69.

Raskin, V., S. Nirenburg, and O. Stock 2000. Humor in Human-Computer Interface. Panel, The Art Gliner Center of Humor Studies, University of Maryland, College Park, MD, January 28, http://www.otal.umd.edu/amst/humorcenter/.

Spafford. E. H. 1991. Preventing weak password choices. **Proceedings of the 14th National Computer Security Conference**, National Institute of Standards and National Security Institute, pp. 446-455.

Spafford, E. H. 1992a. Observing reusable password choices. **3rd USENIX UNIX Security Symposium**, The USENIX Association. pp. 299-312.

Spafford, E. H. 1992b. OPUS: Preventing weak password choices. *Computers & Security*, 11:3, pp. 273-278.

Templeton, S., and K. Levitt 2001. A Requires/Provides Model for Computer Attacks. This volume.

Viegas, E., and V. Raskin 1998. Computational Semantic Lexicon Acquisition: Methodology and Guidelines. Memoranda in Computer and Cognitive Science, MCCS-98-315. Las Cruces, NM, New Mexico State University, Computing Research Laboratory.

Appendix 1: Natural Language and NLP

‘Natural language’ is the term used to denote all human languages that have evolved in the course of human history and been used monolingually to serve fully the needs of the communities that employ them. There are around 5,600 extant natural languages, e.g., live languages and dead languages, for which there are records and/or descriptions

Natural language is the basis of all human activity and, therefore, an important component of many fields of study. The main discipline, however, which studies natural language as a universal human faculty and the most observable function of the human mind is linguistics. One of the most ancient academic disciplines, dating back several millennia and associated closely, in antiquity, with the study of foreign languages and preservation of dying languages and sacred texts in them, linguistics finally won its independence from such related fields as philosophy, theology, and philology early in the 19th century by developing a particular historical domain that was of little interest to other fields, viz., the study of language families and, later in the century, reconstruction of ancestor languages, such as proto-Indo-European, with the help of a well-defined methodology based on a systematic comparison of germane words in descendant languages.

This emphasis on precise methodologies served linguistics well in the 20th century when a momentum for an algorithmic, mathematicalized study of language developed in the 1920s, with the advent of structuralism and the subsequent emphasis on synchronic description of language, largely at the expense of its historical exploration. The American branch of structural linguistics, Leonard Bloomfield’s descriptive linguistics, became most influential in the 1940-1950s after introducing a near-algorithmic procedure for a field description of an unknown language, starting with the collection of a corpus and continuing with the recursive application of the segmentation and distribution procedures from the lower to higher levels of language structure.

These levels define the central subdisciplines of linguistics. Phonetics and phonology study the sound of language: the former empirically so and the latter theoretically. Morphology studies such parts of the words as the root, prefix, suffix, and infix, as well as the words themselves and their groupings into parts of speech, such as nouns, verbs, prepositions, etc. Syntax is concerned with the complex organization of words into phrases, clauses, and sentences. Semantics deals with the meaning in natural language; pragmatics with meaning in context.

The mathematicalization, or more accurately, formalization of linguistics was institutionalized by Noam Chomsky in the late 1950s, whose twofold contribution included the development of the theory of formal grammars as a branch of mathematical logic and of transformational generative grammar as an implementation of a formal grammar. A finite set of several types of grammatical rules is postulated as the syntax of a language, and the syntactic structure of each sentence in the language is a formal derivation resulting from a consecutive application of a subset of these rules to the initial symbol *S* (sentence).

The primary function of natural language is communication among humans. There are several different modes of human-human communication. It is customary, after Paul Grice's influential work in the 1950-70s, to think of the fact-conveying, bona-fide mode of communication as primary. In this mode, the speaker and hearer are committed to the literal truth of what is said. A slight and necessary extension of the mode allows such non-literal devices as metaphors and implicatures as long as the speaker makes it clear to the hearer how the utterance is to be understood. This mode is based on complete linguistic cooperation—and trust—between the speaker and hearer. Other, non-bona-fide modes of communication, such as humor, play-acting, advertising, propaganda, and lying, have their own principles of cooperation but none of the modes implies a commitment to the truth of what is being said.

All of these modes have to contend with two essential characteristics of natural language that complicate human-human communication, namely, underspecification, sometimes rather misleadingly referred to as 'vagueness,' and ambiguity. Underspecification of reality by language means simply that no utterance is capable of containing all the details of the situation it attends to describe. Thus, when the speaker says, "John was late for the calculus class this morning," those hearers who know who John is and what calculus class is meant will have a feeling of complete understanding. The sentence, however, leaves an infinite number of questions unanswered, and humans intuitively use presuppositions (prior knowledge) and inferences (subsequent, derived knowledge) to answer these questions in their own minds.

Underspecification is accepted by native speakers (and hearers) as a fact of life and is rarely commented upon or much researched. (It cannot, however, be accepted by the computer.) Ambiguity is a different matter: much of what native speakers do with language is a pretty sophisticated and almost entirely unconscious procedure of disambiguation. The fact is that just about every word in a natural language has multiple meaning, many syntactic structures can be analyzed in two or more ways, and as an obvious result of that, sentences, which are made up of words put together by syntactic structures, tend to be at least potentially many ways ambiguous. Thus, the much-analyzed sentence *The man hit the colorful ball* is believed to be 4-way ambiguous; the specially concocted of hackneyed semantic examples *The paralyzed bachelor hit the colorful ball* is 25-way ambiguous.

Native speakers negotiate this potentially disastrous situation skillfully by using sentences in disambiguating contexts and by providing to the hearers the extra information needed to understand each sentence in the intended meaning. Sure enough, this sometimes falls through and misunderstandings occur, but because these are unacceptable to speakers, every effort is made, in the process of Gricean cooperation, to prevent ambiguity from affecting comprehension. The problem of ambiguity, along with the matching problem of underspecification looms much more seriously in NLP.

Much, if not already most human productive activity is conducted now with the help of computers. Human-computer interaction and the human-factors aspect of computer development and use have, therefore, achieved paramount importance. NLP is the application of linguistics to the study of these phenomena through its application. NLP designs and implements automatic systems that, typically, take text in a natural language as input, process it according to the predefined tasks, and then generate output, which may be in the same or another natural language or in some other stipulated format, such as, for instance, a database report or a chart.

Historically, the first task, for which NLP was used in the early 1950s, was machine translation (MT) between pairs of such best-known and –described languages as English and Russian or English and French. While realizing the impossibility of simple word-for-word translation and devising an ingenious system of overcoming this difficulty syntactically, MT soon ran into the "semantic barrier": it became clear to the MT pioneers that no high-quality fully automatic translation was possible without building the understanding of the text into the system, and this was not considered feasible at the time—to some extent, because of ambiguity and underspecification. As a result, MT degraded to human-assisted post-editor based systems, some of which were found to be practically useful enough to last for decades, and even to machine-aided translation systems, in which humans performed all the intellectual tasks and computers performed simple but time-consuming tasks, such as dictionary look-ups. This temporary collapse of MT as a naive ambitious goal of replacing a huge army of expensive human translators and interpreters with machines gave both MT and NLP a bad name in other fields in the 1960-70s, which persisted longer than it should have in spite of the crucial changes in NLP and crucially contributed to the myth of the non-feasibility of NLP that we mentioned in Section 1.

In the mid-1980s, however, MT started making a spectacular comeback as a knowledge-based, meaning-oriented application. This was due partly to progress in general AI and partly to the increasing role of computational semantics, which has moved from the fledgling efforts of its pioneers in the 1970s to the breakthrough developments of the early 1990s in handling ambiguity and underspecification and on to its currently dominant position in NLP. Contrary to the prevalent NLP ideology of the late 1960s-late 1980s, when an enormous amount of talent, effort, and funds were invested in avoiding computational semantics, usually through increasingly detailed syntactic parsing, the current state of the art easily assumes the necessity for the computer to understand the meaning of the text it processes. This makes it possible to diversify NLP applications to intelligent information retrieval and Web searches, automatic abstracting, summarization, etc. In computational semantics, input text is converted into its meaning representation at the required level of granularity, the system manipulates these representations according to the prescribed task and then converts them into natural language text or any other format. It is the meaning-based ontological-semantic approach that we apply to AIS in our research.

NLP can be seen, in security terms, as the effort of decoding the meaning of a text from its surface form, something that the German team failed to achieve with Navajo texts in World War II. What the team lacked was the knowledge of the language and not, unlike in “ordinary codes,” some sophisticated combinatorics of a reasonably high order of mathematical complexity. There is, of course, a key to this code but it amounts to the entire structure of the language from its phonological level up to the pragmatical level of meaning in context, and an NLP system can be only successful if it is based on all this knowledge, which is rather hard to describe and present formally. But these difficulties, which NLP has had to contend with for decades, become an advantage in AIS if the language structure is used to encode, to encrypt the transmitted message. This is the updated and upgraded intuition, much expanded from the Navajo incident and reinforced with NLP, that underlies all of our venues of NLP / AIS research.

Appendix 2. A Sketch of Ontological Semantics

Ontological semantics is an approach to NLP which uses a constructed world model, or ontology, as the central resource for extracting and representing meaning of natural language texts as well as synthesizing natural language texts based on representations of their meaning. The architecture of a prototypical application of ontological semantics comprises, at the most coarse-grain level of description,

- a set of static knowledge sources, namely, a single language-independent ontology and a lexicon connecting the ontology with a natural language (one such lexicon is needed for each language in an application), so that each lexical entry is explicitly anchored in the ontology, pointing to a node in it or a property of a node;
- knowledge representation languages for specifying meaning structures, ontologies and lexicons; and
- a set of processing modules, namely, a semantic analyzer and a semantic text generator.

In ontological semantics, the module that produces specifications of input for a text generator is the semantic analyzer. This means that ontological semantics directly supports such applications as machine translation of natural languages. However, the approach in principle supports other applications, such as information extraction, text summarization, support of networks of human and software agents, etc. An additional reasoning module that a) manipulates meaning representations produced by the analyzer; and b) uses other methods of producing meaning representations fit to be inputs to semantic generation can be added to the basic architecture. The static knowledge sources in ontological semantics are capable of serving the knowledge needs of such a module.

Any large, practical, multilingual computational-linguistic application, such as machine translation, requires many knowledge and processing modules integrated in a single architecture and control environment. For maximum output quality, such comprehensive systems must have knowledge about speech situations, goal-directed communicative actions, rules of semantic and pragmatic inference over symbolic representations of discourse meanings and knowledge of syntactic, morphological and phonological / graphological properties of particular languages. Heuristic methods, extensive descriptive work on building world models, lexicons and grammars as well as a sound computational architecture are crucial to the success of this overall paradigm. Ontological semantics is responsible for a subset of these capabilities. The approach is also based on the important assumption that it is possible to integrate processing modules based on unconnected theories through matching their input and output structure formats.

Historically, ontological semantics has been developed by two of the coauthors and their associates in several completed NLP projects at Colgate University, Purdue University, Carnegie Mellon University, and New Mexico State University in 1982-1995. It has also been used in the current Expedition / Boas, MINDS, Savona, and TIDES projects at CRL (http: / / crl.nmsu.edu /). The goals of the practical semantic theory for NLP are somewhat compatible with and intersect the claimed goals of some other approaches to purely theoretical compositional semantics. There are also important differences along at least the following two dimensions: a) the domain of the theory (e.g., lexical and compositional semantics, syntax, morphology, pragmatics, reasoning); and b) the degree to which the theory has been actually developed through language description and computer system construction. Ontological semantics is also indebted to the various approaches to processing meaning in artificial intelligence, among them conceptual dependency, preference semantics, procedural semantics and related approaches.

Our theoretical work in semantics is devoted to developing not so much a general semantic theory but rather a semantic theory for natural language processing. Therefore, issues of text meaning representation, semantic (and pragmatic) processing and the nature of background knowledge required for this processing are among the central topics of our effort. A number of differences exist between the mandates of general semantic theory and semantic theory for NLP. In what follows we suggest a number of points of such difference.

While it is agreed that both general and NLP-related theories must be formal, the nature of the formalisms can be quite different because different types of reasoning must be supported. A general linguistic theory must ensure a complete and equal grain-size coverage of every phenomenon in the language; an NLP-related theory analyzes only as much as is needed for the purposes of a particular application. The ultimate criterion of validity for a general linguistic theory is explanatory adequacy; for an NLP-related theory it is the success of the intended application. A general linguistic theory can avoid complete descriptions of phenomena once a general principle or method has been established. A small number of clarification examples will suffice. In NLP the entire set of phenomena present in the sublanguage of an application must be covered exhaustively. A general linguistic theory has to be concerned about the boundary between linguistic and encyclopedic knowledge. This distinction is spurious in NLP-oriented semantic theories because in order to make semantic (and pragmatic) decisions a system must have access equally to both types of data.

While a general linguistic theory can be method-driven, that is, seek ways of applying a description technique developed for one phenomenon in the description of additional phenomena (this reflects the predominant view that generalization is the main methodology in building linguistic theories), an NLP-related theory should be task-driven—which means that adequacy and efficiency of description takes precedence over generalization.

As any semantic theory for natural language processing, ontological semantics must account for the processes of generating and manipulating text meaning. An accepted general method of doing this is to describe the meanings of words and, separately, specify the rules for combining word meanings into meanings of sentences and, further, texts. Hence the division of semantics into lexical (word) semantics and compositional (sentence) semantics. Semantics for NLP must also address issues connected with the meaning-related activities in both natural language understanding and generation by a computer. While the semantic processing in these two tasks is different in nature—for instance, understanding centrally involves resolution of ambiguity while generation deals with resolution of synonymy for lexical selection—the knowledge bases, knowledge representation approaches and the underlying system architecture and control structures for analysis and generation can be, to a realistic degree, shared.

In ontological semantics, the text-meaning representation (TMR) is derived through:

- establishing the lexical meanings of individual words and phrases comprising the text;
- disambiguating these meanings;
- combining these meanings into a semantic dependency structure covering
  - the propositional-semantic content, in-

cluding causal, temporal and other relations among individual statements;
  - the attitudes of the speaker to the propositional content; and
  - the parameters of the speech situation;
- filling any gaps in the structure based on the knowledge instantiated in the structure as well as on ontological knowledge.

The final result of the process of text understanding may include some information not overtly present in the source text. For instance, it may include results of reasoning by the consumer, aimed at filling in elements required in the representation but not directly obtainable from the source text. It may also involve reconstructing the agenda of rhetorical goals and plans of the producer active at the time of text production and connecting its elements to chunks of meaning representation.

Early AI-related natural language understanding approaches were criticized for not paying attention to the halting condition of meaning representation. They were open to criticism because they did not make a very clear distinction between the information relayed in the text and information retrieved from the understander's background knowledge about the entities mentioned in the text. This criticism is only valid when the program must apply all possible inferences to the results of the initial representation of text meaning and not when a clear objective is present, such as resolution of ambiguity relative to a given set of static knowledge sources, beyond which no more processing is required.

It follows that the meaning is, on this view, a combination of the information directly conveyed in the NL input; the (agent-dependent and context-dependent) ellipsis-removing (lacuna filling) information which makes the input self-sufficient for the computer program to process; and pointers to any background information which the system expects might be brought to bear on the understanding of the current discourse, the formation of a record (a "memory") of the discourse in the discourse participants' episodic memory or generation of further discourse turns.

Additionally, text understanding in this approach includes detecting and representing a text component as an element of a script / plan or determining which of the producer goals are furthered by the utterance of this text component. We stop the analysis process when, relative to a given ontology, we can find no more producer goals / plans which can be furthered by uttering the sentence. But first we extract the propositional meaning of an utterance using our knowledge about selectional restrictions and collocations among lexical units. If some semantic constraints are violated, we turn on metonymy, metaphor and other ``unexpected" input treatment means. After the propositional meaning is obtained, we actually proceed to determine the role of this utterance in script / plan / goal processing. In doing so, we extract speech act information, covert attitude meanings, and eventually irony, lying, etc.

There is a tempting belief among applied computational semanticists that in a practical application, such as MT, the halting condition on representing the meaning of an input text can, in many cases, be less involved than the general one. The reason for this belief is the observation that when a target language text is generated from such a limited representation, one can in many cases expect the consumer to understand it by completing the understanding process given only partial information. Unfortunately, since without human involvement there is no way of knowing whether the complete understanding is, in fact, recoverable by humans, it is, in the general case, impossible to posit a shallower (and hence more attainable) levels of understanding. To stretch the point some more, humans can indeed correctly guess the meaning of many ungrammatical, fragmentary and otherwise irregular texts. This, however, does not mean that an automatic analyzer, without specially designed extensions, will be capable of assigning meanings to such fragments—their semantic complexity is of the same order as that of "regular" text.